



Politechnika Warszawska



OKNO.pw.edu.pl

Ośrodek Kształcenia na Odległość

Architektura Systemów Komputerowych



dr inż. Paweł Wnuk

CZEGO SIĘ DOWIECIE W TRAKCIE TYCH ZAJĘĆ ?

Witamy serdecznie.

Pozwolicie, że na samym początku posłużymy się cytatem z książki Alana Coopera "Wariaci rządzą domem wariatów":

"Uciekajcie, komputery atakują! Przerażającym, potężnym komputerom powierza się coraz ważniejsze zadania, wykonywane za pomocą niewygodnych, staroświeckich interfejsów. W miarę ich ingerowania w nasze codzienne życie będą nas drażnić, irytować, a może nawet kogoś zabiją. Odczujemy chęć rewanżu, zniszczenia ich. Nie odważymy się jednak, gdyż będziemy już całkowicie, nieodwracalnie zależni od tych potworów, bo dzięki nim życie we współczesnym świecie stało się możliwe."

Tak ... straszna wizja, straszne słowa, napisane w dodatku przez informatyka ... ale skoro nie możesz czegoś zwalczyć - zrozum i polub to. Co do lubienia - to kwestia osobowa ... natomiast postaramy się pomóc Wam zrozumieć.

Zaczynamy właśnie kurs informatyki pierwszej - systemów komputerowych. Zadaniem tego przedmiotu jest wprowadzenie Was w zagadnienia informatyczne, które potem dokładniej poznacie na przedmiotach dedykowanych poszczególnym zagadnieniom. W zależności od toku i typu studiów w których uczestniczycie - poznacie podstawy programowania, zarówno strukturalnego jak i obiektowego, budowę systemów operacyjnych, zagadnienia związane z sieciami komputerowymi, techniki wykorzystywane w internecie, bazy danych, techniki sztucznej inteligencji, itd ... Natomiast podczas tego kursu postawiliśmy sobie dwa cele:

- przekazać Wam wiedzę o tym jak działa komputer. Jak jest zbudowany, jakie są jego elementy, w jaki sposób wykonuje programy (i co to są w ogóle programy), co oznaczają poszczególne określenia z którymi będącie się (jako informatycy) później spotykali. Postaramy się wyjaśnić większość z magicznych TLS (Trzy Literowy Skrót ;)) którymi tak często posługuje się ta dziedzina wiedzy;
- nauczyć Was bezpiecznego i świadomego korzystania z komputera. Zanim staniecie się specjalistami w wybranych dziedzinach informatyki, chcielibyśmy byście potrafili dobrać i dostosować do Waszych potrzeb podstawowe narzędzie - system komputerowy.

Pamiętajcie, że systemy komputerowe stanowią dla nas - powtarzając za "Bombą megabitową" Stanisława Lema - *Artificial servility* ... czyli sztucznych niewolników:

"[...] sztuczne niewolnictwo dotyczy wszystkiego co potrafią obecnie tak rozpowszechniające się na całym świecie elektroniczne urządzenia do przetwarzania, uruchamiania, przechowywania i last but not least do przekazywania informacji. [...] w całym owym tłumie komputerowym, we wszystkich generacjach hardware i software, modemach, serwerach nie ma śladu rozumu. Nie ma ani krzyły inteligencji. Pracują jak niewolnicy - na nasz rozkaz. Potrafią prowadzić do rajów "płciowej rozkoszy" i do "tarpejskiego piekła". Bredni (junk mail) nie odróżniają od informacji nawet tak niesłychanie doniosłej jak np. równanie $E=mc^2$ [...]"

Jak wykorzystamy owych niespotykanie posłusznych niewolników - zależy już tylko od naszej wiedzy, naszej inteligencji - komputer sam z siebie nic nie wymyśli i nie okaże się sam z siebie przydatny.

Cały podręcznik podzielony został na trzy główne części tematyczne. W każdej z nich zawarliśmy zarówno wiedzę teoretyczną, jak i zestaw praktycznych uwag pozwalających na wykorzystanie tej wiedzy w codziennym życiu.

TEORETYCZNE PODSTAWY DZIAŁANIA SYSTEMÓW

KOMPUTEROWYCH

W tym bloku przedstawiona zostanie zasada działania współczesnych systemów komputerowych wraz z zarysem historii rozwoju maszyn cyfrowych. Przedstawimy schemat funkcjonalny głównych podzespołów komputera, definicje pojęć, taksonomia architektur komputerów ze wskazaniem na najczęściej spotykane architektury komputerów klasy PC. Pokazane zostaną podstawowe typy pamięci stosowane w komputerach, zasada ich działania, komunikacja pomiędzy elementami systemu komputerowego oraz podstawowe rodzaje interfejsów.

Chcielibyśmy, abyście po zakończeniu tej części rozumieli z czego składa się komputer, jak działa, oraz byli w stanie optymalnie dobrać elementy systemu komputerowego.

KONFIGURACJA TYPOWEJ STACJI ROBOCZEJ

W tej części przedmiotu przedstawione zostaną główne (z punktu widzenia użytkownika) moduły systemu operacyjnego wraz z omówieniem metod zarządzania nimi oraz ich konfiguracji. Duży nacisk został położony na zagadnienia zabezpieczania zasobów lokalnych, kontroli poprawności działania stacji roboczej, analizę dzienników systemowych, oraz dobór i uaktualnianie sterowników dla poszczególnych podzespołów komputera. Omówione zostaną również aspekty bezpiecznej pracy stacji roboczej w sieci internet.

Student po zakończeniu tej części powinien być w stanie zainstalować i skonfigurować system operacyjny na komputerze wykorzystywanym jako stacja robocza.

ZAGADNIENIA SIECIOWE

Trzecia część przedmiotu jest poświęcona sieciom, zarówno tym lokalnym, jak i sieci Internet. Przy czym będzie to jedynie wprowadzenie. Przedstawimy Wam zasady obowiązujące przy transmisji danych cyfrowych, zaprezentujemy teoretyczny model sieci komputerowej i omówimy najpopularniejsze rozwiązania. Przedstawimy Wam również najpopularniejszy protokół - TCP/IP, oraz opowiemy trochę o bezpieczeństwie w systemach komputerowych.

Student po zakończeniu tej części powinien orientować się w terminologii stosowanej przy opisie usług sieciowych.

ZAMIAST ZAKOŃCZENIA WSTĘPU

Podręcznik, który właśnie otrzymaliście, jest wersją rozwojową ;) Będziemy go uzupełniali o nowe treści wraz z trwaniem tych zajęć. Dlatego też gorąco prosimy - korzystajcie z wersji internetowej. Będziemy ją aktualizowali na bieżąco, dodając nowy materiał. Oczywiście na zakończenie każdy z Was będzie mógł pobrać z sieci nową, pełną wersję podręcznika do nagrania na płytce CD. Będzie także istniała możliwość uzyskania już nagranej płyty - wystarczy napisać do nas - a wyślemy ją klasyczną pocztą.

Pozostaje nam życzyć miłej pracy ...

DEFINICJE I HISTORIA SYSTEMÓW KOMPUTEROWYCH

DEFINICJA SYSTEMU KOMPUTEROWEGO

Dawniej w potocznym obiegu komputery były często nazywane maszynami liczącymi, maszynami cyfrowymi (cytując za Stanisławem Lemem moglibyśmy powiedzieć "cyfraki" ;)) lub podobnie. Patrząc jednak na te nazwy z perspektywy historycznego rozwoju komputerów, należy zauważać, że wszystkie one kryją w sobie pewien błąd - główny nacisk kładą na liczenie (przetwarzanie cyfr). Oczywiście - liczenie jest podstawowym zadaniem komputera jako takiego, ale jeśli miałby to być jedyny wyróżnik systemu komputerowego, to czym różniłby się on od prostych kalkulatorów, czy nawet liczydeł? Potrzebujemy jakiejś innej definicji systemu, który moglibyśmy nazwać komputerem.

Na potrzeby tego przedmiotu przyjmiemy następującą definicję systemu komputerowego:

System komputerowy - układ współdziałających ze sobą dwóch składowych - sprzętu komputerowego oraz oprogramowania, najczęściej podłączony do sieci komputerowej, którego celem jest przetwarzanie danych w sposób możliwy do określenia poprzez program wczytywany z zewnątrz i przechowywany w pamięci.

Kluczem w tej definicji systemu komputerowego jest możliwość definiowania sposobu przetwarzania danych (czyli programowania) w swobodny sposób, a nie poprzez sztywny układ połączeń elementów elektronicznych czy mechanicznych - dlatego też współcześnie nikt już nie określa kalkulatora czy liczydeł mianem komputera.

Bardziej formalne metody określenia komputera były i są nadal podejmowane poprzez naukowców zajmujących się teorią informatyki, a dokładniej teorią obliczeń (ang. *computability theory*).

Podstawą do dalszych rozważań jest zazwyczaj Maszyna Turinga.

MASZYNA TURINGA

Maszyna Turinga stanowi najprostszy matematyczny model komputera, wszystkie algorytmy przetwarzania danych dają się do niej sprowadzić. Problem jest rozwiążalny na komputerze, jeśli da się zdefiniować rozwiązującą go maszynę Turinga.

Dokładniej: Maszyna Turinga to stworzony przez Alana Turinga abstrakcyjny model komputera służący do wykonywania algorytmów. Każdy algorytm wyrażalny na maszynie Turinga można wyrazić w rachunku lambda i odwrotnie. Ponieważ jednak maszyny Turinga rozszerza się bardzo trudno, zaś rachunek lambda bardzo łatwo, w praktyce są one o wiele mniej popularne jako rzeczywiste modele obliczeń. Są za to używane często do udowadniania nierozstrzygalności różnych problemów.

Maszyna Turinga składa się z nieskończonie długiej taśmy podzielonej na pola. Taśma może być nieskończona jednostronnie lub obustronnie. Każde pole może znajdować się w jednym z N stanów. Maszyna zawsze jest ustawiona nad jednym z pól i znajduje się w jednym z M stanów. Zależnie od kombinacji stanu maszyny i pola maszyna zapisuje nową wartość w polu, zmienia stan, a następnie może przesunąć się o jedno pole w prawo lub w lewo. Taka operacja nazywana jest rozkazem. Maszyna Turinga jest sterowana listą zawierającą dowolną ilość takich rozkazów. Liczby N i M mogą być dowolne, byle skończone. Czasem dopuszcza się też stan (M+1)-szy, który oznacza zakończenie pracy maszyny. Lista rozkazów dla maszyny Turinga może być traktowana jako jej program.

Maszyna posiadająca zdolność wykonywania dowolnego programu jest nazywana uniwersalną

maszyna Turinga. Praktyczną realizacją uniwersalnej Maszyny Turinga jest właśnie komputer.

Idąc dalej tym tropem, natkniemy się na twierdzenie Churcha-Turinga (opublikowane po raz pierwszy poprzez Stephena C. Kleene w 1943 roku). Mimo iż występuje tu słowo "twierdzenie", w praktyce jest to definicja.

Nie będziemy jej tutaj przytaczać w oryginalnej postaci, ograniczymy się do clou: *każde zadanie (każda funkcja) którą możemy określić jako obliczeniowe, może zostać wykonane przez każdy komputer zgodny z maszyną Turinga (zakładając brak ograniczeń sprzętowych i pamięciowych).*

Innymi słowy - dowolne obliczenia, które są możliwe do wykonania, mogą zostać wykonane poprzez uruchomienie programu na komputerze, któremu zapewnimy odpowiednią ilość czasu i pamięci. Opierając się na tym twierdzeniu, możemy przyjąć, że:

systemem komputerowym będziemy nazywali system, który jest w stanie wykonać te same zadania, co inne systemy komputerowe, o ile ograniczenia czasu i zasobów nie sąbrane pod uwagę.

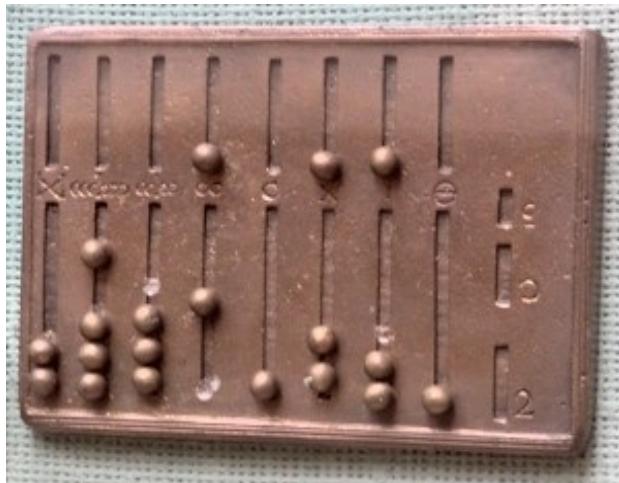
I taka definicja wreszcie jest jasna i klarowna - komputerem jest urządzenie które wykonuje wszystko to, co inne komputery ;)

PRZED KOMPUTERAMI ...

Patrząc na rozwój systemów komputerowych z historycznego punktu widzenia, nie sposób zaprzeczyć, że w pewnym sensie liczydła i kalkulatory są ich poprzednikami, mimo iż nie spełniają formalnej definicji systemu komputerowego.

Trudno jest określić moment powstania pierwszego komputera i jednoznacznie wskazać, które z urządzeń faktycznie było tym pierwszym komputerem. Sama definicja komputera zmieniała się z czasem i część urządzeń które kiedyś były uważane za komputery, według naszej definicji nimi nie są.

Samo słowo **komputer** pochodzi od angielskiego określenia *computer*, które pierwotnie oznaczało osobę, która wykonywała obliczenia ("ludzki" komputer), często przy pomocy mechanicznych urządzeń obliczeniowych. Przykładami takich wczesnych mechanicznych urządzeń służących do obliczeń są: abakus, linijka logarytmiczna czy mechanizm Antikythera (służący do obliczania pozycji astronomicznych, Grecja, 150 - 100 rok p.n.e)



Rekonstrukcja rzymskiego abakusa, zrobiona przez RGZ Museum w Mainz, 1977. Oryginał jest z brązu

Koniec średniowiecza przyniósł kolejny wzrost zainteresowania badaniami w dziedzinie matematyki i szybki rozwój techniki w Europie, czego bezpośrednim wynikiem było skonstruowanie pierwszego mechanicznego kalkulatora przez Wilhelma Schickarda w 1623 roku.

Jednakże żadne ze wspomnianych urządzeń nie spełniało postulatu "[programowalności](#)" - czyli nie było komputerem. Pierwsze programowalne urządzenie mechaniczne zostało opracowane w 1801 przez Josepha Marie Jacquarda. Była to maszyna tkacka, w której wzór materiału był definiowany za pomocą serii dziurkowanych kart. I mimo że wspomniana maszyna nie była w stanie policzyć czegokolwiek - to jednak najczęściej ona jest uznawana za protoplastę współczesnych komputerów - dzięki "[programowalności](#)" (choć mocno ograniczonej).



Maszyna tkacka Jacquarda. Aktualnie znajduje się w Muzeum Nauki i Przemysłu w Manchesterze.

I tak oto zbliżamy się do pierwszego ważnego nazwiska w rozwoju komputerów i informatyki. W 1837 [Charles Babbage](#) jako pierwszy opracował koncepcję oraz projekt w pełni programowalnego, mechanicznego komputera, który nazwał "maszyną analityczną". Niestety, głównie ze względu na brak możliwości finansowych, prototyp tej maszyny nigdy nie został zbudowany.

Równolegle w Niemczech [Gottfried Leibniz](#) opracował system binarny, który jest podstawą działania wszystkich współczesnych komputerów, a nawet szerzej - wszystkich urządzeń mających w nazwie człon "cyfrowa" (włączając w to telewizję cyfrową ;)).

Jako kolejny ważny etap rozwoju systemów komputerowych można uznać pierwsze zastosowanie automatycznego przetwarzania danych na wielką skalę podczas spisu ludności w Stanach Zjednoczonych Ameryki w 1890 roku. Wtedy to zaprojektowane przez Hermana Holleritha i wyprodukowane przez [IBM](#) (ówczesna nazwa tej firmy brzmiała Computing Tabulating Recording Corporation) maszyny przetwarzają dane wprowadzane za pomocą kart dziurkowanych.

Koniec XIX wieku to czas rozwoju wielu technologii, które posłużyły później do budowy pierwszych rzeczywistych komputerów: karty dziurkowane, logika dwuwartościowa, rozwój elektrotechniki i mechaniki precyzyjnej, teleks, lampa elektronowa, itp... Na początku XX wieku rosnące potrzeby

obliczeniowe naukowców najczęściej były realizowane przez analogowe (elektryczne bądź mechaniczne) układy, które były zaprojektowane do jednego celu - jak najdokładniej naśladować modelowane zjawisko. Niemniej, mimo swojego stopnia skomplikowania oraz często wyrafinowanej konstrukcji, nie mogły być programowane, co ograniczało ich szersze wykorzystanie.

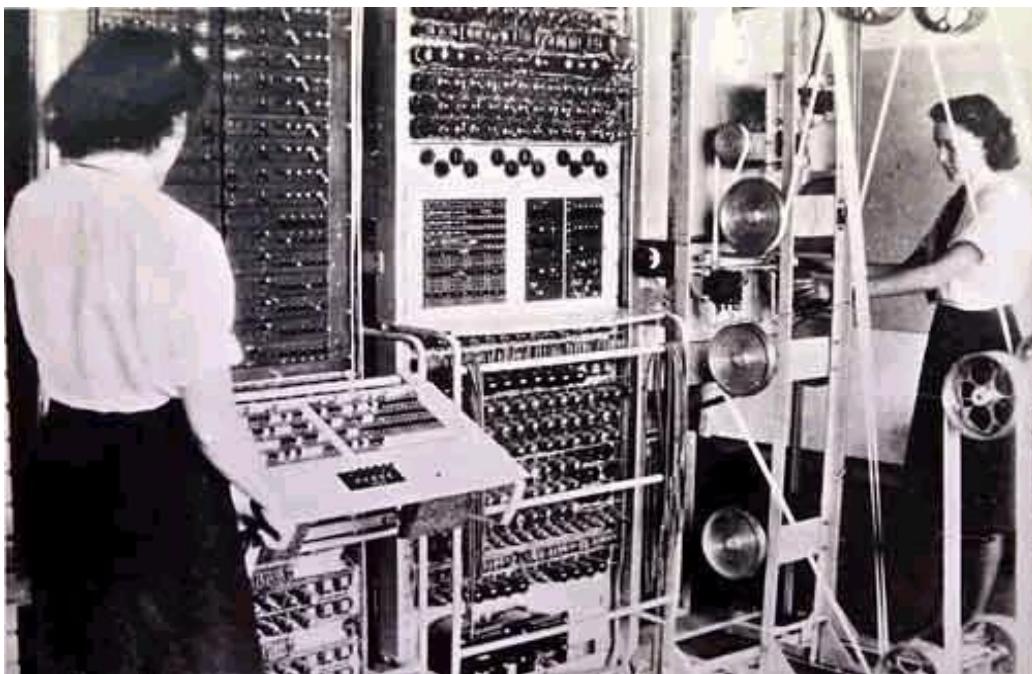
Charakterystyka pierwszych pięciu maszyn uważanych za komputery

Zuse Z3	Maj 1941	Tak	Nie	Tak (karty dziurkowane)	Tak (sprawdzona na podst. rekonstrukcji)
Atanasoff-Berry Computer	Lato 1941	Tak	Tak	Nie	Nie
Colossus	Styczeń 1944	Tak	Tak	Częściowo, poprzez zmianę okablowania	Nie
Harvard Mark I/IBM ASCC	1944	Nie	Nie	Taśma dziurkowana	Nie
ENIAC	1944	Nie	Tak	Częściowo, poprzez zmianę okablowania	Tak
ENIAC	1948	Nie	Tak	Tablica funkcji w ROM	Tak

Pierwsze działające komputery zgodne z [maszyną Turinga](#) zostały opracowane w latach 30 i 40 ubiegłego wieku (pięć pierwszych maszyn wymieniamy w tabeli wyżej). Wykorzystanie elektroniki oraz coraz bardziej swobodne programowanie były ważnymi osiągnięciami, jednak wskazanie "pierwszego działającego elektronicznego komputera" jest trudne:

- elektromechaniczne maszyny serii "Z" opracowywane w Niemczech przez Konrada Zuse. Z3 (1941) był pierwszym działającym urządzeniem, w którym wykorzystano arytmetykę binarną, włączając w to możliwości wykonywania obliczeń na liczbach zmiennoprzecinkowych. Z3 można także było programować, tak więc aktualnie jego uważa się za pierwszy działający komputer na świecie. Nie jest to jednak pierwszy elektroniczny komputer - został zbudowany w całości w oparciu o przekaźniki elektromechaniczne;
- przy budowie Atanasoff-Berry Computer (1941) po raz pierwszy wykorzystano lampy elektronowe, niestety nie został on nigdy w pełni uruchomiony;
- tajny British Colossus Mark II (1944) posiadał ograniczone możliwości programowania, lecz pokazał iż jest możliwe zbudowanie urządzenia opartego na tysiącach lamp elektronowych. Ten komputer był wykorzystany do łamania niemieckich szyfrów wojennych.
- Harvard Mark I (1944) był wielkim projektem (zbudowano go z 765 000 elementów), co pozwoliło na pamiętanie 72 liczb i wykonanie dzielenia w czasie ... ok 15 sekund. Jako ciekawostkę można potraktować fakt, że Mark I działał w oparciu o system dziesiętny;

- zbudowany przez Laboratorium Badań Balistycznych armii USA ENIAC (1946) również wykorzystywał system dziesiętny. Był to pierwszy elektroniczny komputer ogólnego przeznaczenia, mimo że początkowo jedyną możliwością programowania była zmiana okablowania.



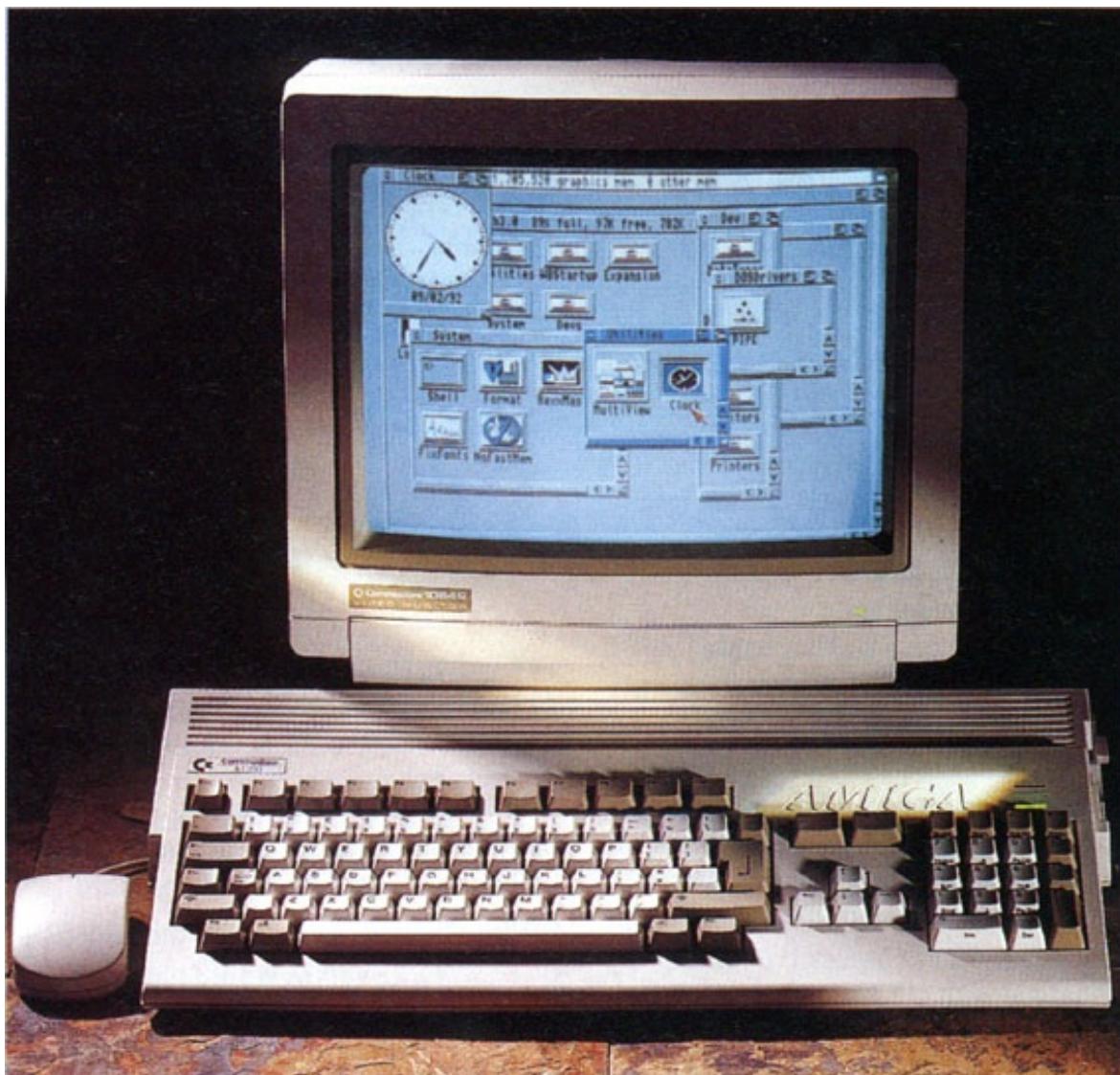
Komputer Colossus Mark II. Fakt jego istnienia i wykorzystywania ujawniono dopiero pod koniec lat 70, przez 25 lat nie był ujawniany i nie znajdował się na listach pierwszych komputerów.

ROZWÓJ SYSTEMÓW KOMPUTEROWYCH

KOMPUTERY ELEKTRONICZNE

Komputery pracujące w oparciu o lampy elektronowe były wykorzystywane do końca lat 50 dwudziestego wieku. W latach 60 zostały wyparte przez mniejsze, szybsze, tańsze i potrzebujące mniej energii komputery oparte na tranzystorach. To właśnie tranzystor spowodował wcześniej nieprzewidywalny boom na komercyjne wykorzystanie komputerów. W latach 70 pojawiła się następna technologia - układy scalone - i produkowane przy jej wykorzystaniu mikroprocesory takie jak np. Intel 4004. W latach 80 komputery stały się na tyle małe, tanie i niezawodne by zastąpić proste, mechaniczne układy sterowania w sprzęcie powszechnego użytku, takim jak pralki.

Jednocześnie spadek cen umożliwił pojawienie się na rynku pierwszych komputerów dostępnych dla ogólnego społeczeństwa, pod nazwą komputerów domowych (Apple II, ZX Spectrum, Commodore 64 czy Atari). Początkowo były to konstrukcje oparte o ósmio-bitowe procesory, gdzie pamięcią masową był zazwyczaj magnetofon (przykładowo ZX Spectrum czy Commodore 64). Potem pojawiły się bardziej zaawansowane konstrukcje 16-to bitowe, takie jak Atari ST czy Amiga. Cechą wspólną tych maszyn było zabudowanie całej elektroniki (procesora, pamięci) w jednej obudowie z klawiaturą, oraz możliwość wykorzystania telewizora jako ekranu. Mimo swoich możliwości, były one wykorzystywane głównie do rozrywki.



Amiga 1200 - jeden z najdoskonalszych (i jednocześnie ostatnich) komputerów domowych.

Jednocześnie firmy zaczęły na szeroką skalę wykorzystywać komputery klasy PC (komputery osobiste). Ich pierwszym, szeroko znany przedstawicielem był IBM PC. Komputery te od samego początku były wyposażone w napęd dyskietek oraz wymagały specjalizowanego monitora. Były wykorzystywane głównie do prac biurowo-inżynierskich. W późnych latach 80 technicznie pod każdym względem przegrywały z ówczesnymi komputerami domowymi. Atari ST czy Amiga miały pełny, wielozadaniowy system operacyjny z GUI, podczas gdy w świecie PC dominował DOS i tryb znakowy. Podobnie z procesorami - nowoczesne 16-bitowe jednostki Motoroli wykorzystywane przez komputery domowe były lepsze niż ówczesne produkty Intela. Ale docelowo wygrała otwartość standardu - genialny chwyt marketingowy. Firma IBM opublikowała pełną dokumentację techniczną swojego modelu PC, zastrzegając jedynie jako patent zawartość BIOS. Pomysł natychmiast pochwyciły tysiące firm na całym świecie (głównie na dalekim wschodzie) i rozpoczęła się produkcja tzw. klonów (kopii) oryginalnych PC. W finale doprowadziło to do dzisiejszej dominacji następców tych komputerów. Jedyną firmą, która do dziś oparła się wszechobecnym PC, jest Apple. Wprowadzony przez nią Macintosh zyskał (dzięki technicznej perfekcji) grono wiernych zwolenników, głównie w Stanach Zjednoczonych. Firma Apple także w pełni zagospodarowała rynek profesjonalnych systemów małej poligrafii oraz grafiki w latach 80 i 90 ubiegłego wieku, co w sumie pozwoliło jej na utrzymanie się na rynku do dzisiejszego dnia.



Macintosh Classic - często uważany za pierwszy z komputerów nowej generacji.

Współcześnie zanikły komputery domowe, natomiast komputery osobiste w połączeniu ze wszelobecnym Internetem są niezbędnym elementem każdego gospodarstwa. Lecz ich wykorzystanie nie ogranicza się jedynie do urządzeń jawnie nazwanych komputerami. Każdy nowoczesny telefon jest komputerem zgodnym z Maszyną Turinga, samochód, telewizor, pralka czy inne urządzenia elektroniczne mają zabudowany w środku komputer, którego zazwyczaj nawet nie widzimy. Do niedawna tego typu urządzenia rozwijały się najszybciej. Dziś jesteśmy na progu kolejnej zmiany epoki - komputery osobiste i notebooki zaczynają tracić dotychczas zajmowane miejsce na rzecz tabletów, smartphone-ów, czy "ubieralnej elektroniki".



Smartphone firmowany przez Google - przyszłość komputerów osobistych?

Ponadto, trwają już prace nad komputerami optycznymi, chemicznymi czy kwantowymi - tak więc wydaje się, że stoimy na progu kolejnej rewolucji ... a jeszcze nie wszyscy zdążyli oswoić się z tym, co już mamy :)

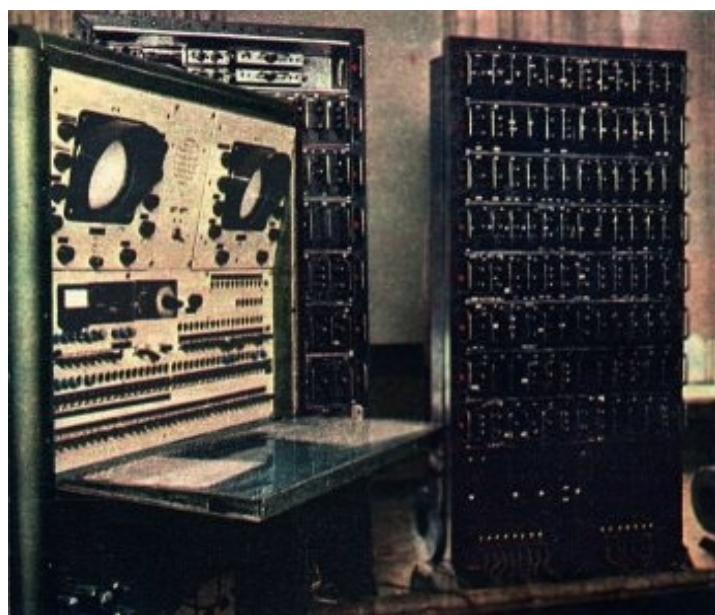
Ten przedmiot jest poświęcony głównie komputerom osobistym, lecz warto pamiętać, że w obecnych czasach wykorzystanie komputerów jest praktycznie nieograniczone i dotyczy każdego elementu naszego życia.

Wkład polaków w powstawanie i budowę pierwszych komputerów był dwojaki. Polski matematyk [Jan Łukasiewicz](#) opracował tzw. beznawiasową notację polską, na podstawie której powstała odwrotna notacja polska (ang. *Reverse Polish Notation*, RPN). Jest to sposób zapisu wyrażeń arytmetycznych, w którym znak wykonywanej operacji umieszczony jest po operandach (zapis postfiksowy), a nie pomiędzy nimi jak w konwencjonalnym zapisie algebraicznym (zapis infiksowy).

Wydawałoby się, że to nie jest żadne wielkie osiągnięcie. Ale ... zapis ten pozwala na całkowitą rezygnację z użycia nawiasów w wyrażeniach, jednoznacznie określając kolejność wykonywanych działań. Dzięki temu można było zapisywać naprawdę skomplikowane wzory z jednej strony - w sposób zrozumiały nawet dla bardzo prostych komputerów, a z drugiej - możliwy do pojęcia nawet dla mało zdolnego technika. RPN bardzo ułatwia wykonywanie na komputerze obliczeń z nawiasami i zachowaniem kolejności działań. Zarówno algorytm konwersji notacji konwencjonalnej (infiksowej) na odwrotną notację polską (postfiksową), jak i algorytm obliczania wartości wyrażenia danego w RPN są bardzo proste i wykorzystują stos. Jednocześnie działają one szybko i pojęciowo są na tyle proste, by stała się możliwa ich implementacja nawet przez osoby uczące się programowania. Z tego powodu przez długi czas RPN była podstawą przy programowaniu wszelkiego rodzaju komputerów, była także wykorzystywana w kalkulatorach programowalnych.

Z drugiej strony, w latach 60 i 70 polscy naukowcy dotrzymywali kroku światowym potentatom w tej dziedzinie, choćby koncernowi IBM, osiągając nawet chwilowe przewagi. Pierwszą polską maszyną matematyczną, która znalazła zastosowanie praktyczne, był zbudowany w 1958 roku w warszawskim Zakładzie Aparatów Matematycznych PAN komputer [XYZ](#), pracujący na lampach i przekaźnikach. Nie przypominał on współczesnych komputerów, ale był czymś więcej niż zwykłym kalkulatorem.

Wykorzystywał algorytmy umożliwiające dokonywanie obliczeń numerycznych z szybkością 800 operacji na sekundę, był swobodnie programowalny w makroasemblerze SAS i polskim języku algorytmicznym SAKO - tak więc konstrukcyjnie nie odbiegał od najnowocześniejszych komputerów zachodnich. Na przełomie lat 50 i 60 XYZ pracował głównie dla potrzeb wojska, wyznaczając tzw. przeliczniki artyleryjskie. Udoskonaloną wersję XYZ były budowane od 1960 roku maszyny typu ZAM-2.



XYZ (1958) to pierwszy elektroniczny komputer cyfrowy zbudowany i uruchomiony w Polsce. Wyprzedził o kilka miesięcy EMAL-2, a wcześniejszy EMAL nie został w pełni uruchomiony. Poprzedziły go: lampowy komputer analogowy ARR oraz nieelektroniczne komputery zerowej generacji: dydaktyczny GAM-1 i użytkowy PARK. Zbudowany i uruchomiony został w Warszawie przy ul. Śniadeckich 8, w lokalu Biura Obliczeń i Programów Zakładu Aparatów Matematycznych PAN (późniejszego Instytutu Maszyn Matematycznych). Zespołem twórców kierował prof. dr inż. Leon Łukasiewicz (wówczas docent). XYZ był modelem laboratoryjnym użytkowej maszyny. Zdjęcie pochodzi

W roku 1962 we wrocławskich zakładach Elwro podjęto seryjną produkcję lampowej maszyny UMC-1 do obliczeń inżynierskich. Od następnego roku zaczęto budować tam tranzystorowe komputery ODRA, oznaczane numerami 1001, 1003, 1204, 1304. Ostatni model ODRA 1305 (już na układach scalonych) powstał przy współpracy następcy ZAM PAN – Instytutu Maszyn Matematycznych.

W 1973 roku powstały pierwsze mikrokomputery: MOMIK 8b i tzw. komputer Karpińskiego K-202 wraz z urządzeniami peryferyjnymi takimi jak monitory, drukarki i pamięci (początkowo taśmowe i bębnowe).

Komputer MOMIK rozbudowano i nazwano MERA 300. Był on szeroko stosowany w księgowości oraz przemyśle, gdzie używano go do sterowania procesami technologicznymi.

Dalej już tylko kopiowano rozwiązania zachodnie. Jako pierwsza powstała Mazovia - klon IBM PC XT (oparty na produkowanym w ZSRR procesorze zgodnym z Intellem 8086). Potem był Elwro 800 Junior - zgodny z ZX Spectrum. Ponieważ były to rozwiązania w dużej mierze odtwórcze i w dodatku nigdy nie produkowane na wielką skalę, po zmianie ustroju zniknęły z rynku.

W powyższym segmencie przedstawiliśmy Wam jedynie zarys historii powstawania komputerów. Zainteresowani znajdą więcej informacji [na tej stronie](#).

GŁÓWNE TYPY ARCHITEKTURY KOMPUTERÓW

W pierwszym odruchu termin "architektura", rozumiany tradycyjnie jako "sztuka projektowania, wznoszenia budowli", może dziwić i być mylący. Można bowiem zadać sobie pytanie: co ma z tym wspólnego system komputerowy? Przecież nie chodzi tu chyba o "wystrój", kształty i kolorystykę komputera. Śłusznie - nie w tym znaczeniu używamy terminu "architektura". Za twórcę terminu "architektura systemu komputerowego" uważa się Wernera Buchholza, redaktora pracy "Planning a Computer System" opublikowanej w 1962 roku. Architekturę systemu komputerowego definiują następujące cechy:

- budowa słowa rozkazowego;
- format danych;
- sposób adresowania pamięci;
- sposób współpracy z urządzeniami zewnętrznymi;
- sposób reagowania na przerwania;
- tryby użytkownika komputera.

Architektura systemu komputerowego to po prostu organizacja jego elementów. System komputerowy można opisywać na różnych poziomach abstrakcji: funkcjonalnym, struktury i technologii. Architektura to sposób opisu systemu komputerowego na poziomie strukturalno-funkcjonalnym, lecz nie technologicznym.

Na początek zajmiemy się jednak architekturą na najwyższym poziomie abstrakcji. Nie będziemy jeszcze rozważać adresowania pamięci czy budowy słowa rozkazowego, etc ... na to wszystko przyjdzie pora. Najpierw zastanówmy się, z czego powinien składać się komputer.

ARCHITEKTURA HARVARDZKA

W świecie informatyki wyróżnia się dwa podstawowe typy architektury systemów komputerowych. Historycznie starsza - lecz mniej popularna - jest architektura Harvardzka.

W przypadku tak zbudowanych komputerów pamięć danych programu jest oddzielona od pamięci rozkazów, co oznacza z jednej strony prostszą budowę i większą szybkość działania, z drugiej - na znaczące ograniczenie elastyczności.

W dzisiejszych rozwiązaniach ten typ architektury jest często wykorzystywany w procesorach sygnałowych oraz przy dostępie procesora do pamięci cache.

Separacja pamięci danych od pamięci rozkazów sprawia, że architektura harwardzka jest również powszechnie stosowana w mikrokomputerach jednoukładowych, w których dane programu są najczęściej zapisane w nieulotnej pamięci ROM (EPROM/EEPROM), natomiast dla danych tymczasowych wykorzystana jest pamięć RAM (wewnętrzna lub zewnętrzna).

Prawie wszystkie nowoczesne komputery jako podstawę wykorzystują nowszą architekturę - von Neumann'a, przez co często jako komputer rozumie się jedynie urządzenie zgodne z tą architekturą. W takim wypadku wielu wcześniejszych urządzeń (włączając w to 5 wymienionych przez nas w poprzednim segmencie) nie powinno się nazywać "komputerami", lecz zwykle pozostaje się przy tej nazwie ze względu na historyczny kontekst.

ARCHITEKTURA VON NEUMANN'A

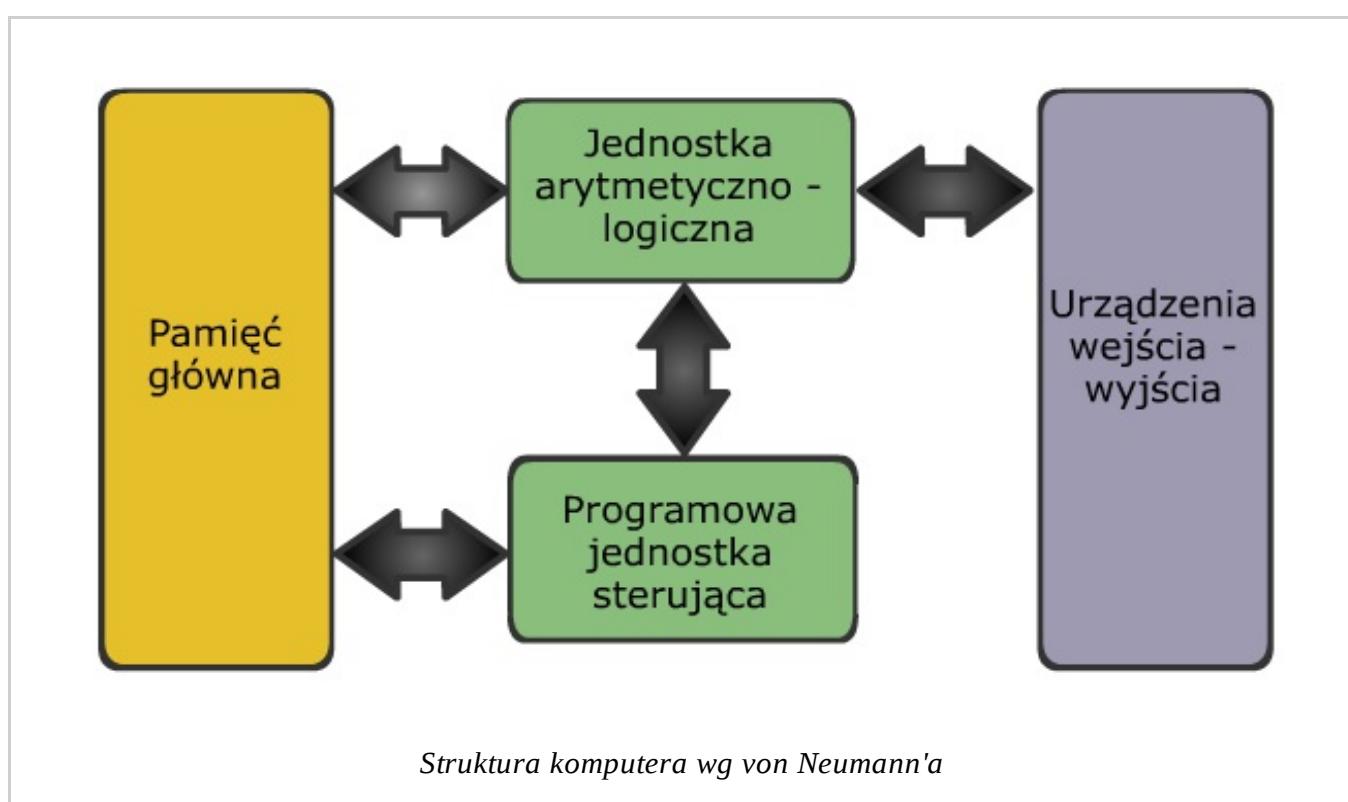
Dzięki twórcom ENIAC-a oraz analizie jego właściwości i ograniczeń, zaproponowana została obowiązująca do dziś w przypadku większości komputerów architektura von Neumann'a. Po raz pierwszy została ona ogłoszona przez [John'a von Neumann'a](#) w opublikowanym w 1945 roku artykule "First Draft of a Report on the EDVAC", a jej głównym założeniem było zastosowanie jednego typu pamięci do przechowywania danych i programu. Idea ta miała położyć podwaliny pod nowy komputer von Neumann'a - EDVAC (ang. *Electronic Discrete Variable Computer*).

W roku 1946 von Neumann w Princeton Institute for Advanced Studies rozpoczął projektowanie komputera, który jest uważany za protoplastę współczesnych maszyn.

W tym czasie rozpoczęto kilka projektów opracowania komputerów, które przechowywałyby wykonywany program w pamięci operacyjnej. Pierwszy z nich ukończony został w Wielkiej Brytanii, szerzej znany jako Manchester Small-Scale Experimental Machine (SSEM) lub "Baby". Jednakże to nie "Baby" lecz ukończony rok później EDSAC jest pierwszą praktyczną implementacją architektury von Neumann'a. Niewiele później zaprezentowany został komputer opisywany przez samego von Neumann'a (EDVAC), lecz nie został wdrożony do praktycznych zastosowań przez kolejne 2 lata.

System komputerowy von Neumann'a nie posiada oddzielnych pamięci do przechowywania danych i instrukcji. Instrukcje jak i dane są zakodowane w postaci liczb. Bez analizy programu trudno jest określić czy dany obszar pamięci zawiera dane czy instrukcje. Wykonywany program może się sam modyfikować traktując obszar instrukcji jako dane, a po przetworzeniu tych instrukcji - danych - zacząć je wykonywać.

Model komputera wykorzystującego architekturę von Neumanna jest często nazywany przykładową maszyną cyfrową (PMC) lub IAS.



Powyższy rysunek przedstawia ogólną strukturę komputera, na którą składają się:

- **pamięć główna** - w tej pamięci przechowywane są dane oraz rozkazy;
- **jednostka arytmetyczno-logiczna (ALU)** - wykonuje działania na danych binarnych;
- **jednostka sterująca** - interpretuje i wykonuje rozkazy z pamięci;
- **urządzenia wejścia/wyjścia** - ich pracą kieruje jednostka sterująca.

System komputerowy zbudowany w oparciu o architekturę von Neumann'a charakteryzuje:

- obecność skończonej i funkcjonalnie pełnej listy rozkazów;
- możliwość wprowadzenia programu do systemu komputerowego poprzez urządzenia zewnętrzne i jego przechowywanie w pamięci w sposób identyczny jak danych;
- jednakowy dostęp dla procesora do danych i instrukcji;
- przetwarzanie informacji dzięki sekwencyjnemu odczytywaniu instrukcji z pamięci komputera i wykonywaniu tych instrukcji w procesorze.

John von Neumann tak opisał niegdyś powyższą strukturę komputera:

1. Po pierwsze, ponieważ urządzenie to jest przede wszystkim komputerem, najczęściej będzie wykonywało elementarne operacje matematyczne - dodawanie, odejmowanie, mnożenie, dzielenie. Jest więc rozsądne, że powinno posiadać wyspecjalizowane "organy" do wykonywania tych operacji.
Należy jednak zauważać, że chociaż powyższa zasada jako taka brzmi rozsądnie, to szczegółowy sposób jej realizacji wymaga głębokiego zastanowienia [...] W każdym przypadku *centralna, arytmetyczna* część urządzenia będzie prawdopodobnie musiała istnieć, co oznacza występowanie *pierwszej specyficznej części komputera: CA*.
2. Po drugie, logiczne sterowanie urządzeniem, to znaczy odpowiednie szeregowanie jego operacji może być najefektywniej realizowane przez centralny organ sterujący. Jeżeli urządzenie ma być *elastyczne*, to znaczy możliwie *uniwersalne*, należy rozróżnić specyficzne rozkazy związane z określonym problemem i ogólne "organy" sterujące, dbające o wykonanie tych rozkazów - czymkolwiek by one nie były. Te pierwsze muszą być w jakiś sposób przechowywane; te drugie - reprezentowane przez określone działające części urządzenia. Przez *sterowanie centralne* rozumiemy tylko tą ostatnią funkcję, a "organy", które ją realizują, tworzą *drugią specyficzną część urządzenia: CC*.
3. Po trzecie, jakiekolwiek urządzenie, które ma wykonywać długie i skomplikowane sekwencje działań (w szczególności obliczeń), musi mieć odpowiednio dużą pamięć [...] (b) Rozkazów kierujących rozwiązywaniem skomplikowanego problemu może być bardzo dużo, zwłaszcza wtedy, gdy kod jest przypadkowy (a tak jest w większości przypadków). Muszą one być pamiętane [...]
4. Trzy specyficzne części CA, CC (razem C) oraz M odpowiadają neuronom skojarzeniowym w systemie nerwowym człowieka. Pozostają do przedyskutowania równoważniki neuronów *sensorycznych* (doprowadzających) i *motorycznych* (odprowadzających). Są to "organy" *wejścia i wyjścia* naszego urządzenia.
Urządzenie musi mieć możliwość utrzymania kontaktu z wejściem i wyjściem za pomocą specjalistycznego narzędzia. Narzędzie to będzie nazwane *zewnętrznym narzędziem rejestrującym urządzenia: R* [...]
5. Po czwarte urządzenie musi być wyposażone w organy przenoszące [...] informację z R do swoich specyficznych części C i M. "Organy" te stanowią jego wejście, a więc *czwartą, specyficzną część: I*. Zobaczmy, że najlepiej jest dokonywać wszystkich przeniesień z R (poprzez I) do M, a nigdy bezpośrednio do C [...]
6. Po piąte, urządzenie musi mieć "organy" służące do przenoszenia [...] ze swoich specyficznych części C i M do R. Organy te tworzą jego *wyjście*, a więc *piątą specyficzną część: O*. Zobaczmy, że znowu najlepiej jest dokonywać wszystkich transferów z M (poprzez O) do R, nigdy zaś bezpośrednio z C [...]

Podane warunki pozwalają przełączać system komputerowy z wykonania jednego zadania (programu) na inne bez fizycznej ingerencji w strukturę systemu, a tym samym gwarantują jego uniwersalność.

KLASYFIKACJA SYSTEMÓW KOMPUTEROWYCH

Systemy komputerowe można porównywać stosując różne kryteria. W poprzednim segmencie

pokazaliśmy dwa podstawowe typy architektury komputerów. Lecz podział taki zazwyczaj nie oddaje wszystkich możliwości systemu komputerowego. Jeśli jako kryterium przyjmuje się przeznaczenie systemu, wówczas wyróżnia się komputery:

- ogólnego przeznaczenia (konfiguracja tak dobrana, aby komputer mógł służyć wielu celom). Tutaj znajdują się wszystkie współczesne komputery osobiste, lecz także - większość systemów serwerowych, stacji roboczych, tabletów itp.
- specjalizowane (przeznaczone do wykonywania określonego zdania lub klasy zdań). Tutaj ekstremalnym przykładem może być system osadzony do sterowania wahadłowcem kosmicznym. Ale także większość mikrokontrolerów umieszczonych w samolotach, samochodach czy pralkach. Do tego grona możemy też zaliczyć konsole do gier.

Kryterium klasyfikacji może być tryb przetwarzania informacji. Tryb ten wynika z powiązań funkcjonalnych, istniejących w systemie komputerowym. Pod tym względem systemy komputerowe można sklasyfikować następująco:

- systemy jednoprogramowe - jednozadaniowe (jednocześnie jest wykonywany tylko jeden program, wykonanie kolejnego programu jest możliwe po całkowitym zakończeniu programu poprzedniego). Taki tryb pracy współcześnie jest wykorzystywany jedynie w urządzeniach osadzonych o niewielkich wymaganiach
- systemy wieloprogramowe - wielozadaniowe (wiele programów może być wykonywanych pozornie jednocześnie, wykonywanie wielu programów następuje na przemian. Więcej o tym powiemy później). Do tej grupy należą nowoczesne komputery osobiste;
- systemy wielodostępne - wieloużytkownikowe (możliwe jest korzystanie z systemu jednocześnie przez wielu użytkowników), czyli wszelkiego rodzaju komputery klastrowe, ale także systemy oferujące zdalny dostęp. Do tego grona możemy zaliczyć serwery usług terminalowych, zarówno Windows (niektóre wersje, zazwyczaj z "server" w nazwie) jak i Unix (Linux)

Kolejnym kryterium podziału systemów komputerowych, wiążącym się z gabarytami komputera, może być sposób jego użytkowania. Według tego kryterium wyróżnić można:

- systemy osadzone (ang. *embedeed*) - komputery sterujące wbudowywane w inne urządzenia;
- komputery osobiste (biurowe, domowe, używane w jednym miejscu - stacjonarne);
- komputery przenośne (ang. *laptop, notebook*);
- komputery kieszonkowe (ang. *plamtop*) - choć dziś używa się raczej podziału angielskiego na tablety i smartphone-y;
- systemy serwerowe - wykorzystywane główne poprzez sieć.

Następny podział wynika z długości słowa procesora komputera. W tym przypadku możemy wydzielić systemy:

- 4-bitowe (raczej jako zaszłość historyczna);
- 8-bitowe;
- 16-bitowe;
- 32-bitowe - odchodzący standard komputerów osobistych, lecz ciągle często spotykany w urządzeniach kieszonkowych lub energooszczędnnych;
- 64-bitowe - aktualnie standard w większości systemów ogólnego przeznaczenia.

Podobnych podziałów można zamieścić jeszcze wiele. Mamy nadzieję, że to co zamieściliśmy do tej pory da wam wyobrażenie jak szerokie jest aktualnie pojęcie systemów komputerowych. W trakcie tego przedmiotu zajmiemy się jednak tylko i wyłącznie systemami klasycznymi - takimi jak komputery osobiste i systemy serwerowe.

ARCHITEKTURA SYSTEMU KOMPUTEROWEGO

Chcielibyśmy przedstawić Wam trochę bliżej poszczególne elementy systemu komputerowego. Ta lekcja w całości będzie poświęcona sprzętowi - przedstawimy jak zbudowany jest system komputerowy, jaka jest jego logiczna struktura oraz najważniejsze elementy. Przy czym skupimy się bardziej na ich funkcjach i właściwościach, niż na budowie wewnętrznej.

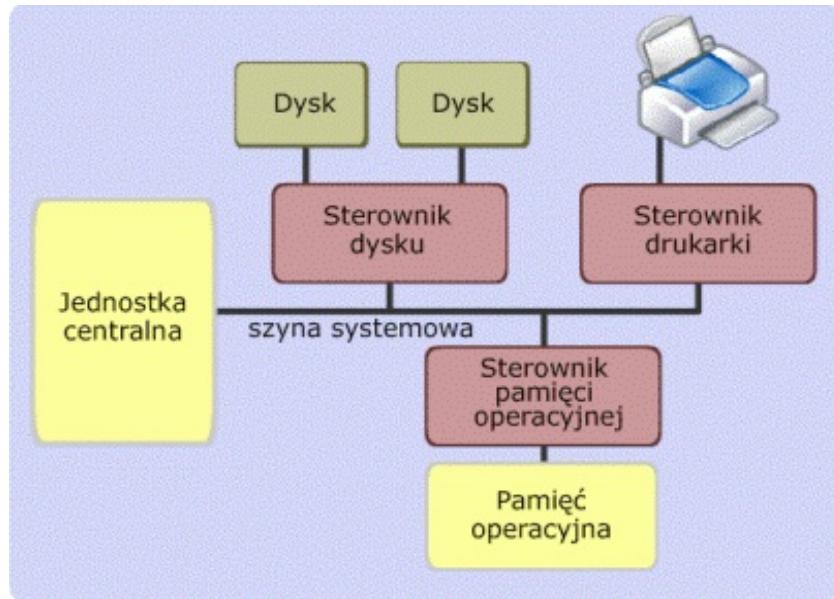
SKŁADNIKI SYSTEMU KOMPUTEROWEGO

Wstęp historyczno-teoretyczny mamy już za sobą - teraz pora zacząć mówić o konkretach ;). Zaczniemy więc od opisu strukturalnego systemu komputerowego: co wchodzi w jego skład, do czego służą poszczególne elementy i jak w dużym uproszczeniu działają. Ciągły wzrost zastosowań komputerów stymulował ich rozwój, zarówno w aspekcie funkcjonalnym, jak i technicznym. Komputer z wiekiem obrastał w piórka, aby zwiększyć swoją funkcjonalność, aby uprościć obsługę, aby połączyć się ze światem zewnętrznym, itd. ... można tu jeszcze zapewne wiele powodów podać. Dzięki temu doczekaliśmy się systemu komputerowego. Doczekaliśmy się czasów, w których komputer przekształcił się z urządzenia elektronicznego o dość jednolitej i regularnej strukturze w zespół wzajemnie powiązanych ze sobą urządzeń o rozproszonej odpowiedzialności, różnym stopniu skomplikowania i różnej szybkości działania. Podzespoły wchodzące w skład komputera są wyposażane przez producenta w oprogramowanie zapewniające jego efektywne wykorzystanie. Ten zbiór urządzeń wraz z oprogramowaniem zaczęto nazywać systemem komputerowym. Zmiana terminologiczna była również powiązana z pojawiением się Teorii Systemów i zastosowaniem jej do opisywania współdziałania elementów komputera. Nie będziemy Was męczyć tymi opisami - zainteresowani mogą się w nie zagłębić samodzielnie. Lecz uwaga - lojalnie Was ostrzegamy - to dział matematyki ... ;).

Wróćmy jednak do tematu. Jak wspominaliśmy na samym początku tego kursu - systemem komputerowym określamy zarówno sprzęt, jak i oprogramowanie. W tej części podręcznika skupimy się na sprzęcie. Przy czym główny nacisk położymy na komputery osobiste i konstrukcje do nich zblizone, natomiast w skrócie omówimy rozwiązania specyficzne, takie jak systemy wbudowane czy klastry obliczeniowe.

Każdy system komputerowy powinien składać się z co najmniej kilku elementów. Pamiętając o koncepcji von Neumann'a, ktoś mógłby zaryzykować stwierdzenie, iż musi być ich co najmniej 4: pamięć, jednostka arytmetyczno-logiczna, jednostka sterująca oraz układ wejścia/wyjścia. W praktyce ten podział nie zawsze jest konieczny w sensie fizycznym. Istnieją bowiem tak zwane komputery jednoukładowe (mikrokontrolery - takie informatyczne *wash-and-go*, czyli wszystko w jednym). Jednak nawet w tym przypadku wyraźnie można wydzielić odpowiednie części funkcjonalne w układzie scalonym. Dlatego zaryzykujemy stwierdzenie, iż każdy współczesny system komputerowy składa się z co najmniej jednej **jednostki centralnej**, która łączy następujące elementy występujące w koncepcji von Neumann'a: jednostkę arytmetyczno-logiczną (która liczy) oraz jednostkę sterującą (która interpretuje program jak liczyć) - taką jednostkę centralną będziemy nazywali **procesorem**. Oprócz procesora system komputerowy musi mieć co najmniej jednego rodzaju **pamięć** (przechowującą dane wejściowe do obliczeń wraz z ich wynikami) oraz **układy wejścia-wyjścia** (pozwalające na komunikację ze światem zewnętrznym - wprowadzenie programu, danych do obliczeń czy też prezentację wyników).

Jeśli ograniczymy się do klasycznych komputerów ogólnego stosowania, możemy ten podział jeszcze bardziej uszczegółowić: system komputerowy ogólnego stosowania składa się z jednej bądź kilku **jednostek centralnych** oraz kilku **sprzętowych sterowników** (ang. *device controller*) połączonych wspólną **magistralą danych** umożliwiającą współpracę ze wspólną **pamięcią** (patrz rysunek poniżej).



Architektura systemu komputerowego

Każde urządzenie wchodzące w skład systemu komputerowego ma zdefiniowany standard połączenia się z jednostką centralną oraz posiada swój sterownik (bez względu na to, czy jest to dysk twardy, karta grafiki, czy drukarka). Sterownik ma za zadanie zapewnić uporządkowany, kontrolowany dostęp do wspólnej pamięci, lub innych zasobów, pomimo współbieżnego działania z jednostką centralną. Z drugiej strony - sterownik zapewnia zunifikowany dostęp do urządzeń pełniących tą samą rolę, lecz wyprodukowanych przez różnych producentów.

Ostatecznie możemy powiedzieć, że na architekturę komputera składają się:

1. **Jednostka centralna (CPU)**, czyli przynajmniej jeden procesor. Przynajmniej - gdyż coraz częściej wykorzystywane są dwa (lub więcej) procesory (nawet jeśli są w postaci jednego układu scalonego).
2. **Magistrala systemowa** poprzez którą komunikują się i przesyłają dane komponenty systemu komputerowego.
3. **Wspólna pamięć**, która służy do przechowywania danych i kodu instrukcji procesora, ale - zgodnie z modelem von Neumann'a - ma jednolitą strukturę.
4. **Urządzenia wejścia-wyjścia** (np. klawiatura - monitor), dzięki którym możliwa jest komunikacja ludzi z komputerem.

W następnych segmentach tej lekcji postaramy się wyjaśnić dokładniej budowę oraz działanie wyżej wymienionych elementów systemu komputerowego.

PROCESOR

Co to jest?

Czasem jest utożsamiany z całym systemem komputerowym, czasem sądzi się, że to on i tylko on decyduje o wydajności systemu, a już przekonanie, iż w procesorze kryje się inteligencja komputera - jest powszechnie. Niestety, żadne z powyższych stwierdzeń nie jest prawdą. Procesor nie stanowi systemu komputerowego w całości, bez pozostałych komponentów nie jest do niczego przydatny, co więcej - koszt procesora wcale nie musi być najistotniejszą pozycją w koszcie systemu komputerowego. Wpływ procesora na wydajność jest olbrzymi - to fakt. Lecz nawet najszybszy procesor, który współpracuje ze zbyt małą ilością pamięci, bądź też z pamięcią zbyt wolną, nigdy nie będzie w pełni wykorzystany. Podobnie w przypadku innych, nieprawidłowo dobranych komponentów systemu komputerowego. A co do inteligencji ... no cóż, warto wyraźnie podkreślić, że współczesne komputery z inteligencją sensu stricte nie mają nic wspólnego. A skoro tak - to i w procesorze inteligencji jest tyle, co w kalkulatorze czy liczydłach ...

Czym zatem jest procesor dla systemu komputerowego? W pierwszym przybliżeniu możecie go potraktować jako bardzo szybki i trochę bardziej skomplikowany kalkulator. Procesor potrafi przekształcać liczby binarne (dodawać, odejmować, mnożyć, dzielić, negować, itp.) w sposób definiowany poprzez program składający się również z liczb binarnych. A bardziej formalnie:

procesor (spotyka się również określenie CPU, ang. *Central Processing Unit* jest tym komponentem w systemie komputerowym, który interpretuje program i zgodnie z nim przetwarza dane.

Ze względu na wspomnianą "programalność" procesora, to właśnie jego istnienie determinowało zaliczenie pierwszych maszyn liczących do komputerów. Procesor wykonany w postaci jednego układu scalonego zwyczajowo nazywa się mikroprocesorem.

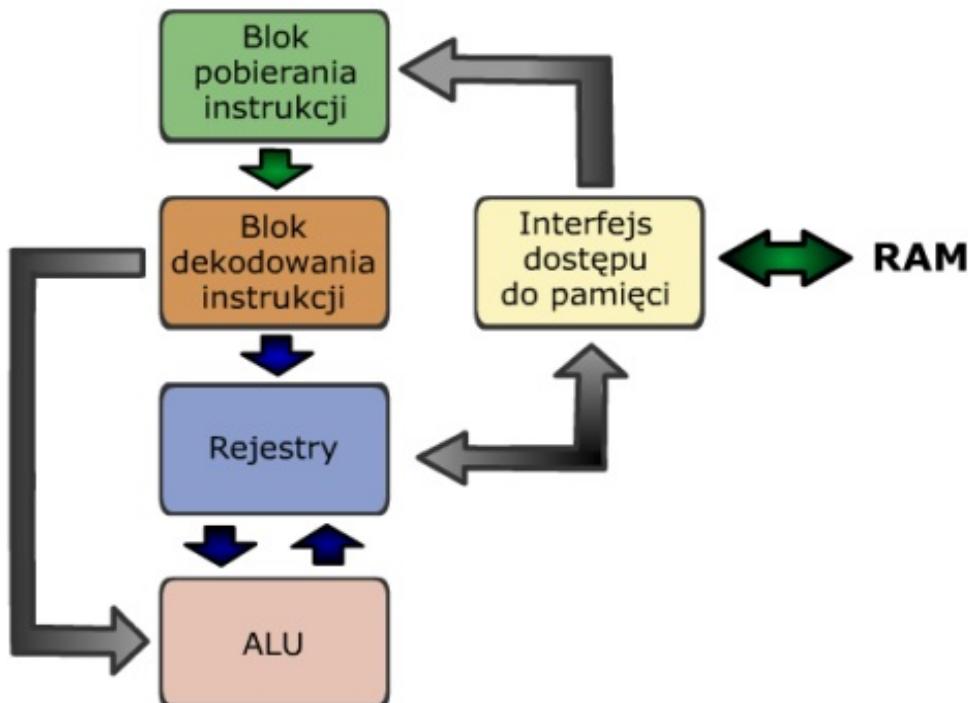
Podstawową funkcją większości procesorów (niezależnie od ich konstrukcji fizycznej) jest wykonanie programu, czyli sekwencji instrukcji przechowywanych w pamięci. Wykonanie każdej z tych instrukcji jest również rozbite na kroki. Zazwyczaj wyróżnia się następujące główne kroki potrzebne do wykonania pojedynczej instrukcji:

- *pobieranie rozkazów* - czyli odczyt poleceń z pamięci. Każde polecenie jest liczbą (możecie je traktować jako numer rozkazu). Ile poleceń "rozumie" nasz procesor - to już kwestia konstrukcji, ale zawsze jest to skończona liczba. Położenie instrukcji w pamięci jest zdefiniowane przez specjalny blok, nazywany licznikiem rozkazów lub słowem stanu programu (program counter, PC), który przechowuje adres aktualnie wykonywanej instrukcji w pamięci operacyjnej. Po pobraniu następuje zwiększenie PC o długość pobranego rozkazu (w jednostkach pamięci, czyli najczęściej w bajtach). Natomiast w przypadku konieczności wykonania skoku - zawartość PC może być zmieniana przez odpowiednią instrukcję procesora.
- *interpretacja rozkazów* - po odczytaniu polecenia z pamięci, w pierwszym etapie przetwarzania, procesor interpretuje polecenie. W trakcie tego kroku instrukcja jest rozbijana na części, które mają znaczenie dla poszczególnych bloków procesora. Sposób w jaki procesor interpretuje instrukcje, jest zależny od jego architektury i zestawu instrukcji (instruction set architecture, ISA). Przykładowo, w jednej długiej liczbie będącej instrukcją, początkowa część bitów (nazywana opcode) jest znacznikiem operacji do wykonania. Pozostała część bitów zazwyczaj zawiera informacje wymagane przez instrukcję, taką jak adresy składników dla operacji dodawania. Przykład dekodowania instrukcji dodawania dla procesora MIPS32:

MIPS 32 – Instrukcja dodawania

001000	00001	00010	0000000101011110
OP Code	Addr 1	Addr 2	Wartość

- wykonywanie rozkazów - po pobraniu i zdekodowaniu przychodzi pora na rzeczywiste wykonanie polecenia. Podczas tego kroku różne moduły CPU współpracują ze sobą w celu wykonania polecenia. Rozpatrzmy jako przykład dodawanie dwóch liczb: jednostka arytmetyczno-logiczna (ALU) zawiera układ elektroniczny, który jest w stanie wykonać proste operacje arytmetyczno - logiczne (jak dodawanie, czy operacje logiczne na bitach) dla danych wejściowych znajdujących się w rejestrach. Układ ten jest więc uruchamiany, a następnie wynik dodawania jest umieszczany w rejestrze wyjściowym. Jeśli w wyniku dodawania powstanie zbyt duża liczba (taka, której nie można zapamiętać), znacznik przepelnienia w rejestrze zawierającym wynik może również być ustawiony.
Zwróćcie uwagę na fakt, iż wykonanie instrukcji odbywa się na danych znajdujących się w rejestrach. Z punktu widzenia procesora pobranie danych z pamięci operacyjnej do rejestrów, lub też odesłanie wyników do pamięci operacyjnej, jest niezależną instrukcją.
- zapis wyników - finalna faza - efekty pracy procesora są zapamiętywane w rejestrze wyjściowym, skąd mogą zostać odesłane do pamięci operacyjnej, lub też mogą być wykorzystane w kolejnym kroku obliczeń (przy wykonywaniu kolejnej instrukcji). Część instrukcji zmienia zawartość licznika instrukcji (PC) w celu wykonania skoku w programie (dzięki tej funkcji możliwe jest tworzenie pętli i instrukcji warunkowych). Inne natomiast zmieniają ustawienie znaczników w specjalnym rejestrze (zwany znacznikowym). Mogą być one później wykorzystywane do podejmowania decyzji o skoku, itp ... ale w tym wypadku zaczynamy się już zbliżać do poziomu programowania w asemblerze, co nie jest tematyką podręcznika. Wystarczy abyście zapamiętali, że wykonanie kolejnych instrukcji kończy się zapisaniem wyników, lecz wynik niekoniecznie musi być zapisywany w pamięci operacyjnej :).



Działanie procesora

Aby procesor miał możliwość wykonywania powyższych zadań musi dysponować małą pamięcią wewnętrzną, która wymagana jest do czasowego przechowywania danych i rozkazów (procesor musi

np. pamiętać lokalizację poprzedniego rozkazu po to, by "odnaleźć" rozkaz następny). Większość (choć nie wszystkie) nowoczesnych systemów komputerowych opiera się na założeniu, że procesor nie operuje na danych bezpośrednio umieszczonych w pamięci operacyjnej, lecz za każdym razem następuje przekazanie zarówno danych, jak i wyników obliczeń, z / do pamięci za pośrednictwem rejestrów.

REJESTRY

Rejestrami procesora będziemy nazywali wydzielone, bardzo szybkie komórki pamięci, umieszczone w samym procesorze. Umożliwiają one szybki dostęp do potrzebnych danych lub rozkazów.

Zatem, jak to działa? Ponieważ procesor to nic innego jak taki duży i skomplikowany kalkulator, który dodatkowo pobiera informację o czynności do wykonania z pamięci operacyjnej, zobaczymy jak przebiegałoby policzenie przykładowego wyrażenia na kalkulatorze. Przyjmijmy, że komputer ma za zadanie obliczyć wartość następującego wyrażenia:

$$(x+y)*(x-z)$$

Jeśli liczymy to my, korzystając z kalkulatora, to możemy uzyskać wynik wykonując następujące czynności:

1. Najpierw ustalam, jaka jest wartość danych oznaczonych we wzorze literami x, y, z.
2. Wpisuję wartość x
3. Naciskam przycisk +
4. Wpisuję wartość y
5. Naciskam przycisk =
6. Odczytuję obliczoną wartość z wyświetlacza i zapisuję ją na kartce oznaczając jako t
7. Naciskam przycisk AC (kasuj)
8. Wpisuję wartość x
9. Naciskam przycisk -
10. Wpisuję wartość z
11. Naciskam przycisk =
12. Naciskam przycisk *
13. Wpisuję wartość t
14. Naciskam przycisk =
15. Odczytuję wynik działania z wyświetlacza i zapisuję jako obliczoną wartość w.

Procesor realizuje to zadanie w zbliżony sposób, przy czym każda podstawowa czynność to pojedynczy rozkaz procesora. Założymy, że rozkazy procesora mogą mieć jedną z trzech postaci (zbliżamy się w ten sposób do maszyny Turinga ;):

- pobierz dane z pamięci operacyjnej do rejestru (rejestr to "prywatna" jednostka pamięci należąca do procesora);
- wykonaj operacje arytmetyczne na danych w rejestrach;
- prześlij dane z rejestru do pamięci operacyjnej.

Założymy też, że przed uruchomieniem obliczeń, gdzieś w pamięci operacyjnej, pod adresami nazwanymi dla wygody x, y, z (zamiast np. 0xfe035ad8, bo tak naprawdę tylko takie adresy akceptuje procesor) umieszczone są wartości danych, dla których ma zostać wykonane działanie $(x+y)*(x-z)$. Lista rozkazów dla procesora realizujących obliczenie wyrażenia (bez pobierania z zewnętrznych danych wejściowych i wyświetlania wyników) będzie więc miała postać:

1. Pobierz wartość spod adresu x do rejestru R1
2. Pobierz wartość spod adresu y do rejestru R2

3. Dodaj zawartość rejestrów R1 i R2 i wynik wstaw do R1
4. Prześlij wynik z rejestru R1 pod adres t
5. Pobierz wartość spod adresu x do rejestru R1
6. Pobierz wartość spod adresu z do rejestru R2
7. Odejmij zawartość rejestru R2 od rejestru R1 i wynik wstaw do R1
8. Pobierz wartość spod adresu t do rejestru R2
9. Pomnóż zawartość rejestrów R1 i R2 i wynik wstaw do R1
10. Prześlij zawartość z rejestru R1 pod adres w.

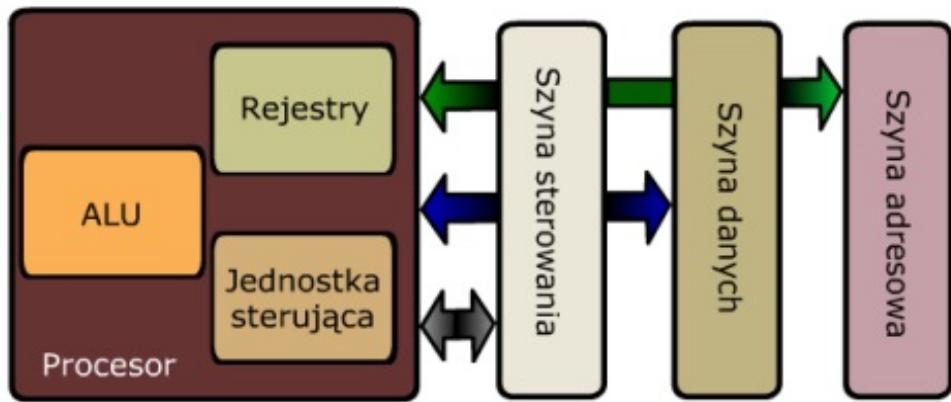
I w zasadzie to wszystko co potrafi zrobić procesor ... ;). Tyle, że te działania wykonuje bardzo szybko.

Liczba rejestrów we współczesnych procesorach różni się w zależności od producenta. Kiedyś można było stwierdzić, że im więcej rejestrów, tym komputer szybszy. Aktualnie to stwierdzenie niekoniecznie musi być prawdziwe, lecz zazwyczaj odpowiada prawdzie. Poniżej znajdziecie liczbę rejestrów wykorzystywanych w wybranych historycznych i współczesnych procesorach:

Zgodne z x86-64 (np. Intel Core)	16	16
Pentium 4	8	8
Athlon MP	8	8
Opteron 240	16	16
Itanium 2	128	128
UltraSPARC IIIi	32	32
Power PC 3	32	32

BUDOWA WEWNĘTRZNA PROCESORA

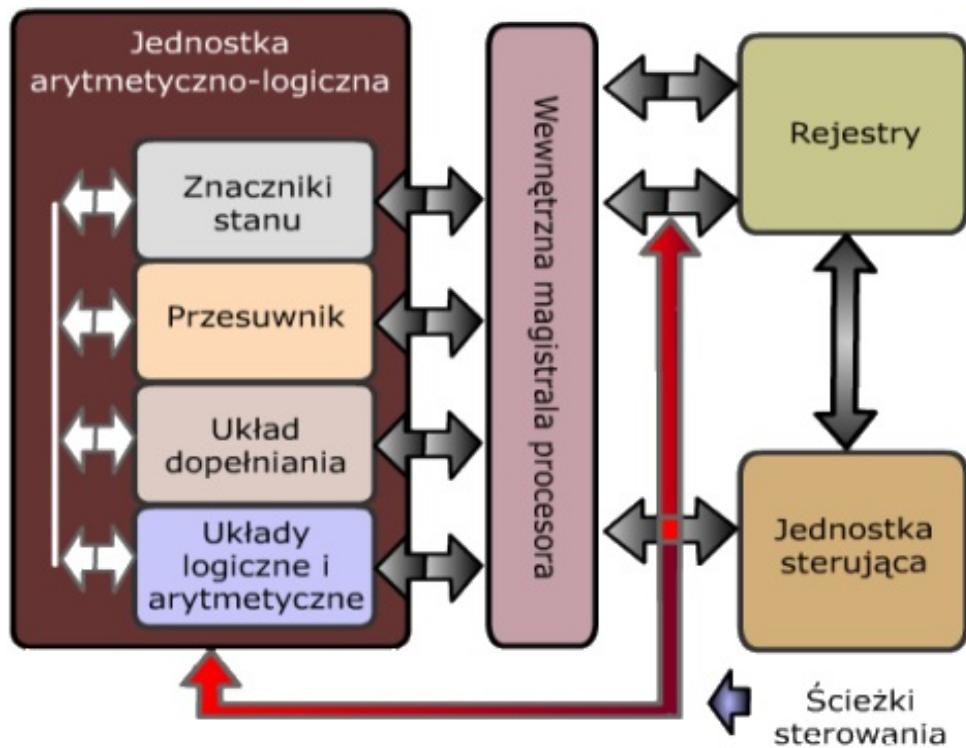
Każdy współczesny procesor jest zbudowany z kilku zespołów funkcjonalnych. Zaczniemy od najbardziej uproszczonego opisu. Poniżej przedstawione są dwa rysunki, które ukazują nam budowę procesora w pierwszym przybliżeniu. Pierwszy z nich przedstawia główne bloki funkcjonalne procesora, w zasadzie odpowiadające odpowiednim blokom funkcjonalnym ze schematu działania procesora. Blok pobierania instrukcji i blok dekodowania instrukcji są umieszczone w jednostce sterującej, natomiast jawnie pojawiają się rejesty i jednostka arytmetyczno-logiczna. Dokładniej rozrysowana jest tutaj komunikacja procesora z pamięcią, ale o tym będzie więcej w następnym segmencie.



Budowa procesora

Komunikacja między procesorem a resztą systemu komputerowego odbywa się poprzez magistralę systemową, której poszczególne składniki (szyny danych, adresowa i sterowania) mogą, ale nie muszą być fizycznie od siebie odseparowane.

Nieco dokładniej strukturę wewnętrzną procesora przedstawia kolejny rysunek. Element określony jako **wewnętrzna magistrala procesora** zajmuje się przesyłaniem danych między różnymi rejestrami a ALU (w rzeczywistości ALU operuje jedynie na danych zaczerpniętych z wewnętrznej pamięci procesora). Rysunek przedstawia również w zarysie poszczególne elementy jednostki arytmetyczno-logicznej.



Procesor: struktura wewnętrzna

Zapamiętajcie, że tak jak procesor jest "sercem" komputera, tak ALU jest "sercem" procesora. Rejestry są używane do przechowywania danych wewnętrznych w procesorze, a niektóre z nich zawierają informacje potrzebne do zarządzania porządkowaniem rozkazów (np. słowo stanu programu), pozostałe zawierają dane przeznaczone dla ALU, pamięci i modułów wejścia-wyjścia lub dane przekazane przez te jednostki. Wewnętrzne ścieżki danych są używane do przenoszenia danych pomiędzy rejestrami oraz między rejestrami a ALU. Zewnętrzne ścieżki danych łączą rejesty z pamięcią i modułami wejścia-wyjścia, często za pomocą magistrali systemowej. Jednostka sterująca koordynuje wykonanie operacji wewnętrz procesora.

PROCESOR - TAKSONOMIA I UKŁADY TOWARZYSZĄCE

POJĘCIA PODSTAWOWE CHARAKTERYZUJĄCE PROCESOR

Często spotkacie się z różnymi określeniami pojawiającymi się w opisie procesora, o których jeszcze nie mówiliśmy. Ten rozdział ma za zadanie przybliżyć Wam, co oznaczają pojęcia słowa maszynowego, architektury procesora, itp... Najbardziej ogólnym jest architektura procesora, która w uproszczeniu oznacza wszystkie najważniejsze z punktu widzenia budowy i funkcjonalności cechy układu, które związane są zarówno z jego modelem programowym, jak i fizyczną budową. Model programowy procesora (ang. *Instruction Set Architecture*) to zestaw instrukcji procesora oraz innych jego cech (np. liczby i numeracji rejestrów czy sposobu adresowania pamięci) istotnych z punktu widzenia programisty, które są niezależne od ich wewnętrznej realizacji w procesorze.

Z kolei fizyczna budowa układu nazywana jest często mikroarchitekturą (ang. *microarchitecture*). To szczegółowa implementacja danego modelu programowego, która związana jest z rzeczywistym wykonywaniem operacji przez procesor. Co ciekawe, operacje te mogą być realizowane w zupełnie odmienny sposób niż wynikałoby to z modelu programowego. Przykładem może być tutaj wykonywanie kilku instrukcji poza kolejnością lub przewidywanie skoków warunkowych, czego nie znajdziemy w modelu programowym.

Mikroarchitekturę procesora przedstawia się jako zbiór diagramów określających poszczególne jego elementy, takie jak bloki funkcjonalne czy jednostki wykonawcze oraz występujące między nimi połączenia. W mikroarchitekturze nie określa się natomiast faktycznej, fizycznej implementacji obwodów logicznych w krzemowej strukturze układu, a więc składających się na nie bramek i wchodzących w ich skład tranzystorów. W tym miejscu warto podkreślić, że procesory realizujące ten sam model programowy mogą w znaczący sposób różnić się od siebie swoją mikroarchitekturą – dobrym przykładem są układy firm AMD i Intel, które są zgodne z programową architekturą x86, ale są całkowicie różnymi układami pod względem swojej mikroarchirektury.

W tym rozdziale zajmiemy się głównie mikroarchitekturą procesorów i różnymi określeniami z nią związanymi.

SŁOWO MASZYNOWE, SZEROKOŚĆ SZYNY DANYCH

Sposób w jaki procesor pamięta przetwarzane liczby całkowite jest jego cechą konstrukcyjną, która wpływa na praktycznie wszystkie dostępne funkcje. Mówiliśmy wcześniej o rejestrach. Pojedynczy register jest określona liczbą bitów, czyli logicznych zer lub jedynek. W zależności od ilości bitów przechowywanych w rejestrze, i co za tym idzie – przetwarzanych przez procesor, zmienia się rozmiar i precyzja danych na których operuje. Liczba bitów przetwarzanych w jednym cyklu procesora przez ALU jest nazywana jego **słowem maszynowym, szerokością szyny danych**, bądź po angielsku: *word size, bit width, data path width* lub *integer precision*.

Napotykając określenie - procesor 32 bitowy, 64 bitowy, itd. powinniście wiedzieć, że chodzi tu o długość rejestrów danych, a co za tym idzie - długość słowa przetwarzanego przez ALU procesora w jednej instrukcji.

Długość poszczególnych rejestrów różni się dla różnych procesorów i architektur, czasami nawet różni się dla poszczególnych modułów jednego procesora. Przykładowo 8-bitowy procesor jest w stanie w jednej instrukcji przetwarzać 8 bitów, co oznacza, że może pracować z liczbami od 0 do 255. Lecz jeśli register używany do przekazywania adresów (szerokość szyny adresowej) miałby również 8 bitów, to komputer oparty na takim procesorze miałby fizycznie ograniczoną wielkość pamięci do 256 komórek (maksymalna możliwa pojemność RAM wynosiłaby 256 bajtów!). Dlatego też zazwyczaj szyna i

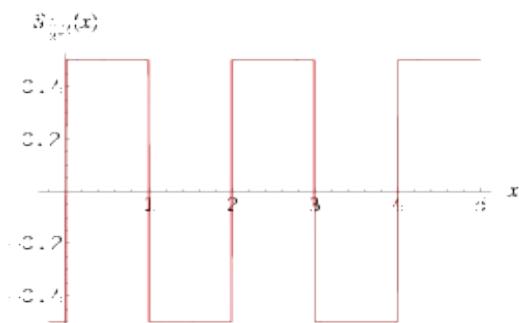
rejestry adresowe są dłuższe, dla procesorów 8-bitowych miały zwyczajowo 16 bitów, co dawało 65536 możliwych adresów i 64 kB RAM. Tyle właśnie maksymalnie mogły mieć komputery ZX Spectrum, Atari czy Commodore. Dla CPU wykorzystującego 32 bity do reprezentowania adresu, maksymalna możliwa pamięć (czyli **przestrzeń adresowa**) wynosi 4 GB (giga bajty). Oczywiście, istnieją metody wirtualnego powiększania pamięci, choćby poprzez stronicowanie, które pozwalają zwiększyć przestrzeń adresową bez poszerzania szyny danych, lecz ich stosowanie zawsze prowadzi do dodatkowego narzutu.

Dłuższe słowo maszynowe czy szersza szyna adresowa wymagają większej ilości tranzystorów i zbudowanych z nich bramek logicznych do obsługi, co pociąga za sobą większy koszt, stopień skomplikowania czy też zużycie energii. Dlatego też często stosuje się 4- lub 8-bitowe mikrokontrolery w systemach osadzonych, mimo że od dawna dostępne są układy 16, 32, 64, czy nawet 128-bitowe. Jednak w systemach komputerowych stosowanych jako komputery osobiste czy serwery, zysk ze wzrostu szybkości czy też dostępnej pamięci dla procesorów o większym słowie maszynowym, zdecydowanie przewyższa zwiększoną koszt czy zużycie energii przez taki procesor.

Zatem, wybierając procesor do komputera osobistego warto kupować taki, który ma jak najdłuższe słwo maszynowe, czyli wspólnie - procesory 64 bitowe. Mimo, że ciągle większość oprogramowania nie potrafi wykorzystać tej cechy, to jednak procesor dłużej pozostanie nowoczesny i wydajny.

Częstotliwość zegara

Większość procesorów, podobnie jak większość sekwencyjnych układów elektronicznych, jest z natury synchroniczna. To oznacza, że aby mogły wykonywać swoje zadania muszą mieć podany sygnał synchronizujący. Taki sygnał, nazywany zegarowym, najczęściej przyjmuje formę prostokątnej fali w czasie:



Patrząc na powyższy rysunek możemy powiedzieć, że jest to po prostu taki sygnał, co pojawia się i znika ... jest napięcie - nie ma, jest - nie ma, jest - nie ma ... i tak miliony razy na sekundę ;). Miarą tego, ile razy sygnał zmienia się w ciągu sekundy, jest częstotliwość. Ale co to ma wspólnego z procesorem?

Nawet nie wyobrażacie sobie jak dużo. Powiedzieliśmy wcześniej, że procesor jest urządzeniem synchronicznym, co oznacza, że kolejne polecenie (kolejny krok wykonania instrukcji) jest inicjowane zmianą sygnału zegarowego. Co oznacza, iż im większa częstotliwość (ilość zmian sygnału zegarowego) - tym szybciej nasz procesor jest w stanie liczyć. Niestety, każdy układ elektroniczny ma swoją graniczną częstotliwość powyżej której nie będzie działał. Obliczając maksymalny czas propagacji sygnału elektrycznego przez obwody procesora (zmiany napięcia poruszają się z prędkością światła) jesteśmy w stanie wyznaczyć graniczną częstotliwość zegara. W praktyce, maksymalna możliwa częstotliwość jest zawsze niższa od granicznej. Jest to powodowane poprzez niedokładności wykonania procesora, zbyt dużą ilość wydzielanego ciepła, problemy z utrzymaniem całego układu w synchronicznej pracy, etc. Dlatego producenci wyznaczają doświadczalnie maksymalną dopuszczalną częstotliwość zegara dla każdego egzemplarza procesora.

Wiemy już, że częstotliwość zegara jest ważnym elementem wydajności procesora. Jednakże nie znaczy to bezpośrednio, że procesor taktowany wyższą częstotliwością będzie zawsze szybszy niż inny - taktowany niższą. Taki związek nie zachodzi między innymi dla

procesorów o różnej długości słowa maszynowego - 32 bitowy procesor taktowany zegarem 1 GHz będzie prawdopodobnie wolniejszy niż 64 bitowy taktowany 800 MHz w typowych zastosowaniach. Wydajność procesora jest w dużym stopniu uwarunkowana również jego budową wewnętrzną, dlatego też procesory AMD zazwyczaj przy wolniejszym zegarze osiągają taką samą wydajność jak procesory firmy Intel, a PowerPC o zegarze 500 MHz jest szybszy niż Pentium IV 2 GHz.

LISTA INSTRUKCJI PROCESORA

Zapewne każdy z Was wie, że wymiana procesora na szybszy nie pociąga za sobą wymiany wszystkich programów. Wiecie również, że program pisany dla Mac'a nie da się uruchmić na PC. Dlaczego?

Poszczególne procesory mogą mieć różne listy instrukcji, czyli rozkazów które rozumieją i potrafią wykonać. Zazwyczaj procesory jednej rodziny mają taką samą listę instrukcji (np. x86 to wspólna nazwa dla zestawu instrukcji rozumianych przez wszystkie procesory Intel od I386 do współczesnych rozwiązań oraz procesory AMD, ale totalnie nieznana procesorom serii UltraSPARC czy ARM).

Przy projektowaniu zestawów instrukcji dla nowej rodziny procesorów zazwyczaj przyjmuje się jedno z dwóch podejść:

- **CISC** (*Complex Instruction Set Computers*) – tworzona jest lista rozkazów o następujących cechach:
 - duża liczba instrukcji;
 - mała optymalizacja – niektóre rozkazy potrzebują dużej liczby cykli procesora do wykonania;
 - występowanie złożonych, specjalistycznych rozkazów;
 - duża liczba trybów adresowania;
 - do pamięci może się odwoływać bezpośrednio duża liczba rozkazów;
 - mniejsza od RISC-ów częstotliwość taktowania procesora;
 - powolne działanie dekodera rozkazów.

Jest to architektura zestawu instrukcji dla mikroprocesora, w którym każda instrukcja może wykonać kilka operacji niskiego poziomu, jak na przykład pobranie z pamięci, operację arytmetyczną, albo zapisanie do pamięci, a to wszystko w jednej instrukcji. Przed powstaniem procesorów RISC, wielu komputerowych architektów próbowało zmostkować lukę semantyczną, aby zaprojektować zestawy instrukcji, które wspierałyby języki programowania wysokiego poziomu przez dostarczenie instrukcji wysokiego poziomu (np. call i return, instrukcje pętli i kompleksowe tryby adresowania). Wszystko to miało na celu połączenie struktur danych i szeregiw dostępu do nich w jedną instrukcję procesora. Rezultatem tego były programy o mniejszym rozmiarze i z mniejszą ilością odwołań do pamięci, co w tamtym czasie bardzo ograniczyło koszty pojedynczego komputera.

Z reguły procesory wykonane w architekturze CISC działają wolniej niż procesory RISC (opisane poniżej, o przeciwnych założeniach), działające z tą samą częstotliwością.

Przykłady rodzin procesorów o architekturze CISC to:

- System/360;
- VAX;
- PDP-11;
- AMD;
- x86;
- M68000.

- **RISC** (*Rationalized Instruction Set Computers*, wcześniej skrót od *Reduced Instruction Set Computers*) - nazwa architektury mikroprocesorów, która została przedstawiona pod koniec lat 70

w teoretycznych pracach na uniwersytecie Berkeley oraz w wynikach badań Johna Cocke z Thomas J. Watson Research Center.

Ówczesne procesory (budowane w architekturze CISC) charakteryzowały się bardzo rozbudowaną listą rozkazów, ale jak wykazały badania, tylko nieliczna ich część była wykorzystywana w statystycznym programie. Okazało się na przykład, że ponad 50% rozkazów w kodzie to zwykłe przypisania (zapis zawartości rejestru do pamięci i odwrotnie).

Ponadto badania wykazały, że podczas działania programu ok. 26-45% wykonywanych instrukcji to instrukcje wywołania podprocedur lub instrukcje obsługujące pętle, ok. 13-15% to wspomniane przypisania, 7-21% to instrukcje warunkowe (jeśli warunek to ...), natomiast reszta stanowi tylko 1-3%.

W związku z powyższym zaprezentowano architekturę mikroprocesorów, w której uwzględniono wyniki badań. Jej podstawowe cechy to:

- zredukowana liczba rozkazów do niezbędnego minimum. Ich liczba wynosi kilkadziesiąt, podczas gdy w procesorach CISC sięga setek. Upraszczają to znacznie dekoder rozkazów;
- redukcja trybów adresowania, dzięki czemu kody rozkazów są prostsze, bardziej zunifikowane, co dodatkowo upraszcza wspomniany wcześniej dekoder rozkazów. Ponadto wprowadzono tryb adresowania, który ogranicza ilość przesyłań - większość operacji wykonuje się według schematu: rejestr_C = rejestr_A \ operacja \ rejestr_B ;
- ograniczenie komunikacji pomiędzy pamięcią, a procesorem. Do przesyłania danych pomiędzy pamięcią a rejestrami służą przede wszystkim dedykowane instrukcje, które zwykle nazywają się *load* (załóż z pamięci), oraz *store* (zapisz do pamięci); pozostałe instrukcje mogą działać wyłącznie na rejestrach. Schemat działania na liczbach znajdujących się w pamięci jest następujący: załóż daną z pamięci do rejestrów, na zawartości rejestrów wykonaj działanie, przepisz wynik z rejestrów do pamięci (ten schemat działania przedstawiliśmy w animacji wprowadzającej w tematykę procesorów);
- zwiększenie liczby rejestrów (przykładowo: 32, 192, 256, podczas gdy np. w architekturze x86 jest zaledwie 8 rejestrów), co również ma wpływ na zmniejszenie liczby odwołań do pamięci;
- dzięki przetwarzaniu potokowemu (ang. *pipelining*) wszystkie rozkazy wykonują się w jednym cyklu maszynowym, co pozwala na znaczne uproszczenie bloku wykonawczego, a zastosowanie superskalarności także na zrównoleglenie wykonywania rozkazów. Dodatkowo czas reakcji na przerwania jest krótszy.

Pierwszym procesorem zaprojektowanym w oparciu o architekturę RISC był RCA1802 wyprodukowany przez firmę RCA. Obecnie popularne procesory Intela z punktu widzenia programisty są widziane jako CISC, ale ich rdzeń jest RISC-owy. Rozkazy CISC są rozbijane na mikrorozkazy (ang. *microops*), które są następnie wykonywane przez RISC-owy blok wykonawczy.

Pozostałe metody doboru zestawu instrukcji dla procesora są zdecydowanie mniej popularne, ale wymienimy je dla porządku:

- **VLIW** (*Very Long Instruction Word*) oparty jest na bardzo długiej instrukcji wykonywanej przez procesor, która w zasadzie składa się z wielu prostych instrukcji. Różnica VLIW w stosunku do CISC czy RISC oparta jest głównie na założeniu, że to nie procesor decyduje które z poleceń mogą być wykonane równolegle. Podczas komplikacji instrukcje, które mogą być wykonywane równolegle, są łączone w jedną długą instrukcję i jako taka są przekazywane do wykonania równoległego na procesorze. Dzięki temu upraszcza się architektura procesora;
- **MISC** (*Minimal Instruction Set Computer*) jest to architektura procesora z bardzo małą liczbą podstawowych operacji, które są zgodne z opcode. Takie zestawy instrukcji są częściej oparte na stosie, niż na rejestrach, mając na celu zmniejszenie rozmiaru określników operacji (opcode). Architektura maszyny stosowej jest właściwie prostsza od kiedy wszystkie instrukcje operują na szczytce większości zapisów w stosie. Rezultatem tego jest mniejszy zestaw instrukcji, mniejsza i

szybsza jednostka do dekodowania instrukcji i szybsze wykonanie pojedynczych instrukcji. Wadą takiego podejścia jest to, że trudniej jest wykonywać wiele instrukcji równolegle (częściej daną wejściową do kolejnej instrukcji jest wynik wykonania poprzedniej);

- **ZISC (Zero Instruction Set Computer)** jest to technologia oparta na pomysłach wziętych ze sztucznej sieci neuronowej. Pomyśl został wynaleziony przez Guy'a Paillet'a oraz został przez niego rozwinięty przy współpracy z dr Pascal-em Tannhof-em z IBM. Pierwsza generacja chipu ZISC zawierała 36 niezależnych komórek, które mogą być uważane za neurony lub równoległe procesory. Każda z nich może porównać wektor wejściowy, którego rozmiar może osiągnąć do 64 bajtów, z podobnym wektorem przechowywanym w komórkach pamięci. Jeśli wektor wejściowy odpowiada wektorowi w komórce pamięci to komórka przechodzi w stan aktywacji. Sygnał wyjściowy jest ważoną sumą stanów aktywacji komórek wejściowych. Równoległość jest kluczem do szybkości systemów ZISC, które eliminują krok seryjnego ładowania i porównywania wzorca rozkazu dla kolejnych instrukcji. Kolejnym kluczowym czynnikiem ZISC jest skalowalność. Sieć ZISC może być rozszerzona przez dodanie większej ilości urządzeń ZISC bez obniżenia szybkości rozpoznawania (komputery ZISC są wykorzystywane głównie do rozpoznawania wzorców) – sieć z ilością 10 000 lub więcej komórek nie jest niczym nadzwyczajnym. Dzisiejsze chipy ZISC zawierają 78 neuronów na chip i mogą znaleźć dopasowanie wśród 1 000 000 wzorców zaledwie w jedną sekundę operując z prędkością przynajmniej 50Mhz. Następna generacja może zawierać nawet 1 000 neuronów lub więcej w jednym chipie. Praktyczne użytkowanie technologii ZISC jest skupione na rozpoznawaniu wzorców, ochronie, wyszukiwaniu informacji (*data mining*) i innych podobnych zadaniach.

DODATKOWE ELEMENTY PROCESORA

Do tej pory omawialiśmy modelowy (uproszczony) procesor. W rzeczywistości w opisie współczesnych rozwiązań pojawi się wiele nowych terminów, elementów procesora, których nie ma na żadnym z zamieszczonych rysunków. Postaramy się je przybliżyć, bez szczegółowego wgłębiania się w ich działanie.

KOPROCESOR

Pierwszym popularnym (a w zasadzie już zawsze obowiązkowym) rozszerzeniem jest koprocesor. Normalny (czyt. przestarzały) procesor komputera wykonywał obliczenia jedynie na liczbach całkowitych - jakiekolwiek operacje na liczbach zmiennoprzecinkowych wymagały wykonywania wielu instrukcji prostych na liczbach całkowitych. Innymi słowy - dodanie 0.1 do 0.1 zajmowało kilkanaście razy więcej czasu niż dodanie 1+1. Od jakiegoś czasu jednak wszystkie procesory mają wbudowaną jednostkę do obliczeń zmiennoprzecinkowych. Jest to odpowiednik ALU, tyle że pozwalający wykonywać podstawowe operacje na zmiennych, które nie są liczbami całkowitymi, co powoduje znaczące przyspieszenie takich obliczeń.

CACHE

Następnym elementem, który pojawia się wyjątkowo często, jest pamięć podręczna (ang. *cache*) procesora. Normalne pamięci operacyjne (a więc to, co znacie pod pojęciem RAM i co jest opisane dalej w tym podręczniku) są zdecydowanie wolniejsze od samego procesora. Stosowanie w komputerach osobistych pamięci RAM będących w stanie pracować równie szybko jak procesor byłoby bajońsko drogie - zamiast tego stosuje się pamięci podręczne, czyli mechanizm, w którym ostatnio pobierane dane z pamięci operacyjnej są przechowywane w pamięci o zdecydowanie lepszych parametrach szybkościowych. Cache jest elementem właściwie wszystkich podsystemów - nie tylko procesor, ale również np. dostęp do dysku jest buforowany w pamięci RAM. We współczesnych procesorach są 2 lub 3 poziomy pamięci cache: L1 (zintegrowana z procesorem), a także L2 i L3 (umieszczone w jednym chipie razem z procesorem, lub na płycie głównej). Każdy kolejny poziom

cache jest wolniejszą pamięcią, ale za to o większej pojemności.

Potokowość

Współczesne procesory często określa się mianem potokowych. Potokowość (ang. *pipelining*) jest techniką budowy procesorów polegającą na podziale logiki procesora odpowiedzialnej za proces wykonywania programu (instrukcji procesora) na specjalizowane grupy w taki sposób, aby każda z grup wykonywała część pracy związanej z wykonaniem rozkazu. Grupy te są połączone sekwencyjnie i wykonują pracę równocześnie, pobierając dane od poprzedniego elementu w sekwencji. W każdej z tych grup rozkaz jest na innym stadium wykonania. Można to porównać do taśmy produkcyjnej. W uproszczeniu, potok wykonania instrukcji procesora może wyglądać następująco:



1. Pobranie instrukcji z pamięci - ang. *instruction fetch* (IF);
2. Zdekodowanie instrukcji - ang. *instruction decode* (ID);
3. Wykonanie instrukcji - ang. *execute* (EX);
4. Dostęp do pamięci - ang. *memory access* (MEM);
5. Zapisanie wyników działania instrukcji - ang. *store; write back* (WB).

W powyższym 5-stopniowym potoku, przejście przez wszystkie stopnie potoku (wykonanie jednej instrukcji) zabiera co najmniej 5 cykli zegarowych. Jednak ze względu na jednoczesną pracę wszystkich stopni potoku, jednocześnie wykonywanych jest 5 rozkazów procesora, każdy w innym stadium wykonania. Oznacza to, że taki procesor w każdym cyklu zegara rozpoczyna i kończy wykonanie jednej instrukcji i statystycznie wykonuje rozkaz w jednym cyklu zegara. Każdy ze stopni potoku wykonuje mniej pracy w porównaniu do pojedynczej logiki, dzięki czemu może wykonać ją szybciej - z większą częstotliwością - tak więc dodatkowe zwiększenie liczby stopni umożliwia osiągnięcie coraz wyższych częstotliwości pracy.

Podstawowym wrogiem techniki potoku są rozkazy skoku, powodujące w najgorszym wypadku potrzebę przeczyśczenia całego potoku i wycofania rozkazów, które następowały zaraz po instrukcji skoku, a następnie rozpoczęcie zapełniania potoku od początku od adresu, do którego następował skok. Taki rozkaz skoku może powodować ogromne opóźnienia w wykonywaniu programu - tym większe, im większa jest długość potoku. Dodatkowo szacuje się, że dla modelu programowego x86 taki skok występuje co kilkanaście rozkazów. Z tego powodu niektóre architektury programowe (np. SPARC) zakładały zawsze wykonanie jednego lub większej ilości rozkazów następujących po rozkazie skoku, tzw. skok opóźniony. Stosuje się także skomplikowane metody predykcji skoku lub metody programowania bez użycia skoków.

Wielowatkowość (Hyperthreading) i Wielordzeniowość (Multicore)

Inną metodą zwiększania wydajności procesora jest równolegle wykonywanie wielu instrukcji zupełnie niezależnie od siebie. Najprościej wydawało się wykonywać jednocześnie kilka różnych programów (np. Word z pracującym w tle Winampem odgrywającym muzykę), każdy na swoim procesorze.

Jedną z technik wykorzystywanych do tego celu jest przetwarzanie współbieżne (ang. *multiprocessing*, MP). W początkowym wydaniu wykorzystywane było tylko tak zwane symetryczne przetwarzanie

współbieżne (SMP), czyli niewielka liczba zupełnie niezależnych procesorów miała dostęp do wspólnej pamięci operacyjnej. Przy takim podejściu, każdy z procesorów musiał być wyposażony w dodatkowe układy sprzętowe zapewniające utrzymywanie spójnej i aktualnej zawartości pamięci (a dokładniej - jej kopii w pamięci cache). Takie podejście sprawdza się w praktyce do systemów o maksymalnie 8 procesorach. W przypadku systemów o większej liczbie procesorów, utrzymywanie spójnej pamięci staje się bardzo trudne i czasochłonne. Dlatego też opracowano architekturę niejednorodnego dostępu do pamięci (ang. *non-uniform memory access*, NUMA) co pozwala na stosowanie tysięcy procesorów w jednym komputerze.

Jednakże NUMA jest architekturą wykorzystywana jedynie przez superkomputery. Natomiast SMP pojawia się w komputerach osobistych lub małych i średnich serwerach. Niektóre procesory, jak Itanium czy Athlon MP, można montować po 2 lub 4 na specjalnych płytach. Co więcej - teraz pojawiły się na rynku rozwiązania jeszcze bardziej zaawansowane, gdzie dwa procesory umieszczone są w jednej kości (chipie) - są to tzw. **procesory dwurdzeniowe** jak np. Intel Core 2 Duo. Dwa rdzenie - oznacza dwa procesory, każdy z wielopotokowym jądrem i własną pamięcią cache poziomu L1. Czyli, mając jeden chip i jeden wiatrak, mamy komputer wieloprocesorowy :)

Oprócz możliwości uruchomienia wielu programów naraz, nowoczesne systemy operacyjne umożliwiają także uruchomienie wielu toków wykonania (wątków) w jednym programie. Dokładniej o różnicach między wątkiem a procesem (niezależnym programem) powiemy w [lekcji 6](#). Już teraz jednak zaznaczmy, że różne wątki jednego programu mają tą samą przestrzeń adresową, co znacznie upraszcza budowę procesora, który mógłby wykonywać je równolegle. Taką technologię nazywa się MT (multithreading). Przykładem procesorów z zaimplementowaną tą technologią są procesory Intel Pentium IV HT.

W przypadku wielowątkowości warto pamiętać, że procesor HT nigdy nie będzie dwa razy szybszy od odpowiadającego mu procesora jednowątkowego. Wzrost wydajności można zaobserwować jedynie w aplikacjach specjalnie do tego przystosowanych, jak niektóre gry czy Adobe Photoshop, natomiast w większości przypadków komputer z procesorem HT będzie miał porównywalną wydajność do komputera ze zwykłym procesorem.

RÓWNOLEGŁE PRZETWARZANIE DANYCH

Zwiększenie wydajności i szybkości obliczeń (co daje w rezultacie szybszy system komputerowy) jest możliwe nie tylko poprzez dodawanie jednostek wykonawczych (ALU) czy też całych procesorów pracujących równolegle. W rzeczywistości do wielu zadań (przetwarzanie grafiki, obliczenia macierzowe, sztuczna inteligencja, statystyka) wykonujemy dokładnie taką samą instrukcję (ciąg instrukcji) na olbrzymiej ilości zmieniających się danych. Jeśli przyjrzymy się budowie procesora oraz przeanalizujemy sposób w jaki działa, szybko możemy dojść do wniosku, iż w przypadku takich operacji wystarczy zwieliokrotnić stosunkowo mało skomplikowany blok ALU, pozostałe elementy procesora pozostawiając bez zmian, a otrzymamy po jednym pobraniu i zdekodowaniu rozkazu możliwość jego wykonania dla wielu danych, częściowo równolegle, a jeśli nawet szeregowo - to dużo szybciej niż w przypadku pracy w zwykłym trybie.

Taki tryb pracy nazywany jest SIMD (*single instruction, multiple data*), a procesory, które go umożliwiają, nazywa się wektorowymi. W obecnych czasach popularność zdobyło jednak trochę inne podejście. Do zwykłego procesora dokładany jest dodatkowy blok funkcjonalny, który jest właśnie takim uproszczonym procesorem wektorowym. Uproszczonym - bo umożliwia wykonanie stosunkowo niewielkiej liczby operacji, zazwyczaj dobieranych tak, by maksymalnie przyspieszyć obsługę multimedialnych.

Handlowa nazwa takich układów to MMX (umożliwia wykonywanie operacji jedynie na liczbach całkowitych) czy SSE (operacje wykonywane są na liczbach całkowitych i zmiennoprzecinkowych) u Intel'a (każdy procesor z tymi oznaczeniami zawiera blok SIMD), 3DNow dla procesorów AMD czy AltiVec dla PowerPC.

MAGISTRALE SYSTEMOWE

System komputerowy zawiera pewną liczbę magistrali. Najważniejsza z nich to **magistrala systemowa** (niektóre systemy wykorzystują kilka magistrali systemowych) łącząca najważniejsze podzespoły komputera, czyli procesor, pamięć i układy wejścia-wyjścia.

Magistrala to zespół linii oraz układów przełączających służących do przesyłania sygnałów między połączonymi urządzeniami. W komputerach jest rodzajem "autostrady", którą dane przenoszą się pomiędzy poszczególnymi elementami komputera.

Zazwyczaj magistrala przesyła całe słowo maszynowe wykorzystywane przez wiele urządzeń. Jedno słowo maszynowe to 8, 16, 32, 64 lub więcej bitów, czyli napięć elektrycznych. Tak więc jedno słowo można przesłać albo jednym "przewodem" (połączeniem elektrycznym) jeden za drugim, albo jednocześnie 8, 16, 32, 64 połączeniami (w zależności od długości słowa). Pierwsze rozwiązanie nazywamy szeregowym, drugie - równoległy.

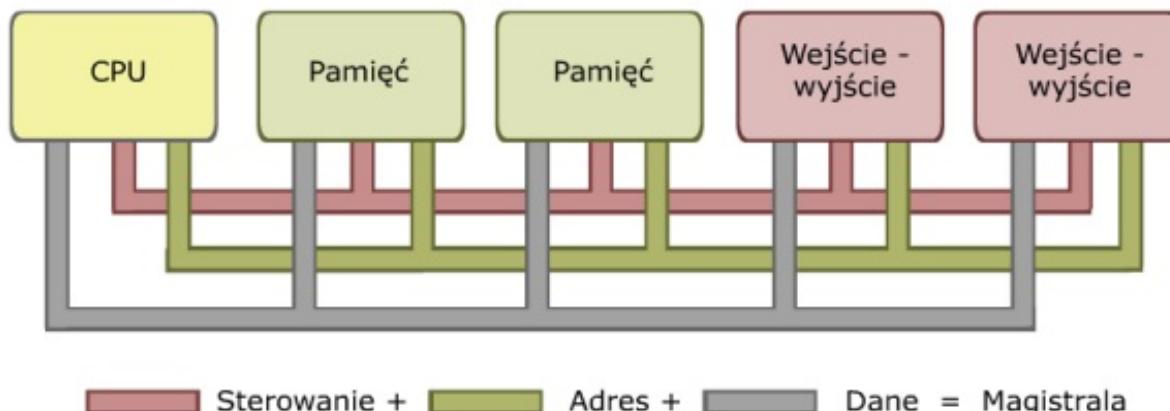
Magistrala systemowa jest w znakomitej większości przypadków magistralą równoległą. Korzysta z niej pewna liczba urządzeń, które poprzez nią komunikują się wykorzystując kilka linii komunikacyjnych, (np. 16, mówimy wtedy o magistrali 16-bitowej). Dostęp do magistrali jest sekwencyjny, co oznacza że w danej chwili może korzystać z niej tylko jedno urządzenie - gdyby w tym samym czasie nadawało kilka urządzeń, ich sygnały zakłócałyby się wzajemnie, co prowadziłoby do przekłamań w transmisji danych.

STRUKTURA MAGISTRALI

Magistrala systemowa składa się z wielu oddzielnych linii o określonym znaczeniu lub funkcji. Linie zawarte w magistrali można podzielić na trzy grupy (patrz rysunek niżej):

- linie danych;
- linie adresów;
- linie sterowania.

Opcjonalnie mogą pojawić się również linie zasilania, jednak nie zawsze one występują.



Połączenie magistrali z elementami systemu komputerowego

Linie danych służą do przenoszenia danych pomiędzy komponentami systemu. Linie te zwane są **szyną danych** (ang. *data bus*). Liczba linii szyny określa jej szerokość (przeważnie występuje 8, 16, 32, 64 linii), czyli ilość bitów, które można jednocześnie przesyłać. Szerokość szyny danych jest jednym z głównych czynników określających wydajność systemu komputerowego. Przykładowo, jeśli szyna danych ma szerokość 16 bitów, a wykonywany rozkaz 32 bity, to procesor chcąc nie chcąc musi łączyć się z modułem pamięci dwukrotnie w czasie tego cyklu rozkazu. Z tego względu optymalnym rozwiązaniem jest stosowanie szyny danych dopasowanej do szerokości rejestrów danych w procesorze.

Linie adresowe wykorzystywane są do określania skąd i dokąd przepływają informacje przesyłane szyną danych. Na przykład procesor, który chce odczytać dane z pamięci, umieszcza adres tych danych na szynie adresowej, a chcąc zapisać efekt ich wykonania wysyła na szynę adresową adres pamięci (lub modułu we-wy), do którego chce przesłać tenże wynik.

Istotnym parametrem również tej szyny jest jej szerokość. Szerokość szyny adresowej określa maksymalną pojemność pamięci systemu, a ściślej rzecz ujmując przestrzeń adresową, która nie musi być pamięcią, ale np. urządzeniem we-wy. Zwykle bity mniej znaczące służą do adresowania pamięci lub określonych portów wewnętrz modułu we-wy, zaś bity bardziej znaczące do wybrania określonego modułu (np. pamięci lub we-wy).

Alternatywnie do szyny adresowej i szyny danych istnieje także kombinowana szyna łącząca obie w jedną. Dane i adresy są przesyłane poprzez jedną taką szynę w obu kierunkach, za ich rozdzielenie odpowiedzialny jest wbudowany multiplekser. Jednakże rozwiązanie takie jako wolniejsze jest rzadko stosowane.

Linie sterowania (nazywane czasem również kontrolnymi) zapewniają regulację dostępu do linii danych i linii adresowych oraz umożliwiają sterowanie ich wykorzystaniem. Sygnały sterujące przesyłane pomiędzy modułami systemu komputerowego zawierają:

- rozkazy, które określają jakie operacje mają być wykonane;
- informacje regulujące czas (taktyczne), określające ważność danych i adresów.

Istnieją dwa główne typy szyny sterowania:

- synchroniczna – czas jaki występuje w przerwach pomiędzy wysyłaniem sygnałów sterowanych jest tylko i wyłącznie przez CPU;
- asynchroniczna – w przypadku tej szyny powolna pamięć lub jednostka I/O może według własnych potrzeb spowolnić czas przesyłania danych przez magistralę. Każda magistrala asynchroniczna może również pracować w trybie synchronicznym.

Reasumując, działanie magistrali systemowej opiera się na następujących zasadach:

- jeśli pewien moduł (nazwijmy go modułem X) zamierza przesłać dane do drugiego modułu (powiedzmy modułu Y), to musi:
 1. uzyskać dostęp do magistrali;
 2. przekazać dane za pośrednictwem magistrali;
- jeśli natomiast pewien moduł (moduł Y) zamierza pozyskać dane od drugiego modułu (modułu X), to musi:
 1. uzyskać dostęp do magistrali;
 2. przekazać zapotrzebowanie do modułu X przez odpowiednie linie sterowania i adresowe.

Przykład przebiegu operacji zapisu do pamięci (synchronicznie):

1. CPU ustala adres i „kładzie” go na magistrali (szyna adresowa)
2. CPU aktywuje CS (Chip Select)
3. Pamięć pobiera adres
4. CPU „kładzie” dane na magistrali (szyna danych)

5. CPU aktywuje funkcję „write” (szyna sterowania)
6. Pamięć pobiera dane i zapisuje je pod wcześniej pobrańym adresem
7. CPU deaktywuje funkcję „write”
8. CPU deaktywuje szynę danych

Przykład przebiegu operacji odczytu (synchronicznie):

1. CPU ustala adres i „kładzie” go na magistrali;
2. CPU aktywuje CS (Chip Select);
3. Pamięć pobiera adres
4. CPU aktywuje funkcję „read”
5. Pamięć „kładzie” dane na na magistrali
6. CPU pobiera dane
7. CPU deaktywuje funkcję „read”
8. CPU deaktywuje szynę danych

Dla tych, którzy wolą prezentacje wizualne, przygotowaliśmy [animacje](#) pokazującą w uproszczony sposób, jak przebiegają obie wyżej wymienione operacje jedna po drugiej, dostępną w wersji on-line podręcznika.

PAMIĘĆ OPERACYJNA

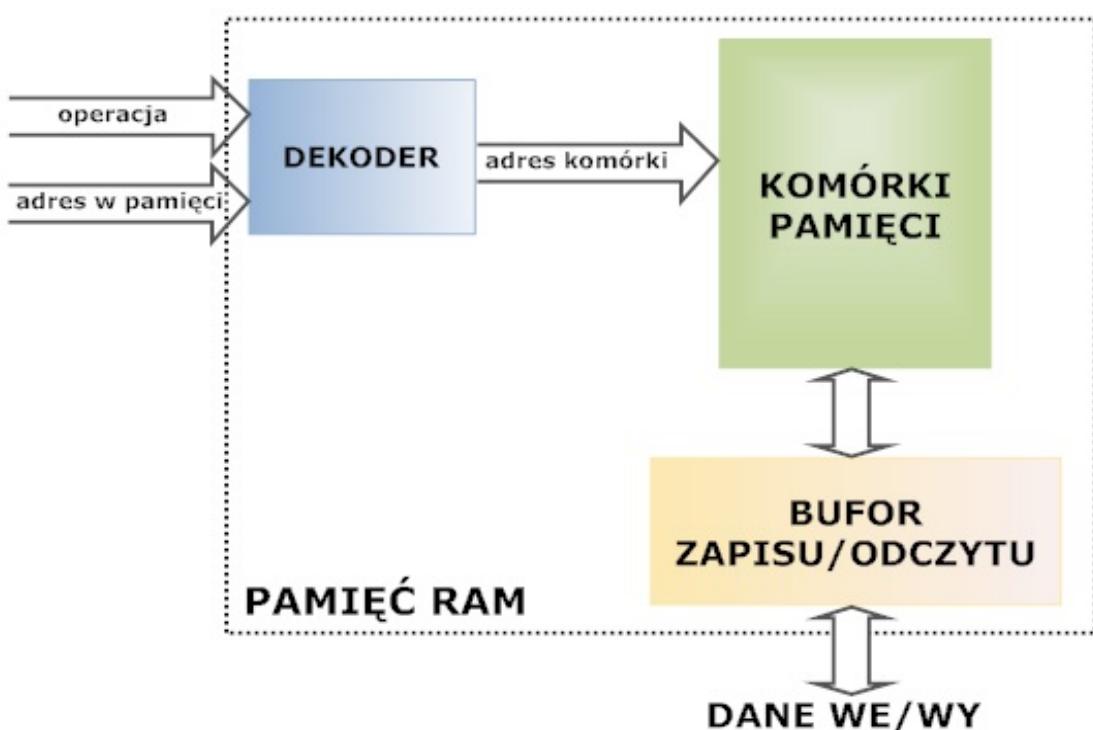
Co to jest?

Pamięć operacyjna jest obok procesora jednym z najważniejszych elementów systemu komputerowego i w równym stopniu wpływa na wydajność komputera. Kości pamięci są szybkimi układami logicznymi potrafiącymi, z punktu widzenia użytkownika, przechowywać dane wszelkiego rodzaju. W rzeczywistości przechowywane są jedynie liczby binarne, a ich interpretacja należy do procesora i układów wejścia - wyjścia. Kolejną cechą pamięci operacyjnej jest przechowywanie danych tylko w trakcie działania komputera. Kości pamięci umieszcza się w specjalnych slotach znajdujących się na płycie głównej - tzw. bankach pamięci.

Z pamięci operacyjnej korzysta procesor umieszczając w niej programy (w tym sam system operacyjny) oraz zapisując i pobierając z niej informacje potrzebne do ich wykonywania.

Pamięć połączona jest z pozostałymi elementami systemu komputerowego za pomocą magistrali systemowej.

Z uwagi na różne sposoby budowy kości pamięci, a dokładniej organizacji pojedynczych komórek z których się składa, adres wykorzystywany przez procesor do określenia konkretnej komórki musi być przetłumaczony na adres fizycznego elementu logicznego będącego częścią układu elektronicznego. Pracę tą wykonuje tzw. dekoder. Oprócz adresu komórki, procesor przekazuje do dekodera również informację o typie operacji, jaką chce na niej wykonać (zapis/odczyt). Jeżeli jako typ operacji został określony odczyt, to po odnalezieniu odpowiedniej komórki pamięci jej wartość jest kopiwana do bufora zapisu/odczytu. W następnym cyklu zegarowym procesor może pobrać pożądaną wartość z bufora. Jeżeli procesor zażądał zapisu informacji do komórki o podanym adresie, to bieżąca wartość bufora zapisu/odczytu jest do niej kopiwana.



Schemat blokowy pokazujący wymianę danych z pamięcią RAM.

PODSTAWOWE CECHY PAMIĘCI

Pamięć operacyjna (nazywana również pamięcią główną) jest pojemną, szybką i wymazywalną pamięcią ulotną o swobodnym dostępie.

Brzmi strasznie ... Na pierwszy rzut oka powyższa definicja niewiele wyjaśnia. Charakteryzuje ona jednak prawie całkowicie pamięć RAM w naszym komputerze. Postaramy się zatem wyjaśnić pojęcia, które się w niej pojawiły.

Pojemność pamięci to jej główny parametr określający ilość informacji, jaką jest w stanie przechować w jednej chwili czasowej. Pojemność pamięci mierzy się w bajtach (bajt to 8 bitów, podstawowych jednostek pamięci, każdy bit może przyjmować wartość zero lub jeden, czyli bajt może przyjmować $2^8=256$ wartości) lub słowach (słowo może być dłuższe niż 8 bitów, to zależy od architektury procesora). Ponieważ dzisiejsze pamięci operacyjne mają pojemności rzędu miliardów bajtów, stosuje się przedrostki Kilo-, Mega- lub Giga-, wszystkie w rozumieniu systemu dwójkowego. A więc 1 KB (Kilo Bajt) to 2^{10} czyli 1024 bajty, podobnie Mega to 1024 Kilo, Giga to 1024 Mega, itd... We współczesnym komputerze osobistym pamięć ma zazwyczaj pojemność od 512 MB (Mega Bajtów - 536870912 bajtów) do 2 GB (Giga Bajtów). Komputery z przeznaczeniem serwerowym mają przeważnie zainstalowane od kilku do kilkudziesięciu GB (Giga Bajtów) pamięci operacyjnej.

Ogólnie rozumiana **szynkość pamięci** operacyjnej określona jest głównie poprzez częstotliwość z jaką procesor i inne urządzenia komputera mogą z niej korzystać oraz przez szerokość szyny danych. Na szybkość składa się co najmniej kilka elementów. Pierwszym z nich jest czas dostępu do komórki pamięci, czyli czas jaki minie od momentu zażądania informacji przez procesor do momentu pojawienia się tej informacji na wyjściu pamięci. Dla dzisiejszych szybkich pamięci RAM czas dostępu to rzad kilku nanosekund. Drugim elementem wpływającym na szybkość pamięci jest czas cyklu, czyli czas, jaki musi upłynąć pomiędzy dwoma żądaniemi dostępu do pamięci. Zwykle jest on nieco dłuższy od czasu dostępu. Częstotliwość pracy dzisiejszych pamięci to kilkaset MHz (milionów operacji na sekundę). Trzecim ważnym elementem jest szybkość transferu pamięci, czyli maksymalna możliwa szybkość, z jaką dane mogą być wprowadzane do jednostki pamięci (zapisywane) lub z niej wyprowadzane (odczytywane). Dla pamięci o dostępie swobodnym jest on równy $1/(czas\ cyklu)$.

Pamięć operacyjna jest **wymazywalna**. Oznacza to, że raz zapisana zawartość komórki pamięci może być swobodnie zmieniana. W przeciwnym wypadku pamięć nie miałaby większego pożytku dla procesora, który przecież w trakcie działania przechowuje tam przetwarzane przez siebie dane, w tym wyniki. Dla danych których nigdy się nie zmienia istnieje specjalny typ pamięci niewymazywalnej - ROM (o którym napiszemy więcej za chwilę). ROM nie służy jednak procesorowi do bieżącej pracy lecz zazwyczaj przechowuje podstawowy program uruchamiający elementy składowe systemu komputerowego przed uruchomieniem systemu operacyjnego.

Cechą pamięci operacyjnej jest także jej **ulotność**. Oznacza to, że po zaniku zasilania wszystkie komórki pamięci tracą swoją zawartość, dane w nich przechowywane po prostu znikają (można to porównać do całkowitej amnezji). Przeciwieństwem jest pamięć nieulotna, która zawsze utrzymuje zawartość - nawet bez zasilania.

Ostatnią cechą pamięci wymienioną w pierwszym akapicie jest **swobodny dostęp**. Oznacza to, że każda komórka pamięci ma unikatowy numer, fizycznie wbudowany i przypisany na stałe jej i tylko jej, a czas dostępu do kolejnych komórek nie zależy od poprzednich operacji wykonywanych na pamięci - jest stały. Dzięki temu każda komórka pamięci jest dostępna bezpośrednio dla procesora w tym samym czasie.

Przeciwieństwem dostępu swobodnego jest dostęp sekwencyjny (stosowany chociażby w przypadku taśm magnetycznych, gdzie w celu uzyskania zawartości pewnej komórki pamięci w połowie taśmy należy sekwencyjnie przejść przez wszystkie komórki poprzedzające tą wybraną). Spójrzmy na świat nie-komputerowy. Przykładowo płyta CD daje nam swobodny dostęp do utworów na niej zapisanych, natomiast kaseta magnetofonowa - dostęp

sekwenncyjny. Szukając t-shirtu w szafie na kupce innych t-shirtów mamy do nich dostęp sekwencyjny (najpierw musimy zdjąć te na górze), lecz jeśli umieścilibyśmy każdą koszulkę w oddzielnej szufladzie - mielibyśmy dostęp swobodny, itp.

Specjalnym rodzajem dostępu swobodnego jest dostęp skojarzeniowy (pamięć asocjacyjna). Cechuje się on możliwością porównania i badania zgodności wybranych bitów wewnętrz słowa, przy czym operacja ta wykonywana jest dla wszystkich słów jednocześnie. Słowo jest wyprowadzane na podstawie nie tylko adresu, ale i części swojej zawartości. Taki sposób dostępu stosuje się w pamięciach podręcznych procesora (cache). Podobnie działa też pamięć człowieka.

PAMIĘĆ STAŁA

Pamięć **ROM** (ang. *Read Only Memory*), jak sama nazwa wskazuje, jest pamięcią tylko do odczytu. Informacja zapisana w takiej pamięci jest trwała i nie może być zmieniona. Zaletą pamięci ROM w pewnych zastosowaniach jest właśnie jej nieulotność, fakt, że przykładowo program znajduje się w niej cały czas, nigdy nie wymaga ładowania i nie grozi mu żaden wirus.

Swoistą "mutacją" pamięci ROM jest pamięć **EPROM** (wymazywalna programowalna pamięć stała), która ma same zalety - można zmieniać dane w niej zapisane, lecz po zaniku zasilania zawartość EPROM nie niknie (takie połączenie RAM i ROM). Przed operacją zapisu wszystkie komórki zostają skasowane poprzez (zazwyczaj) naświetlenie specjalnego układu (wewnętrz obudowy) promieniowaniem ultrafioletowym. Skoro jest taka doskonała, to dlaczego nie jest używana jako jedyna? Po pierwsze - pamięć EPROM jest droższa od pamięci ROM lub RAM, po drugie - można ją zapisać jedynie kilka razy i po trzecie (i najważniejsze) - jest wolniejsza niż RAM (podobnie jak ROM - odczyt z tej pamięci jest wolniejszy).

Bratem bliźniakiem pamięci EPROM jest **EEPROM** (elektryczne wymazywalna programowalna pamięć stała). Pamięć ta może być zapisana bez wymazania poprzedniej zawartości, aktualizowane są tylko bąjty adresowane.

Kolejną powszechnie stosowaną pamięcią stałą jest **pamięć błyśkawiczna** (ang. *flash memory*), która wykorzystuje metodę wymazywania elektrycznego. Cała pamięć może być wymazana w ciągu kilku sekund, ponadto możliwe jest wymazanie zawartości tylko niektórych bloków pamięci a nie całego układu. Na bazie takich właśnie pamięci budowane są tzw. pamięci USB czy Pen-Drive, o których więcej powiemy w dalszej części podręcznika. Jednak mimo postępu w dziedzinie konstrukcji wymazywalnych pamięci stałych, ciągle są one zbyt wolne by zastąpić RAM.

PAMIĘĆ O SWOBODNYM DOSTĘPIE (RAM)

Pamięć o dostępie swobodnym **RAM** (ang. *Random Access Memory*) pozwala w stosunkowo łatwy sposób na odczytywanie/zapisywanie danych z/do pamięci. Zapis oraz odczyt odbywają się za pomocą sygnałów elektrycznych. Ważną cechą pamięci RAM jest jej ulotność. Pamięć RAM potrzebuje źródła zasilania, a w przypadku jego braku dane ulegają skasowaniu. Pamięć RAM można podzielić na dynamiczną i statyczną:

- DRAM (ang. *Dynamic Random Access Memory*) - ten rodzaj pamięci o dostępie swobodnym przechowuje każdy bit danych w oddzielnym kondensatorze. Ze względu na to, że nośnikiem informacji są kondensatory (a dokładniej tranzystory polowe Denarda), które samoistnie rozładowują się - DRAM wymaga okresowego odświeżania polegającego na odczytaniu i zapisaniu tej samej informacji. Ponadto odczyt z pamięci DRAM jest niszczący (informacja jest kasowana w wyniku rozładowania kondensatora przez współpracujący z nim tranzystor), dlatego

też należy powtórnie zapisać odczytane dane tak, aby nie uległy zmianie. Zaletą DRAM jest natomiast niski koszt i możliwość uzyskania wysokiego (jak na tego typu pamięć) stopnia upakowania (dużej ilości bitów pamięci na jednostkę powierzchni). Z tego względu pamięci DRAM są podstawowym typem stosowanym jako pamięć operacyjna.

- SRAM (ang. *Static Random Access Memory*) przechowuje dane tak długo, jak długo włączone jest zasilanie. W odróżnieniu od pamięci typu DRAM nie wymaga okresowego odświeżania. Każdy bit przechowywany jest w pamięci SRAM w układzie zbudowanym z czterech tranzystorów tworzących przerzutnik oraz z dwóch tranzystorów sterujących. Taka struktura umożliwia znacznie szybsze odczytanie bitu niż w pamięci typu DRAM. Pamięci SRAM wykorzystywane są w szybkich pamięciach podręcznych cache wbudowanych w procesor. Tego typu elementy nie wymagają dużych pojemności (gęstość danych w SRAM jest 4 razy mniejsza niż w DRAM), ale prędkość dostępu jest około 10 razy większa od DRAM.

PAMIĘĆ MASOWA

Główną wadą omawianych w poprzednim segmencie pamięci operacyjnych jest ich ulotność oraz stosunkowo niewielka pojemność. Główną zaletą - duża szybkość działania. Jednakże istnieje duża grupa zastosowań, gdzie szybkość działania pamięci jest mniej istotna, natomiast ważne jest trwałe przechowywanie zapisanych danych, czy też znacznie większa pojemność. Często nie bez znaczenia jest również fakt "przywiązania" pamięci operacyjnej tylko do jednego systemu komputerowego, co oznacza brak możliwości przekazania danych w niej zapisanych innej osobie.

To trochę tak, jak z naszą pamięcią. Mózg w sposób bezpośredni przetwarza jedynie to, co ma w swoich szarych komórkach - to one stanowią naszą "pamięć operacyjną". Jednak zdajemy sobie świetnie sprawę, że nie zdołamy zapamiętać wszystkiego oraz że po jakimś czasie większość faktów zapominamy. Dlatego potrzebujemy jakiegoś trwałego nośnika informacji, nośnika wiedzy. Taką rolę dla człowieka pełnią notatki, książki, dokumenty, fotografie - czyli ogólnie biblioteka, miejsce gdzie zawsze możemy umieścić nieprzydatną dla nas w tej chwili wiedzę i później - jak będzie taka potrzeba - w każdej chwili do niej wrócić.

Rolę takiej biblioteki dla komputerów pełnią **pamięci masowe** - nieulotne i przeznaczone do długotrwałego przechowywania dużej ilości danych. W zależności od zjawiska fizycznego wykorzystywanego do zapisu informacji, pamięci masowe dzieli się zazwyczaj na trzy główne kategorie.

PAMIĘCI MAGNETYCZNE

W pamięci tego typu informacja jest przekazywana przy wykorzystaniu zjawisk magnetycznych. Pierwszym urządzeniem wykorzystującym efekt magnetyczny do zapisu był tzw. "drutofon". W 1898 r. duński inżynier Valdemar Poulsen opracował zapamiętywanie ludzkiej mowy na pierwowzorze magnetofonu – dźwięki zapisywano na drucie (strunie fortepianowej) magnesowanym przez elektromagnes, w którym prąd był wzbudzany przez mikrofon telefoniczny przetwarzający dźwięki na sygnał elektryczny. Potem, w trakcie 2 wojny światowej w Niemczech wynaleziono taśmę magnetyczną - cienki celuloidowy pasek pokryty warstwą tlenku żelaza, nawinięty na szpulę.

Rozwój pamięci magnetycznych nastąpił z chwilą pojawienia się komputerów. Szybko zauważono, że idealnie nadaje się ona do zapisu danych binarnych. Logicznemu zeru i jedynce mogą odpowiadać dwa stany różniące się zwrotami namagnesowania.

Pierwsze podejście było związane z pamięciami rdzeniowymi. Tego typu pamięć zawierała tysiące rdzeni magnetycznych o średnicy 2 mm wykonanych z ferrytu. Była ona bardzo szybka (jak na owe czasy) - choć nie pozwalała na zbyt wysoką gęstość zapisu (niewielka pojemność). Magnetyczne pamięci rdzeniowe zostały wyparte przez twarde dyski – o nich za chwilę powiemy więcej.

Równolegle wykorzystywano taśmy magnetyczne ze względu na ich ogromną pojemność. Jednak z powodu ich niskiej szybkości działania oraz sekwencyjnego dostępu do danych, dziś jedynie wykorzystuje się je w systemach do tworzenia kopii zapasowych (tzw. streamerach). Natomiast we wszystkich pozostałych zastosowaniach zostały wyparte albo przez dyski twarde, albo pamięci optyczne.

Dysk magnetyczny ma kształt okrągłej płyty - sztywnego krążka o podłożu z metalu lub szkła, na którym osadzona jest warstwa materiału magnetycznego (kiedyś tlenek żelazowy, z czasem stopy metali, w których główny składnik to kobalt) będąca nośnikiem informacji. Dane są na nim zapisywane i odczytywane za pomocą głowicy, która stanowi w tym wypadku cewka elektryczna. W czasie operacji odczytu/zapisu wiruje dysk, zaś głowica pozostaje nieruchoma.

Podczas zapisu wykorzystywane jest pole magnetyczne wytwarzane przez prąd płynący w cewce. Po

nadejściu żądania zapisu, do głowicy wysyłane są impulsy, które powodują czasowe zmiany kierunku pola magnetycznego przez nią generowanego. Zmiany te powodują analogiczne zmiany kierunku namagnesowania powierzchni dysku, która znajduje się w danej chwili pod głowicą (przypominam, że dysk cały czas wiruje). Odczyt polega na wykorzystaniu zjawiska indukowania przepływu prądu elektrycznego pod wpływem pola magnetycznego powierzchni dysku - tym razem głowica "słucha" zamiast "mówić".

Pierwszym szeroko wykorzystywany typem dysków magnetycznych były elastyczne dyskietki (aktualnie wyparte przez pamięci optyczne lub półprzewodnikowe) o kilku rozmiarach - 8 cali, 5.25 cala oraz 3.5 cala. Dyskietkę zawsze można było wyjąć z napędu, zamienić na inną i zabrać ze sobą - głowica była natomiast na stałe osadzona w napędzie. Jednakże ze względu na ryzyko uszkodzeń mechanicznych oraz obecność zanieczyszczeń, dyskietki nie mogły osiągnąć zbyt wysokiej pojemności.

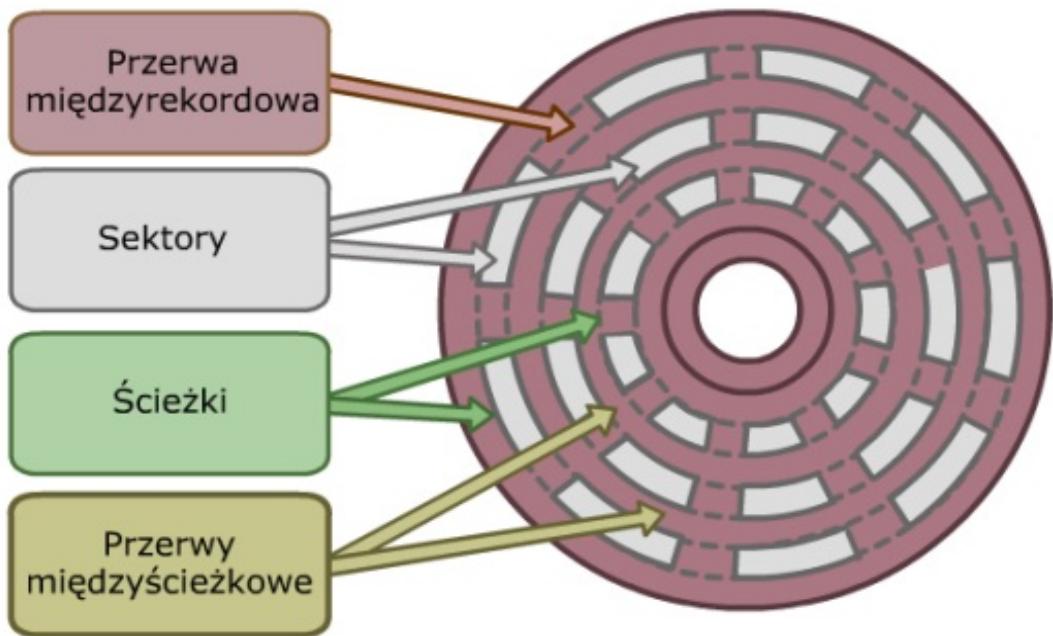
Inne podejście zastosowano w przypadku dysków twardych. Tam jeden bądź więcej talerzy zamknięto wraz z głowicami, silnikami i elektroniką sterującą w jednej, hermetycznej i nieotwieralnej obudowie, co pozwoliło na znaczne zwiększenie gęstości zapisu oraz szybkości działania dysku. Współcześnie prędkość liniowa na dysku przekracza 100 km/h, a odległość unoszącej się nad nim głowicy – 50 milionowych części milimetra. Głowica unosi się nad powierzchnią magnetyczną na poduszce powietrznej generowanej przez obracający się z prędkościami rzędu kilku tysięcy obrotów na minutę dysk. To powoduje, że dyski twarde są dość wrażliwe na wstrząsy w trakcie pracy oraz w zasadzie nierozbieralne (rozhermetyzowanie dysku uniemożliwia jego dalsze wykorzystanie). Jednak ciągle to one mają największą pojemność i szybkość działania wśród wszystkich pamięci masowych oraz stanowią nieodzowny element większości typowych systemów komputerowych.

Dla osób znających przynajmniej podstawy języka angielskiego polecamy przyjemną [animację](#) przygotowaną przez firmę Hitachi - obrazującą między innymi działanie dysku twardego, lecz głównie skoncentrowaną na najnowszych badaniach w tej dziedzinie.

ORGANIZACJA I FORMATOWANIE DANYCH

Dane na dysku twardym zapisywane są w postaci tzw. **ścieżek**. Cała powierzchnia dysku jest podzielona na równo rozłożone pierścienie. Każda ścieżka ma taką samą szerokość jak głowica. Pomiędzy każdym dwoma ścieżkami jest zawsze zostawiony pewien niewielki odstęp, dzięki czemu błędy wynikające z niewłaściwego ustawienia głowicy lub interferencji pola magnetycznego są minimalizowane.

Aby uprościć układy elektroniczne współpracujące z głowicą, na każdej ścieżce przechowywana jest taka sama liczba bitów, a wielkość tą nazywa się **gęstością** i wyraża w bitach na cal.



Organizacja dysku magnetycznego

Później (w [wykładzie](#) poświęconym systemom plikowym) dowiecie się, iż dane na dysku zapisywane są praktycznie zawsze blokami w celu zwiększenia wydajności transferu oraz umożliwienia łatwego wyszukiwania, dodawania i kasowania danych. Takie bloki danych na dysku nazywa się **sektorami**. Na ścieżkę przypadać może wiele sektorów o zmiennej lub stałej długości. Poszczególne sektory wewnętrznych ścieżek na dysku są identyfikowane dzięki dodatkowym danym kontrolnym, umieszczanym pomiędzy nimi - w ten sposób oznacza się początek i koniec każdego sektora względem sektora startowego.

Współczesny dysk twardy jest opisywany kilkoma podstawowymi parametrami. Pierwszym z nich jest pojemność dysku, zazwyczaj podawana w GB oraz jako maksymalna możliwa do uzyskania pojemność, bez uwzględniania struktur kontrolnych. Dlatego też nie należy się dziwić, jeśli dysk o pojemności 200 GB w systemie jest pokazywany jako 180 GB.

Kolejnym, jednym z ważniejszych parametrów, jest szybkość dysku, którą określają dwa czynniki:

1. **Tempo przesyłania** (ang. *transfer rate*) - oznacza szybkość, z jaką dane są transmitowane pomiędzy dyskiem a komputerem.
2. **Czas ustalania położenia głowicy** (ang. *positioning time*, nazywany też ang. *random access time*). Na wielkość tą składają się:
 - *czas wyszukiwania* (ang. *seek time*), czyli czas przesuwania głowicy do odpowiedniego cylindra (cylinder jest to kilka ścieżek umieszczonych fizycznie nad sobą na kilku talarzach dyskowych);
 - *opóźnienie obrotowe*, czyli czas potrzebny na to, aby odpowiedni sektor, przeszedł nad głowicą.

Dyski magnetyczne współpracują zarówno z głowicami nieruchomymi (jedna głowica zapisu/odczytu na jedną ścieżkę) oraz z głowicami ruchomymi - istnieje wtedy tylko jedna głowica zapisu/odczytu na talarz (np. współczesne dyski twarde).

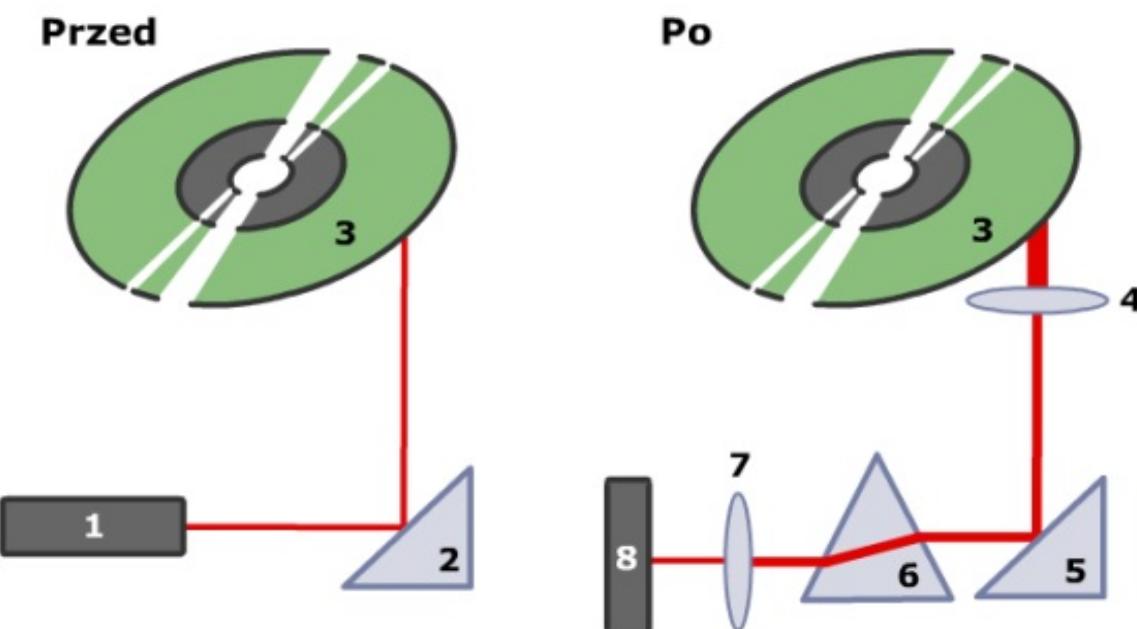
PAMIĘCI Optyczne

Prace nad napędem CD rozpoczęte zostały w roku 1978. Pierwszy napęd powstał dzięki firmie Philips, ale odtwarzacz CD-ROM w dzisiejszej postaci zaprojektowany został przez firmy Philips i Sony. W połowie lat 80 minionego wieku napędy weszły do szerszej dystrybucji, zaś lata 90 przyniosły im niezwykłą popularność głównie dzięki stosunkowo niskiej cenie oraz dużej pojemności jaką oferowały płyty CD-ROM. Przy czym należy tu zaznaczyć, że CD-ROM jest technologią adoptowaną dla potrzeb systemów komputerowych - jej pierwotnym przeznaczeniem było przechowywanie i udostępnianie muzyki.

Płyty CD i CD-ROM są wykonane w ten sam sposób - dysk składa się z czterech warstw: warstwy nośnej z poliwęglanu (plastik), warstwy barwnika (topi się podczas zapisu), warstwy dobrze odbijającej światło (np. aluminium) i lakierowej warstwy ochronnej.

Płyta CD posiada spiralny rowek prowadzący (ang. *groove*), który naprowadza laser, wskazuje mu drogę. Informacja zarejestrowana cyfrowo nanoszona jest w postaci małych zagłębień (ang. *pit*) na powierzchnię odbijającą. Pierwszy egzemplarz wykonuje się to za pomocą dobrze zogniskowanej wiązki laserowej o dużej mocy (podczas zapisu barwnik wytapia się, odsłaniając odbijającą światło powierzchnię). Tak powstaje dysk wzorcowy, który służy jako matryca do tłoczenia kopii. Powierzchnia z naniesioną informacją jest zabezpieczona bezbarwnym lakierem.

Odczyt z płyty odbywa się z wykorzystaniem lasera małej mocy wbudowanego w napęd CD. Laser świeci na wirującą płytę (wprawianą w ruch przez silniczek napędu), pokrytą przezroczystą warstwą lakieru. Zmiana natężenia odbitego światła, zarejestrowana przez komórki światłoczułe, oznacza, że promień napotkał zagłębienie, a więc daną, która następnie zostaje przekonwertowana na sygnał cyfrowy. Dokładniej możecie prześledzić to na rysunku poniżej:



Zasada działania pamięci optycznych - schemat

1. Dioda laserowa (1) emitemy na zwierciadło (2) wiązkę promieni podczerwonych o niskiej energii.
2. Mikrosilnik krokowy (serwonapęd), przesuwając zwierciadło (2) na rozkaz mikroprocesora, ustawia wiązkę światłową we właściwej odległości od środka płyty (na poszukiwanym wycinku ścieżki) (3).

3. Wiązka promieni świetlnych po odbiciu od płyty (3) jest zbierana i skupiana przez soczewkę (4) znajdująca się pod płytą, następnie odbijana od zwierciadła (5) i kierowana do układu pryzmatów (6).
4. Układ pryzmatów (6) kieruje wiązkę promieni świetlnych na następną soczewkę skupiającą (7).
5. Soczewka (7) kieruje wiązkę promieni na fotodetektor (8), który zamienia ją na ciąg impulsów elektrycznych. Następnie impulsy te są dekodowane i przesyłane do komputera.

W zależności od tego, czy dany **pit** na płycie odbija światło czy nie - wiązka promieni będzie zmieniała swoje natężenie na fotodetektorze pomiędzy dwoma stanami: niskim i wysokim. Niski odpowiada wartości logicznej 0, wysoki 1. I oto mamy kolejną metodę na przechowywanie danych binarnych.

Aby umożliwić stałą prędkość odczytu danych przez laser, stosuje się napędy, które obracają krążki:

- **ze stałą prędkością kątową** (ang. *constant angular velocity - CAV*).

Dysk jest podzielony na pewną liczbę sektorów oraz na szereg koncentrycznych ścieżek. Zaletą tego rozwiązania jest łatwe adresowanie z użyciem ścieżki i sektora. Także przesunięcie głowicy pod określony adres wymaga niewielkiego jej ruchu i oczekiwania, aż szukany sektor znajdzie się nad głowicą. Na ścieżkach wewnętrznych i zewnętrznych można umieścić taką samą liczbę danych, dlatego też metoda ta nie znalazła szerszego zastosowania w napędach CD-ROM;

- **ze stałą prędkością liniową** (ang. *constant linear velocity - CLV*).

Dysk obraca się więc wolniej podczas odczytu danych z zewnętrznych sektorów płyty, zaś w pobliżu jej środka "przyspiesza". Taka konstrukcja sprawia, że pojemność ścieżek znajdujących się na obrzeżach dysku znacznie wzrasta.

Dane na dysku CD-ROM, podobnie jak dla dysków magnetycznych, są zorganizowane w bloki. Bloki te składają się z pól:

- *sync* (synchronizacja) - 12 bajtowe pole identyfikujące początek bloku (składa się z bajtu 0, 10 bajtów 1 i bajtu 0);
- *header* (nagłówek) - 4 bajtowe pole zawierające adres bloku i bajt trybu. Tryb 0 to czyste pole danych, tryb 1 - wykorzystanie kodu korekcyjnego dla 2048 bajtów danych, tryb 2 to 2236 bajtów danych bez kodu korekcji;
- *data* (dane) - tu zaczyna się użyteczna dla użytkownika przestrzeń. Standardowo pojemność jednego bloku jest ograniczona do 2048 bajtów;
- *auxiliary* (pomocnicze) - dodatkowe dane użytkownika w trybie 2 lub 288 bajtowy kod korekcyjny w trybie 1.

Pierwotnie napędy czysto optyczne (takie jak opisany powyżej CD) nie umożliwiały użytkownikowi zapisu danych, mogły zawierać jedynie dane wyłoczone na nich raz na zawsze. Później pojawiły się napędy, które umożliwiały jednokrotny zapis czystej płyty. Zasada działania w tym przypadku jest analogiczna jak przy odczytaniu, zmienia się jedynie moc lasera. Podczas odczytu laser pracuje z niewielką mocą, która nie pozwala na uszkodzenie powierzchni nośnika. W trakcie zapisu laser generuje impulsy o dużej mocy - które wypalają pity na płycie. Współcześnie standardem są napędy optyczne typu RW (do odczytu i zapisu), gdzie możliwe jest skasowanie raz zapisanej informacji i powtórne wykorzystanie nośnika. W uproszczeniu efekt ten osiągnięto poprzez wykorzystanie jako warstwy odblaskowej materiałów, które po krótkim ogrzaniu nie odbijają światła, przy nieco dłuższym (do wyższej temperatury) zaczynają odbijać.

PAMIĘCI MAGNETOOPTYCZNE

Połączeniem omówionych wcześniej technik są pamięci magnetooptyczne. W ich przypadku odczyt danych odbywa się przy pomocy światła (lasera), natomiast do zapisu wykorzystywane są efekty magnetyczne. Najczęściej jako warstwę odblaskową w tego typu pamięciach stosuje się materiały, dla których współczynnik odbicia światła zmienia się wraz z kierunkiem namagnesowania.

Zastosowanie takiej pamięci wymaga budowy złożonej głowicy optyczno-magnetycznej, co

spowodowało że napędy magnetoptyczne nie zdobyły szerszej popularności i nie będziemy im poświęcali w tym podręczniku więcej uwagi.

PAMIĘCI PÓŁPRZEWODNIKOWE

W ostatnich latach coraz popularniejsze staje się wykorzystanie jako pamięci masowej pamięci półprzewodnikowych. Najczęściej są to pamięci typu [Flash](#) omówione w poprzednim segmencie, umożliwiające wielokrotny zapis i odczyt poszczególnych komórek. Ich potężną zaletą, w porównaniu z wcześniej omówionymi pamięciami optycznymi i magnetycznymi, jest brak jakichkolwiek mechanicznych ruchomych elementów. Wcześniej mieliśmy do czynienia zawsze z obracającym się z dużą szybkością nośnikiem - co zawsze jest źródłem hałasu, wydzielania się ciepła oraz zużycia znacznych ilości energii przez silnik napędzający. Ponadto nośniki narażone są na mechaniczne uszkodzenia i ogólnie - mniej odporne na zużycie i uszkodzenia.

Pamięci półprzewodnikowe pozbawione są wspomnianych wad. Są znacznie mniejsze, pobierają o rząd wielkości mniej energii i pracują zupełnie bezgłośnie. Problemem w ich przypadku jest ciągle wysoki koszt (w przeliczeniu na MB pojemności znacznie wyższy niż w przypadku pamięci magnetycznych i optycznych) oraz ciągle ograniczona prędkość działania.

Czasem też musicie uważać na nazewnictwo. Współcześnie w komputerach osobistych coraz częściej spotyka się tak zwane dyski SSD (ang. Solid State Drive), które tak naprawdę są pamięcią flash - opakowaną w pudełko dysku i wyposażoną w jego interfejsy. Żadnego fizycznego dysku tam nie ma ... ale ciągle bardzo częstę nazywa się takie rozwiązań po polsku dysk półprzewodnikowy - no cóż, siła przyzwyczajień wygrywa w tym przypadku z logiką.

W języku angielskim nazwy też nie są dobrane idealnie, lecz tam wspólnym mianownikiem nie jest dysk, a napęd. I dlatego mamy, w historycznej kolejności - Floppy Disc Drive, Hard Disc Drive oraz Solid State Drive, przy czym solid state pochodzi od solid state physics i jest potocznie kojarzony ze stosowaniem tranzystorów i układów scalonych.

TRYB DOSTĘPU DO DANYCH

Jedną z wymaganych cech współczesnych pamięci masowych jest bezpośredni dostęp do danych na nich umieszczonych. Oznacza to, że zapis i odczyt realizowany jest przez ten sam mechanizm (głowice, laser) oraz dane są zapisywane w blokach. Poszczególne bloki mają unikatowy adres oparty na lokalizacji fizycznej, a dostęp odbywa się w dwóch etapach. Najpierw następuje pozycjonowanie głowicy w najbliższym otoczeniu danych (na odpowiedniej ścieżce lub jej fragmencie). W drugim etapie następuje sekwencyjne poszukiwanie, liczenie lub oczekiwanie w celu osiągnięcia szukanej lokalizacji (obrót talerza lub płyty do odpowiedniej pozycji). Dlatego całkowity czas dostępu do informacji jest zmienny i zależy od aktualnego stanu urządzenia.

Z tego powodu nie można traktować pamięci masowych (z wyłączeniem półprzewodnikowych) jako pamięci o dostępie swobodnym (jak np. RAM) - postulat o jednakowym czasie dostępu do każdej komórki w tym wypadku nie jest spełniony. W zamian za to nazywamy je pamięcią o dostępie bezpośrednim, w odróżnieniu od np. taśmy magnetycznej, gdzie możliwy był jedynie dostęp sekwencyjny.

PRZERWANIA

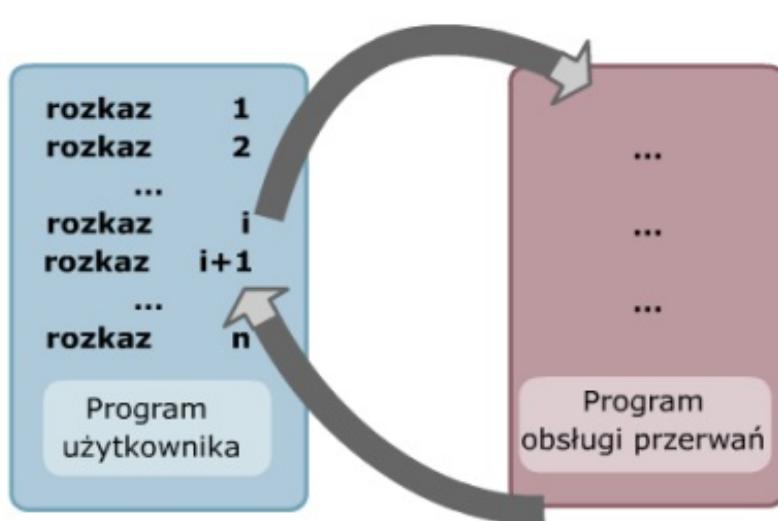
Dla wszystkich, którzy zetknęli się ze współczesnymi systemami komputerowymi, zupełnie normalną cechą jest możliwość wykonywania wielu zadań w jednym czasie. Tak naturalną, że zupełnie nie zastanawiamy się jak to jest możliwe ... a pomyślmy przez chwilę...

Rozważmy bardzo prosty przykład - zlecamy naszemu komputerowi obliczenia, które trwają dajmy na to 10 minut. W międzyczasie ruszamy myszką i ... i wskaźnik na ekranie poruszył się! Biorąc pod uwagę fakt, że o zawartości ekranu decyduje procesor i że ciągle znakomita większość wykorzystywanych procesorów to procesory jednordzeniowe - czyli mogą wykonywać **tylko jeden** program w danej chwili, to jesteśmy świadkami cudu ...

Oczywiście, nie był (i nie jest) to żaden cud. Istnieje kilka mechanizmów pozwalających na przerwanie procesorowi wykonywania aktualnego zadania i zlecenie mu natychmiastowego wykonania małej roboty na boku. Głównym i najważniejszym z tych mechanizmów są **przerwania**. Sama nazwa też nie jest przypadkowa - sygnalizuje, że przerwania mają w zwyczaju przerywać normalną pracę programów.

Dokładnie tak, jak dzieje się w przypadku człowieka, gdy podczas pracy zadzwoni telefon. Dajmy na to, że piszemy ręcznie list do przyjaciela/przyjaciółki. W pewnym momencie słysząc dzwonek telefonu - odkładamy więc pióro, podnosimy słuchawkę i zajmujemy się przez chwilę rozmową telefoniczną. Gdy rozmowa się skończy, bierzemy do ręki pióro i wracamy do wcześniej wykonywanej pracy kontynuując zdanie które przerwaliśmy.

Jak w ogólnym zarysie działają przerwania? Wyobraźmy sobie, że pewne urządzenie zewnętrzne jest gotowe do obsługi (czyli np. przyjmowania danych od procesora, czy też - z wprowadzającego przykładu - zmieniło się położenie myszki i chce ona poinformować o tym procesor). Wtedy moduł wejścia/wyjścia (ang. *I/O, Input/Output*) tego urządzenia wysyła sygnał **żądania przerwania** do procesora (*dzwoni dzwonek telefonu*). W odpowiedzi procesor zawiesza działanie bieżącego programu (*odkładamy pióro*) i wykonuje skok do **programu obsługi przerwania** (ang. *interrupt handler*) obsługującego to urządzenie (*rozmawiamy przez telefon*). Po obsłużeniu urządzenia następuje powrót do programu, który został przerwany (*wracamy do pisania*). Żeby opisany wyżej schemat działania był możliwy, potrzebna jest jeszcze współpraca z systemem operacyjnym - współcześnie to on odpowiada za obsługę wszystkich przerwań. Więcej o tym przeczytacie w trakcie omawiania systemów operacyjnych.



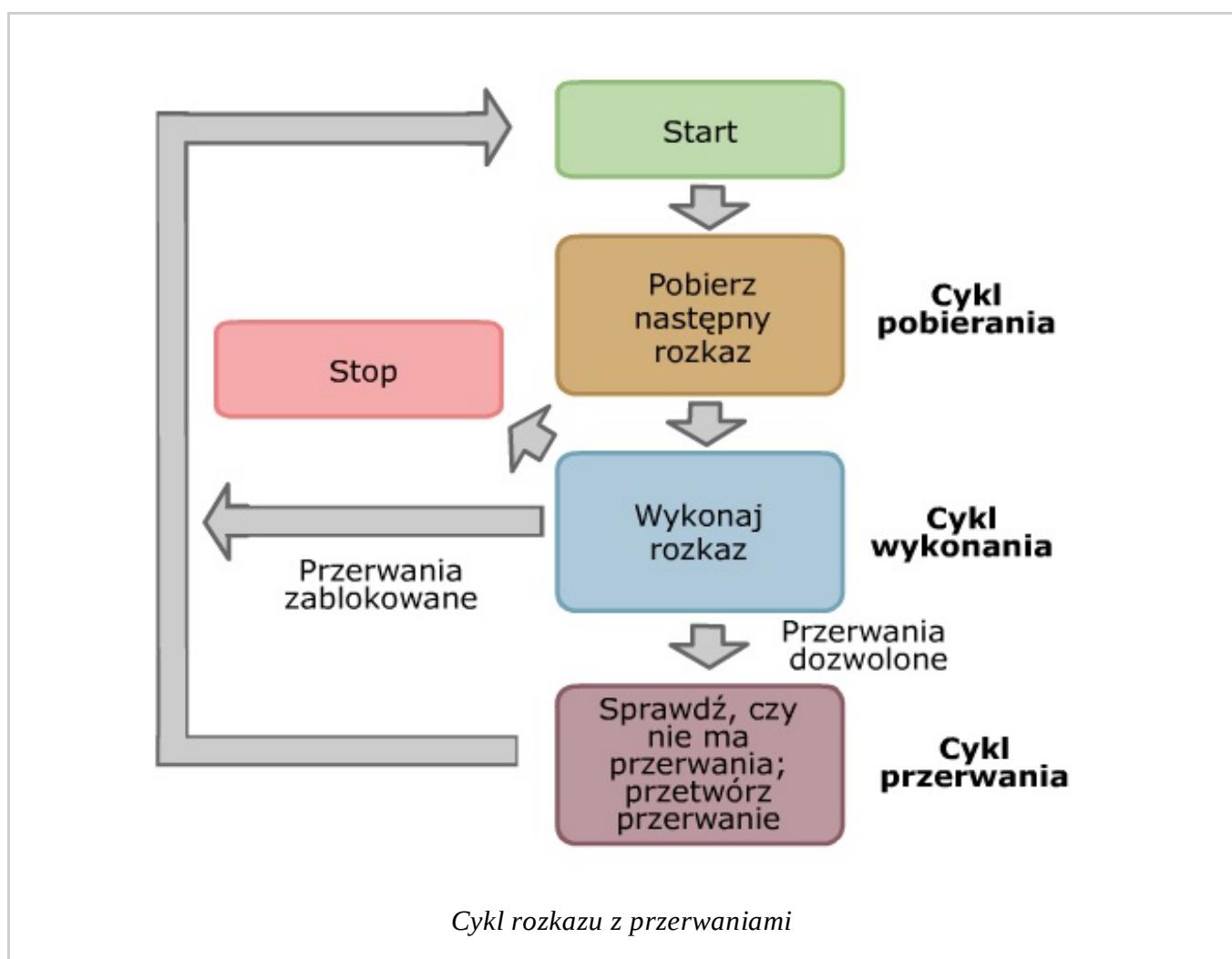
Przekazywanie sterowania przez przerwania

Ponieważ przerwania powinny być obsługiwane szybko, system posługuje się tablicą wskaźników do procedur obsługujących przerwania. Owa tabela nazywana jest **wektorem przerwań** (ang. *interrupt vector*) i jest indeksowana jednoznacznym numerem urządzenia.

Występowanie przerwań modyfikuje nieco schemat cyklu rozkazowego. Do cyklu rozkazu dodawany jest **cykl przerwania**. W trakcie cyklu przerwania procesor sprawdza, czy nie pojawiły się jakieś przerwania. Jeśli przerwania nie są wykonywane, procesor pobiera następny rozkaz danego programu. Jeśli natomiast wystąpi zgłoszenie przerwania procesor:

1. Zawiesza wykonywanie bieżącego programu i zachowuje jego kontekst, czyli przesyła do specjalnie wydzielonego obszaru pamięci zawartość rejestrów oraz adres następnego rozkazu (zawartość licznika rozkazów). Ten wydzielony obszar pamięci nazywa się **stosem systemowym**.
2. Licznik rozkazów jest ustawiany na początkowy adres programu obsługi przerwań.

Następnie procesor pobiera pierwszy rozkaz z programu obsługi przerwań i zaczyna go wykonywać.



Warto wspomnieć, że zaawansowane architektury przerwań pozwalają na obsługę nowego przerwania przed zakończeniem poprzedniego. System operacyjny korzysta ze specjalnego schematu priorytetów, który określa jak ważna jest obsługa danego urządzenia. Przerwanie o wyższym priorytecie będzie obsługiwane również wtedy, gdy jest w trakcie wykonywane przerwanie o priorytecie niższym, zaś przerwania o tym samym lub niższym priorytecie będą *maskowane* (ang. *masked interrupt*).

OPERACJE WEJŚCIA / WYJŚCIA

Mówiliśmy już o mózgu komputera (procesorze), następnie o pamięci, przyszła zatem pora na odpowiednik zmysłów - czyli układy wejścia/wyjścia, przy pomocy których procesor komunikuje się ze światem zewnętrznym. Nie będziemy teraz omawiali szczegółowo zasady działania takich układów, ograniczymy się do omówienia sposobu przesyłania danych z urządzeń wejścia / wyjścia do komputera.

Na początek zajmiemy się fizycznymi aspektami transmisji sygnałów, a następnie omówimy tryby pracy procesora i pamięci w trakcie obsługi sygnałów zewnętrznych. Zanim jednak zaczniemy - ważna uwaga:

wszystkie sygnały przetwarzane przez procesor i wszystkie wartości, które są przechowywane w pamięci, muszą mieć postać cyfrową.

Nasz świat nie jest cyfrowy - jest analogowy. Temperatura powietrza jest wartością, która zmienia się w sposób ciągły, co oznacza że pomiędzy dwie dowolne, ale różne wartości zawsze da się wstawić jeszcze jedną, np.: pomiędzy 20 a 21 stopni jest 20,5, pomiędzy 20 a 20,5 jest np. 20.1, pomiędzy 20 a 20,1 jest 20,09 i tak dalej - w nieskończoność.

Komputer nie rozumie wartości analogowych. Dla niego istnieją tylko wartości zdyskretyzowane. Na czym to polega? Umówmy się przykładowo, że 0 stopni Celsjusza przypiszemy wartość 0, 100 stopniom Celsjusza przypiszemy wartość 1000. Teraz 20 stopni to liczba 200, 20.5 stopnia to 205, ale już zarówno 20.1 jak i 20.09 to ta sama liczba - 201. Komputer stracił zdolność rozróżniania tych liczb. W większości przypadków to nic nie szkodzi, przecież my też nie odróżniamy 20 stopni od 20.1 - i jakoś żyjemy ;) Tyle tylko, że nasze zmysły działają w sposób ciągły i w ten sposób analizują przychodzące informacje, natomiast w przypadku komputerów najpierw wartość analogowa (ciągła) musi zostać zamieniona na cyfrową (dyskretną). Takiego przetworzenia dokonują przetworniki AC. Wynik pracy komputera, jeśli ma zostać zaprezentowany w postaci analogowej (np. muzyka którą generuje karta dźwiękowa), także musi zostać poddany przekształceniu - tym razem odwrotnemu, z sygnału cyfrowego na analogowy. Do tego służą przetworniki CA.

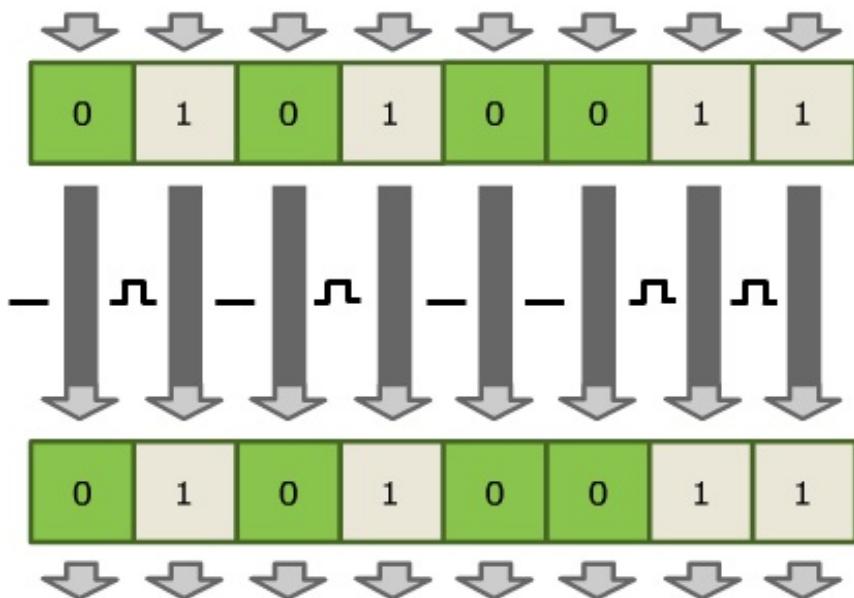
Przetworniki są urządzeniami elektronicznymi i nie będziemy się tutaj zagłębiać w ich budowę. Natomiast z powyższych rozważań jeden fakt jest dla nas istotny - wewnątrz komputera wszystkie przesyłane wartości mają postać cyfrową (zestaw bajtów, z których każdy składa się z 8 zer lub jedynki). I jak taką liczbę przesyła się pomiędzy procesorem, pamięcią i urządzeniami zewnętrznymi?

RODZAJE TRANSMISJI

Najbardziej podstawowym z podziałów jest podział transmisji sygnału na równoległą i szeregową.

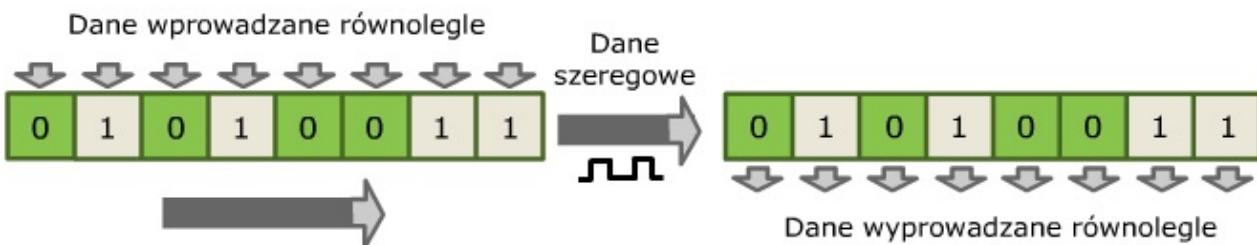
Transmisja równoległa polega na przesyłaniu wszystkich pól słowa danych jednocześnie, czyli przykładowo; całego bajtu, dwóch bajtów, 64 bitów, itp.. w zależności od standardu interfejsu komunikacyjnego. Ze względu na fakt, iż sprzęt komputerowy w naturalny sposób wspiera taki sposób wprowadzania/wyprowadzania danych (procesor operuje na całym rejestrze a nie na poszczególnych bitach w nim umieszczonych), transmisja równoległa jest bardzo popularna. Ponadto, dzięki zrównolegleniu przesyłu danych, uzyskujemy znaczny przyrost prędkości przesyłu. Udogodnienia te wiążą się jednak z koniecznością stosowania dużej liczby fizycznych kanałów transmisyjnych (np. przewodów). Równoległy interfejs Centronics używany kiedyś do podłączania drukarek wymagał 18 przewodów, z kolei taśmy do podłączania dysków twardych zgodne z równoległym EIDE mają 80 żył.

Dane wprowadzane równolegle



Transmisja równoległa

Transmisja szeregową polega na sekwencyjnym przesyłaniu danych bit po bicie. W tym celu, przed przesłaniem informacji jest wprowadzana do tzw. **rejestru przesuwnego** styku szeregowego, który następnie bit po bicie wysyła ją na wyjście układu.



Transmisja szeregowa

Zaletą tego rodzaju transmisji jest możliwość przesyłu danych na dużo większą odległość niż w przypadku transmisji równoległej, a przy tym konieczne są jedynie dwa przewody do transmisji (jednostronnej) sygnału. Niegdyś główną wadą tej techniki była niższa szybkość transmisji oraz bardziej skomplikowana budowa wewnętrzna interfejsu (konieczność istnienia rejestru przesuwnego). Lecz zalety tego typu przesyłania danych przeszliśmy wady i dziś, po zwiększeniu szybkości łącz szeregowych głównie poprzez wzrost częstotliwości pracy interfejsów, transmisja równoległa jest w zasadzie wykorzystywana jedynie we wnętrzu komputera, natomiast większość wyprowadzeń interfejsów na zewnątrz to łącza szeregowe (USB, FireWire).

TRYBY PRACY PODCZAS TRANSMISJI

Fizyczna metoda transmisji to jedno, a sposób w jaki jest ona sterowana, w jaki sposób określa się które dane i kiedy mają zostać przesłane, i kto nadzoruje transmisję - to zupełnie inne, choć równie istotne zagadnienie. Istnieją trzy podstawowe sposoby realizacji operacji wejścia-wyjścia (ang. *input-output* - IO):

- **programowane wejście-wyjście** - komunikacja odbywa się pomiędzy procesorem a modułem I/O. Procesor za pośrednictwem specjalnego programu ma możliwość sterowania operacją wejścia-wyjścia, wysyłania rozkazu zapisu/odczytu i transferu danych. Wadą tego rozwiązania jest fakt, że procesor musi czekać na zakończenie operacji wejścia-wyjścia przez moduł I/O. Jeśli weźmiemy pod uwagę zdecydowanie większą szybkość współczesnych procesorów niż prędkość osiągana przez urządzenia zewnętrzne, to nie ma co się dziwić, że tego typu podejście nie jest już w praktyce stosowane;
- **wejście-wyjście sterowane przerwaniami** - procesor wydaje rozkaz I/O, później zajmuje się wykonywaniem innych rozkazów, gdy moduł I/O zakończy wykonywanie swojego zadania przerywa czynność procesora.

W obu powyższych rozwiązańach procesor odpowiada za odczyt danych z pamięci głównej i za zapis tych danych;

- **bezpośredni dostęp do pamięci** (ang. *direct memory access - DMA*) - tryb ten charakteryzuje się tym, że moduł I/O i pamięć główna wymieniają dane bez udziału procesora.

PROGRAMOWE WEJŚCIE/WYJŚCIE

Urządzenia zewnętrzne są zwykle połączone z systemem poprzez moduły wejścia-wyjścia. Oczywiście każde urządzenie posiada unikatowy identyfikator i adres, który wystawia procesor chcąc odwołać się do urządzenia. Jeżeli procesor, pamięć główna i moduły wejścia-wyjścia używają wspólnej magistrali, możliwe są dwa tryby adresowania:

1. **Odwzorowany w pamięci** - ta sama przestrzeń adresowa przeznaczona jest dla komórek pamięci i urządzeń I/O. Procesor wykonuje takie same instrukcje maszynowe niezależnie od tego, czy odwołuje się do pamięci czy do urządzeń I/O. Tak więc, na magistrali potrzebna jest tylko jedna linia zapisu i jedna linia odczytu.
2. **Izolowane wejście-wyjście** - przestrzeń adresowa jest podzielona: inną pulę adresów mają urządzenia I/O, a inną komórki pamięci. Tak więc, część linii magistrali wykorzystywana jest do odczytu/zapisu komórek pamięci, a część do adresowania modułów I/O, dzięki czemu możliwe jest wykorzystanie całej przestrzeni adresowej zarówno przez pamięć jak i urządzenia I/O.

WEJŚCIE / WYJŚCIE STEROWANE PRZERWANIAMI

Wadą programowanego wejścia-wyjścia był długi czas oczekiwania przez procesor na gotowość wybranego modułu I/O. Dlatego też wykorzystuje się wejście-wyjście sterowane przerwaniami, którego działanie można przedstawić następująco:

1. Procesor wydaje modułowi rozkaz wejścia-wyjścia, po czym przechodzi do wykonywania "ciekawszych" zadań (np. do obliczania całki Fouriera ...).
2. Kiedy moduł I/O jest gotowy do wymiany danych z procesorem (czyli np. wczytał już dane z urządzenia peryferyjnego), wysyła żądanie obsługi przerwania i czeka aż procesor będzie gotowy do wymiany danych.
3. Gdy następuje przerwanie od modułu I/O procesor zachowuje kontekst bieżącego programu (np. obliczania całki Fouriera) czyli minimalną zawartość licznika programu i rejestrów procesora, a następnie zgłasza chęć transmisji.
4. Moduł I/O umieszcza dane na magistrali, po czym jest gotowy do kolejnej operacji wejścia-wyjścia.

wyjścia.

5. Procesor transmituje dane, a następnie odnawia kontekst programu (np. całki Fouriera) i wznowia jego działanie.

Wejście-wyjście sterowane przerwaniami jest wydajniejsze niż programowane, jednakże i w tym rozwiązaniu traci się dużo czasu procesora, który musi nadzorować każde słowo transmisji pomiędzy pamięcią a modułami I/O. Każde słowo - co oznacza że procesor 32 bitowy ma "przerwę" jedynie na czas potrzebny na przygotowanie przez urządzenie zewnętrzne 4 bajtów ... co nie trwa długo. I co każde 4 bajty urządzenie przesyłające np. zdjęcie z aparatu cyfrowego "dzwoni" do procesora i chce pogadać ... przecież tak nie da się wydajnie pracować! ;) Dlatego też i ten tryb pracy nie jest współcześnie zbyt chętnie stosowany.

DMA

Metoda **bezpośredniego dostępu do pamięci DMA** (ang. *Direct Memory Access*) stosowana jest zawsze tam, gdzie muszą być przenoszone duże ilości danych przy minimalnym obciążeniu procesora. To "minimalne" obciążenie w praktyce oznacza zupełne zwolnienie procesora z nadzorowania przesyłania danych między pamięcią a urządzeniem zewnętrznym.

Metoda ta wymaga zainstalowania dodatkowego modułu DMA na magistrali systemowej. Moduł ten potrafi przejmować od procesora sterowanie magistralą systemową. Transfer ciągły inicjuje procesor, ale w tym wypadku jest to raczej polecenie szefa do dobrze wykształconego zespołu specjalistów. Gdy procesor chce odczytać/zapisać dane wydaje rozkaz modułowi DMA, wysyłając następujące informacje:

- czy potrzebny jest odczyt, czy zapis?
- adres urządzenia I/O wymaganego do komunikacji;
- adres początkowej komórki pamięci, do której nastąpi zapis/odczyt;
- liczbę słów, które należy odczytać/zapisać.

Procesor powraca do wykonywania swoich zadań (np. programu użytkowego), zaś moduł DMA zajmuje się transmisją danych pomiędzy modułem I/O a pamięcią z pominięciem procesora. Gdy moduł DMA zakończy przesył danych, wysyła sygnał przerwania do procesora. Procesor jest więc używany tylko na początku i końcu transmisji - niezależnie od ilości przesyłanych danych.

Jak już wspomnieliśmy moduł DMA przejmuje działanie procesora w komunikacji pomiędzy pamięcią a modułami I/O. Dlatego też konieczny jest dostęp do magistrali. I tu pojawiają się dwie opcje:

1. Moduł DMA używa magistrali tylko wówczas, gdy nie potrzebuje jej procesor.
2. Moduł DMA wymusza czasowe zawieszenie operacji procesora.

Ta ostatnia metoda jest bardziej popularna i określa się ją mianem **wykradania cyklu** ze względu na fakt, że moduł DMA zajmuje cykl magistrali.

Wspomniane "wykradanie cyklu" nie jest równoważne przerwaniu - procesor zatrzymuje się na jeden cykl magistrali bez zachowywania kontekstu. Oczywiście "wykradanie cyklu" sprawia, że procesor pracuje nieco wolniej, ale dla dużych bloków danych metoda DMA jest daleko bardziej efektywna niż opisane wcześniej *programowane wejście-wyjście* i *wejście-wyjście sterowane przerwaniami*.

MECHANIZMY OCHRONY WEJŚCIA/WYJŚCIA, PAMIĘCI, JEDNOSTKI CENTRALNEJ

OCHRONA WEJŚCIA/WYJŚCIA

Podobnie jak w przypadku omawiania przerwań, również w tym segmencie zajmiemy się problemem, którego rozwiążanie wymaga współdziałania elementów sprzętowych i systemu operacyjnego. Przerywania są nam potrzebne, by można było chwilowo wstrzymać wykonywanie jednego programu i przesyłać dane z urządzeń wejścia/wyjścia lub przekazać sterowanie tymczasowo do innego programu. To pozwala na stosowanie we współczesnych systemach operacyjnych wielozadaniowości - mamy jeden procesor, który jednocześnie wykonuje wiele programów (w rzeczywistości w danej chwili wykonuje zawsze jeden, ale bardzo szybko przełącza się między nimi). Jednak, aby takie działanie było możliwe na szeroką skalę, samo przydzielanie czasu procesora nie wystarczy. Trzeba jeszcze zadbać, by jeden niedbale napisany program nie uszkodził pozostałych, pracujących z nim równolegle.

Działania systemu mogą zostać zakłócone poprzez nieautoryzowany dostęp do pamięci: program A korzysta z pamięci programu B zamazując jego dane. Mogą być również zakłócone poprzez jednoczesny dostęp do tego samego urządzenia wejścia/wyjścia - zdają sobie dobrze z tego sprawę osoby, które pisały programy w asemblerze ... pozostali niech na razie uwierzą nam na słowo ;)

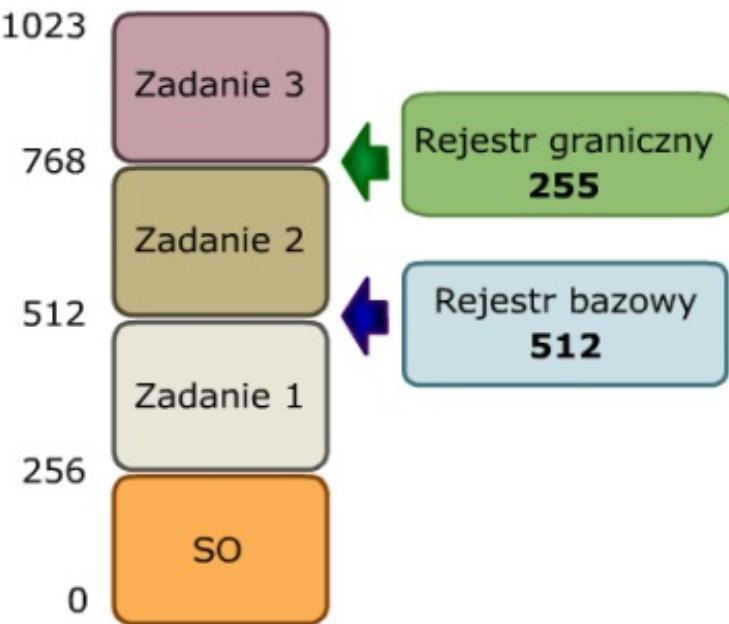
Niedozwolone operacje I/O prowadzą do nieokreślonych zachowań systemu komputerowego. Dlatego też, aby ustrzec się przed wykonaniem niedozwolonej operacji wejścia-wyjścia przyjęto, że wszystkie rozkazy I/O są uprzywilejowane, zaś użytkownicy mogą z nich korzystać jedynie za pośrednictwem systemu operacyjnego. Uzyskanie pełnej ochrony wejścia-wyjścia wymaga uniemożliwienia przejęcia przez program użytkownika kontroli nad systemem komputerowym. Więcej o tym powiemy przy okazji omawiania budowy systemu operacyjnego.

OCHRONA PAMIĘCI

Podstawowymi jednostkami pamięci, które muszą być chronione, są wektor przerwań oraz systemowe procedury obsługi przerwań. Zaniedbanie tej ochrony mogłoby doprowadzić do sytuacji, w której program użytkownika zastąpiłby systemowe procedury obsługi przerwań skokami do własnego obszaru, przechwytując w ten sposób sterowanie od procedury obsługi przerwania. To doprowadzić mogłoby do nieprawidłowego działania systemu komputerowego ... co zresztą skwapliwie wykorzystywały wczesne wersje wirusów komputerowych.

Bezpieczeństwo wektora przerwań i systemowych procedur obsługi przerwań jest istotne. Ale co zrobić, gdy chcemy chronić całą pamięć systemu operacyjnego przed wpływem "z zewnątrz" oraz obszary pamięci kilku różnych programów użytkowych? Otóż z pomocą przychodzą rozwiązania sprzętowe.

Najpopularniejsza metoda ochrony pamięci programu opiera się na wykorzystaniu dwóch rejestrów: bazowego (ang. *base*) i granicznego (ang. *limit*). Rejestr bazowy przechowuje podstawowy, najmniejszy, dopuszczalny, fizyczny adres pamięci przydzielony danemu programowi, zaś rejestr graniczny zawiera jej rozmiar.



Rejestr bazowy i graniczny

Dzięki ww. rejestrów można zdefiniować zakres adresów (obszar czy też fragment pamięci), przyznawany niezależnie i w sposób nie pokrywający się - dla każdego z uruchomionych programów, oraz uzyskujemy możliwość ochrony pamięci poza tymi adresami - to czyni każdy sensowny system operacyjny.

Ochrona w skrócie polega na tym, iż procesor zanim zapisze / odczyta coś z pamięci, sprawdza dla każdego adresu, dla którego zleci takie działanie system operacyjny, jaka jest zawartość opisanych powyżej rejestrów. Jakakolwiek ingerencja wykonana przez program do pamięci innego programu lub innego, nie przydzielonego obszaru RAM, kończy się wygenerowaniem przez system błędu, który w przypadku Windows zazwyczaj nazywa się *Access Violation Error* (chyba najbardziej nielubiany błąd spotykany przy programowaniu, szczególnie w językach niskiego poziomu lub C/C++).

OCHRONA JEDNOSTKI CENTRALNEJ

System operacyjny powinien sprawować stałą kontrolę nad komputerem, dlatego też należy zapobiec sytuacji, w której program użytkownika wpadłby w nieskończoną pętlę permanentnie zajmując procesor. Wykorzystuje się do tego **czasomierz** (ang. *timer*). Timer pozwala na generowanie przerwania po wyznaczonym czasie - czas ten może być stały lub zmienny (przyrostowo o pewien krok).

Tak więc system operacyjny, przed oddaniem sterowania do programu użytkownika, ustawia timer na przerwanie. Gdy czasomierz wywołuje przerwanie, sterowanie wraca do systemu operacyjnego, który decyduje: czy wystąpił błąd, czy też należy dać programowi użytkownika więcej czasu. W ten łatwy sposób można zapobiec zbyt długiemu działaniu programów lub wymusić zakończenie działania programu po upływie określonego czasu.

Czasomierz wykorzystywany jest również do podziału czasu procesora. Umożliwia on przełączanie się programów użytkowych co pewien określony czas (stały lub zmienny), stwarzając w ten sposób wrażenie pracy równoległej.

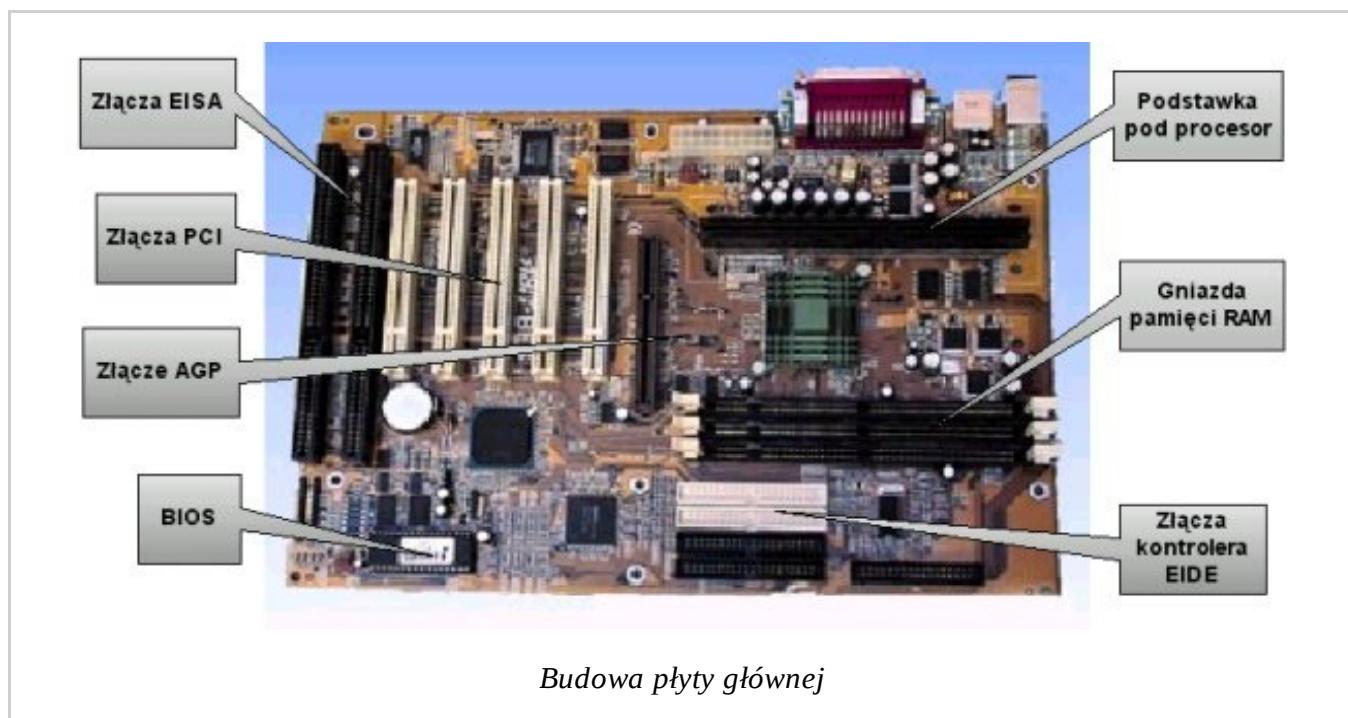
Omówione w tym segmencie mechanizmy koncentrują się głównie na sprzętowych aspektach bezpiecznej współpracy wielu programów oraz systemu operacyjnego. Na razie jeszcze nie skupialiśmy się na tym, *jak* przydzielać pamięć dla wielu programów, *ile* czasu przeznaczyć dla każdego z nich, *kiedy* przyznać dostęp do urządzeń zewnętrznych a kiedy go odmówić. Postaramy się odrobinę wprowadzić Was w te zagadnienia w lekcjach 5 i 6, lecz to także nie wyczerpie tematu. Część z Was będzie miała okazję później, w dalszym toku studiów, poznać wspomniane zagadnienia dokładniej, na przedmiocie **Systemy operacyjne**.

BUDOWA WSPÓŁCZESNEGO KOMPUTERA OSOBISTEGO

PŁYTA GŁÓWNA

Omawianie współczesnych podzespołów komputera rozpoczniemy od jego szkieletu - czyli płyty głównej.

Płyta główna jest płytka z obwodami drukowanymi, na której montowana jest znaczna część komponentów niezbędnych do poprawnego funkcjonowania systemu komputerowego. Na płycie głównej umieszczona jest podstawa umożliwiająca montaż procesora, tutaj znajdują się również magistrale systemowe, pamięć ROM (służąca do przechowywania głównego programu obsługi komputera - BIOSU, a także programów testujących podzespoły komputera przy starcie systemu - POST), gniazda (ang *slot*) na karty rozszerzeń, gniazda pamięci RAM, kontroler urządzeń I/O, kontroler DMA, wszelkiego rodzaju wyprowadzenia interfejsów służących do podłączania dysków i urządzeń peryferyjnych, i - *last but not least* - chipset płyty głównej.



CHIPSET

Chipset stanowi "serce" płyty głównej i odpowiada za sterowanie przepływem strumienia danych (sterowanie magistralą systemową). Jest to taka "szara eminencja". Niby rządzi procesor, ale bez zgody chipsetu, żadna jego decyzja nie zostanie dostarczona do urządzeń zewnętrznych. Chipset zwykle jest podzielony logicznie na dwa osobne układy, tzw. mostki.

Mostek południowy (ang. *south bridge*) umożliwia dołączenie do procesora portów I/O (tj. interfejsy szeregowe/równoległe, magistrala USB), zapewnia możliwość korzystania z magistrali ISA oraz pozwala podłączyć urządzenia do złącz IDE. Ponadto mostek południowy steruje funkcjami

zarządzania energii oraz monitoruje parametry systemu.

Mostek północny (ang. *north bridge*), który steruje przepływem danych (jest kontrolerem FSB - głównej szyny procesora) pomiędzy procesorem, pamięcią operacyjną i podręczną, złączem AGP i PCI, a także mostkiem południowym. Mostek północny zapewnia bezkonfliktową współpracę magistrali pracujących nierazdrożko z różnymi częstotliwościami taktowania.

Pojęcie chipsetu ściśle wiąże się z typem obsługiwanyego przez płytę główną procesora - innego chipsetu wymaga procesor Intel Core a innego AMD. Rodzaj chipsetu zależy również od specyfikacji elektrycznej i mechanicznej wyprowadzeń gniazda procesora (a więc typu gniazda). Dlatego przy budowaniu własnego zestawu komputerowego w pierwszej kolejności powinniśmy wybrać procesor, w oparciu o który będziemy składać resztę podzespołów. Typ gniazda procesora obecny na płycie głównej musi być zgodny z konkretnym modelem układu. Do najpopularniejszych zaliczamy:

Procesory Intel:

- LGA 775 – układy Pentium 4, Celeron, Core 2 Duo
- LGA 1156 – układy Intel Core i3, i5, i7 oraz Core 2 Duo
- LGA 1155 – procesory oparte o architektury Sandy Bridge i Ivy Bridge
- LGA 2011 – procesory oparte o architektury Sandy Bridge-E i Ivy Bridge-E
- LGA 1150 – procesory oparte o architektury Haswell i Broadwell

Procesory AMD:

- Socket AM2+ – układy Phenom X4, Phenom II, wsparcie dual-channel DDR3-SDRAM i HyperTransport 3
- Socket AM3 – układy Phenom X6, Athlon X4, wsparcie dual-channel DDR3-SDRAM i HyperTransport 3
- Socket AM3+ – procesory oparte o architekturę Bulldozer
- Socket FM2 – procesory oparte o architekturę Trinity

Sam chipset może być wyprodukowany przez kilka firm, najbardziej znane to Intel, AMD, NVidia czy SIS. Dokładne omówienie istniejących chisetów to temat - rzeka, przy czym, podobnie jak w przypadku procesorów - rozwiązań w tej dziedzinie zmieniają się wyjątkowo szybko. Zamiast więc zaprzątać Wam głowę szczegółami, chcielibyśmy, byście pamiętały o znaczeniu płyty głównej. To od jakości płyty właśnie w dużym stopniu zależy stabilność pracy komputera oraz jego możliwości rozbudowy. Dlatego też nie polecamy stosowania rozwiązań najtańszych (aktualnie są to płyty kosztujące ok. 100-300 PLN). Jeśli macie wpływ na wybór płyty głównej do komputera - miejcie świadomość, że większość z tych najtańszych ma problemy z synchronizacją, obsługą pamięci z pełną szybkością, oraz pracą z najszybszymi procesorami i kartami rozszerzeń. Na drugim krańcu są rozwiązania przeznaczone dla tzw. overclock-erów - czyli osoby które samodzielnie podnoszą częstotliwości taktowania oraz dobierają napięcia zasilające poszczególne elementy systemu komputerowego (głównie procesor i pamięć). Takie płyty kosztują zazwyczaj ponad 1000 PLN, są wykonane z najwyższą jakością, jednak przeciętny użytkownik komputera nigdy nie wykorzysta ich wszystkich możliwości. Do większości zastosowań najlepsze jest rozwiązanie pośrednie - firmowe płyty główne, solidnie wykonane lecz bez wodotrysków. Wydając trochę więcej niż na najtańsze rozwiązanie - kupujemy sobie spokój... ;)

Istotne jest dopasowanie płyty do rozmiaru obudowy. Aktualnie na rynku w tej dziedzinie obowiązuje kilka standardów:

- ATX – wymiary płyty to 305 x 244 mm
- microATX – wymiary płyty to 244 x 244 mm
- miniITX – wymiary płyty to 170 x 170 mm
- nanoITX – wymiary płyty to 120 x 120 mm
- picoITX – wymiary płyty to 100 x 72 mm

Ponadto ważne jest, by płyta była wyposażona we wszystkie niezbędne interfejsy.

WAŻNIEJSZE WYPROWADZENIA

ISA (ANG.)

Najstarszy, i w zasadzie już wychodzący z użycia standard 16-bitowej magistrali danych, umożliwiającej dołączanie dodatkowych kart rozszerzeń i oferującej "imponującą" przepustowość 8,33 MB/s. Rozwiązań nie występuje już we współczesnych płytach głównych.

EISA (ANG.)

Zgodna ze swoją 16-bitową poprzedniczką, 32-bitowa magistrala zewnętrzna, która ze względu na kompatybilność wstecz, pracuje z prędkością 8,33 MHz, jednak dostęp do pamięci odbywa się z pełną szybkością 33 MB/s. Ze względu na niezbyt spektakularne osiągi i dużą cenę wytwarzania, magistrala ta jest coraz rzadziej wykorzystywana. Rozwiązań nie występuje już we współczesnych płytach głównych.

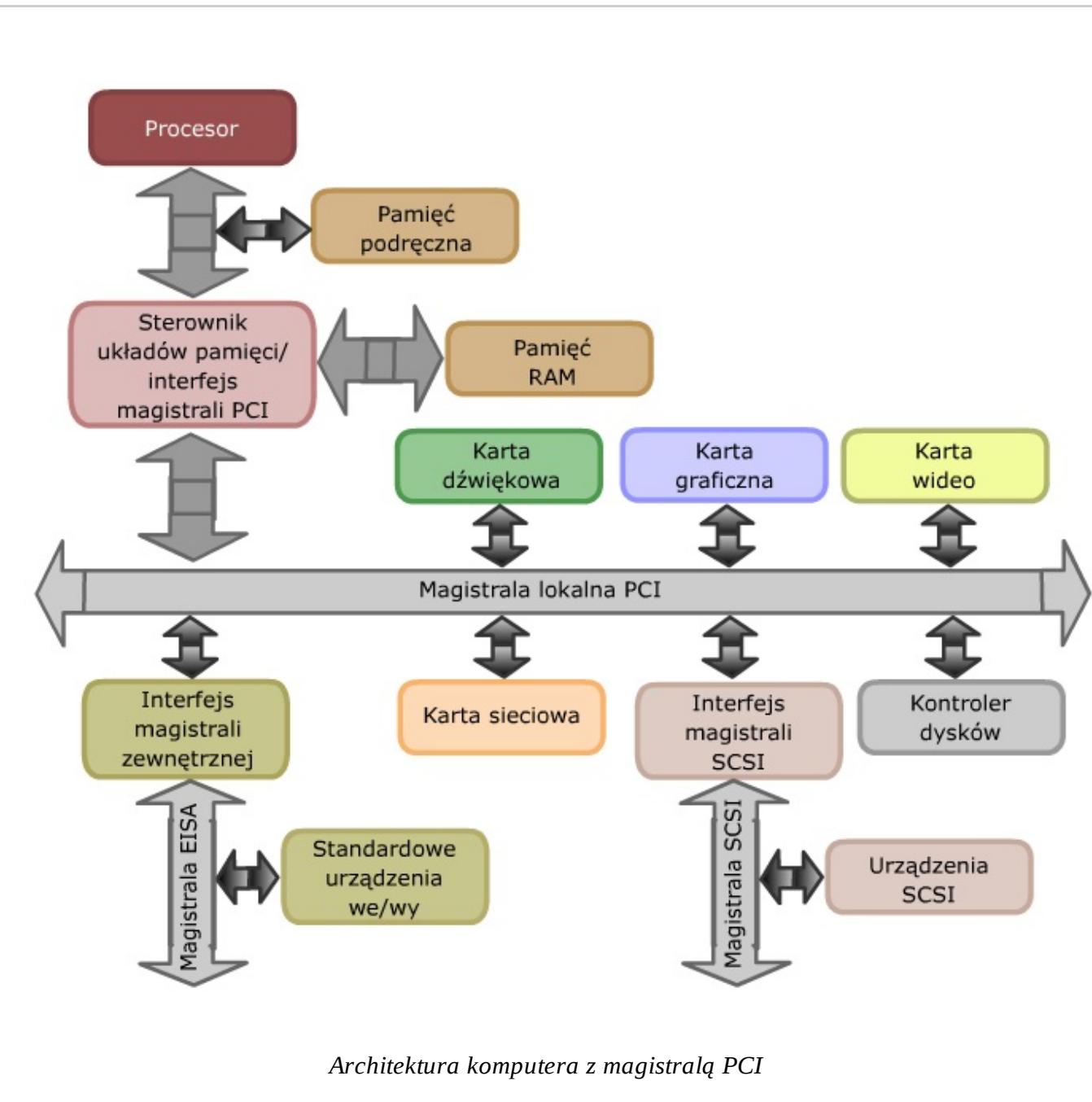
VESA LOCAL BUS (ANG.)

Standard ten najbardziej popularny w latach 1993-1996 miał stanowić wsparcie dla szybkich kart graficznych. Stanowił swoiste rozszerzenie technologii ISA o dodatkową magistralę danych, zwiększającą szybkość transferu pomiędzy procesorem a kartą graficzną, teoretycznie do 120 MB/s. 32-bitowa szyna VESA taktowana była zegarem procesora, którego częstotliwość pracy nie mogła przekraczać 40 MHz [1]. W pewnym czasie technologia stanowiła konkurencję dla droższej EISA, jednak szybko została wyparta przez technologię PCI. Rozwiązań nie występuje już we współczesnych płytach głównych.

PCI (ANG.)

Architektura ta została wprowadzona w 1993 roku przez firmę Intel i obecnie jest najpopularniejszym rodzajem gniazd rozszerzeń. Zaletą magistrali PCI jest możliwość tworzenia złożonych systemów (specyfikacja 2.1 PCI opisuje, że możliwa jest współpraca do 256 magistrali, z których każda może obsługiwać do 32 urządzeń PCI). Ciekawostką świadczącą o możliwościach eskalacji PCI niech będzie fakt, że w typowym PC wykorzystywana jest tylko jedna magistrala PCI obsługująca do 10 urządzeń [1]. Magistrala PCI może pracować z szybkością od 0 do 33 MHz (ver. 2.1 do 66 MHz), co daje przepustowość 132 MB/s. Magistrala pozwala na dostęp do adresowanego obiektu w jednym takcie zegara. Zdefiniowane jest również 64-bitowe złącze magistrali PCI, pracujące z prędkością do 264 MB/s.

Architekturę komputera ze złączem PCI przedstawia poniższy rysunek:



AGP (ANG.

AGP stanowi pewne przedłużenie magistrali PCI, dlatego określana jest jako interfejs komunikacyjny. Magistrala AGP nie przyspiesza operacji graficznych, a jedynie umożliwia bardziej wydajną pracę procesora graficznego [1]. Dzieje się tak, gdyż ów procesor sprawuje wyłączną kontrolę nad magistralą. AGP bazuje na specyfikacji PCI 2.1, zmienia jednak znaczenie niektórych sygnałów i wprowadza szereg nowych. Magistrala AGP może pracować w jednym z trzech trybów:

- Tryb 1x. Rozszerzenie standardu PCI, w którym dzięki podwojeniu częstotliwości pracy zegara taktującego do 66 MHz, uzyskano teoretyczny maksymalny transfer 264 MB/s.
- Tryb 2x. Częstotliwość zegara jest taka jak poprzednio, ale wymiana danych odbywa się podczas narastającego i opadającego zbocza sygnału taktującego. Daje to wzrost częstotliwości zegara do 133 MHz, a teoretyczna przepustowość wynosi 532 MB/s.
- Tryb 4x. Różnica w stosunku do trybu poprzedniego polega na tym, że w czasie zbocza wykonywane są dwie transmisje, a więc teoretyczna przepustowość wzrasta do 1064 MB/s. Tryb ten pracuje na znacznie obniżonym poziomie napięć - 1.5V.

PCI EXPRESS

Magistrala PCI Express (PCI-s PCIe, PCI-E), znana również jako 3GIO (od 3rd Generation I/O), jest pionową magistralą służącą do podłączania urządzeń do płyty głównej. Ma ona zastąpić magistralę PCI oraz AGP. PCI-Express stanowi magistralę lokalną typu szeregowego, łączącą dwa punkty (Point-to-Point). Nie jest to więc magistrala w tradycyjnym rozumieniu, i nie jest rozwinięciem koncepcji "zwykłego" PCI. Taka konstrukcja eliminuje konieczność dzielenia pasma pomiędzy kilka urządzeń - każde urządzenie PCI-Express jest połączone bezpośrednio z kontrolerem. Sygnał przekazywany jest za pomocą dwóch linii, po jednej w każdym kierunku. Częstotliwość taktowania wynosi 2.5GHz. Protokół transmisji wprowadza dwa dodatkowe bity, do każdych ośmiu bitów danych. Zatem przepustowość jednej linii wynosi 250MB/s. W związku z tym, że urządzenia mogą jednocześnie przekazywać sygnał w obydwu kierunkach (full-duplex) to można ewentualnie przyjąć, że w przypadku takiego wykorzystania złącza, transfer może sięgać 500MB/s. Możliwe jest kilka wariantów tej magistrali - z 1, 2, 4, 8, 12, 16, 24 lub 32 liniami (każda składająca się z dwóch 2 pinowych części - nadawczej i odbiorczej). Wraz ze wzrostem liczby linii wydłużeniu ulega gniazdo, jego konstrukcja (poprzez wspólną część początkową i jedynie dodawanie na końcu nowych linii) umożliwia włożenie wolniejszej karty do szybszego gniazda (w drugą stronę jest niemożliwe). Gniazdo 1x ma 18 pinów z każdej strony, gniazdo 4x - 32, gniazdo 8x - 49, zaś gniazdo 16x - 82 piny z każdej strony.



Gniazdo magistrali PCI Express

Na płytach głównych gniazda 16x montuje się zwykle w miejscu gniazda AGP na starszych płytach (ponieważ większość chipsetów z kontrolerem PCI Express nie zawierają kontrolera AGP, najczęściej obecność PCI-E eliminuje możliwość użycia kart graficznych ze złączem AGP, przykłady chipsetów obsługujących zarówno AGP jak i PCI-E to: Via PT880 Pro dla procesorów Intela i ULI M1695 + ULI M1567 dla procesorów AMD), pod nim gniazda 8x, 4x i 1x, najdalej zaś od procesora - gniazda PCI.

Specyfikacja określa też mniejsze rozmiarowo warianty kart: miniExpress cards, ExpressCards (następca PCMCIA) oraz AdvancedTCA (następca CompactPCI).

AMR, CNR i PTI

Wszystkie trzy gniazda zostały opracowane przez firmę Intel i ułatwiać mają instalowanie prostych urządzeń rozszerzających.

AMR (ang. *Audio modem Riser*) - pozwala na dołączenie modemu lub karty dźwiękowej.

CNR (ang. *Communication and Network*) - pozwala na rozbudowę o kartę sieciową 10/100 Mb/s.

PTI (ang. *Panel link TV-out Interface*) - pozwala na podłączenie kart TV.

ZŁĄCZA PAMIĘCI OPERACYJNEJ I MASOWYCH

SATA II / III to złącza wykorzystywane do podpinania pamięci zewnętrznej. Dzięki nim do komputera podłączymy dysk twardy lub napęd optyczny. Zdecydowanie lepszą opcją jest SATA III, gdyż jego przepustowość wynosi 6 Gbit/s, czyli ponad 2 razy więcej od SATA II. Warto także zadbać by płyta miała kilka takich złącz - nie ograniczy to możliwości rozbudowy jednostki w przyszłości. Istnieje także wersja standardu SATA wyprowadzana poza obudowę komputera - eSATA. Dzięki temu interfejsowi podłączymy do komputera dysk zewnętrzny, przy szybkości transferu 3 Gbit/s. Maksymalna długość kabla eSATA może wynosić 2 m, a w przeciwieństwie do USB, port ten nie potrzebuje zasilania (wyjątkiem jest hybrydowy port eSATAp). W przypadku pamięci operacyjnej - warto wybierać modele płyt głównych, które mają co najmniej 4 sloty na kości, dzięki czemu będziemy mieli możliwość rozbudowy pamięci operacyjnej w przyszłości.

BIOS , EFI

BIOS z definicji to zapisany w pamięci stałej komputera zestaw podstawowych procedur stanowiących ogniwo spajające między systemem operacyjnym a zamontowanymi podzespołami. BIOS jest programem zapisanym w pamięci ROM płyty głównej i innych kart rozszerzeń. Większość BIOS-ów obecnie jest zapisywana na pamięciach typu EEPROM, dzięki czemu jest możliwa ich aktualizacja.

W większości przypadków nie będziecie mieli potrzeby modyfikacji ustawień BIOS. Warto jednak mieć świadomość, że narzędzie to pozwala na bardzo wiele. W podstawowych wersjach BIOS-a, w niedrogich płytach głównych, znajdują się funkcje dotyczące zarządzania energią, bootowania systemu i ustawień dotyczących daty oraz godziny. Wiele modeli pozwala jednak na monitorowanie tak ważkich parametrów jak temperaturę procesora, napięcie czy prędkość obrotową wentylatora. To jednak nie wszystko – im bardziej wyspecjalizowana płyta główna (np. dla graczy lub overclockerów), tym bardziej zaawansowane funkcje są dostępne.

BIOS to nie jedyna opcja; już od jakiegoś czasu na rynku jest dostępny EFI (UEFI) – następca BIOS-u w komputerach osobistych z interfejsem znacznie bardziej przyjaznym dla użytkownika. Pomysł EFI zrodził się w firmie Intel na potrzeby procesora serwerowego Itanium. Jedną z jego cech charakterystycznych jest możliwość pisania dla niego sterowników, które są niezależne od systemu operacyjnego. EFI ma własne minisystemy operacyjne (UEFI Shell), które obsługują się poleceniami tekstowymi. Co ciekawe, EFI umożliwia obsługę dysków twardych większych od 2 TB, maksymalnie do 8192 EB (eksabajtów). Od 2006 r. firma Apple stosuje EFI na swoich maszynach, choć nie ma możliwości wejścia jego w zaawansowane ustawienia. Z EFI może się jednak wiązać kilka niedogodności - spośród których najdotkliwszą jest skomplikowana procedura instalacji systemu operacyjnego nie zawierającego wsparcia dla EFI (czyli Windows 7 czy starszych dystrybucji Linux-a).

PROCESORY

Dominujące rozwiązania

Współczesnie na rynku dominują dwie architektury procesorów. Jedna z nich, ogólnie lepiej znana, to procesory rodziny x86 i jej rozwinięcia x86-64. Ta architektura dominuje współczesnie na rynku komputerów osobistych, przenośnych i serwerów, w zasadzie nie mając tam znaczącej konkurencji. Spośród firm, które produkują procesory zgodne z tą architekturą należy wymienić Intel oraz AMD - jako najpoważniejszych graczy. Z drugiej strony - stale rosnący rynek systemów kieszonkowych - tabletów, smartphone-ów, i podobnych rozwiązań opiera się na innej architekturze - ARM, bardziej wyrafinowanej i z mniejszym "historycznym ogonem" niż x86. Tutaj także mamy wielu producentów procesorów, z których najbardziej znani to Qualcomm, Apple, Freescale, Samsung czy NVidia. W niniejszym rozdziale w skrócie przedstawimy Wam informacje o obu architekturach, wraz z opisem ich typowych przedstawicieli, włączając w to rozwiązania współczesne - jednakże pamiętajcie - ten opis zestarzeje się wyjątkowo szybko, dlatego też warto zawsze uzupełnić swoją wiedzę korzystając z internetu.

ARCHITEKTURA X86

Architektura x86 po raz pierwszy pojawiła się wraz z modelem programowym procesorów wykorzystanym przez Intela w 1978 roku w 16-bitowej procesorze Intel 8086. Procesor ten w 1981 roku wykorzystany został przez firmę IBM – a właściwie nieco jego okrojona wersja Intel 8088, w której zmniejszono do 8 bitów szerokość magistrali danych – do produkcji protoplasty dzisiejszych komputerów osobistych, pierwszego komputera IBM PC. Popularność jaką zyskały komputery typu PC, a później ich następcy doprowadziła do popularyzacji architektury x86 i jej dalszego rozwoju, który trwa do dzisiaj, finalnie wypierając alternatywne, nowocześniejsze rozwiązania. Swoją nazwę architektura x86 zawdzięcza generacjom układów Intel, wykorzystujących zastosowany w procesorze Intel 8086 model programowy, a które oznaczane były kolejno jako Intel 80286, Intel 80386 oraz Intel i486. Procesor Intel i486 był też ostatnim modelem zawierającym na końcu nazwy cyfry 86. Jego następca, procesor piątej generacji zgodnej z modelem programowym x86, nosił nazwę Pentium.

Pierwsze układy x86 były typowymi realizacjami CISC - przetwarzającymi złożone rozkazy w kilku / kilkunastu cyklach zegara, co skutkowało dość skomplikowaną realizacją fizyczną procesora z jednej strony - z drugiej upraszczało jego programowanie. Co ważne, ten podstawowy model programowy architektury x86 przetrwał do dzisiaj – został on oczywiście przez lata rozbudowany i zmodyfikowany, ale kompatybilność w dół została zachowana. Dzięki temu nawet dziś na najnowszych układach zgodnych z architekturą x86 można bez problemu uruchomić najstarsze aplikacje pochodzące jeszcze z 1981 roku. Nie umożliwia tego w prosty sposób żadna inna platforma.

Pierwsze zmiany w architekturze x86 wprowadzone zostały już w drugim z tej rodziny układzie 80286, gdzie dodano nowe instrukcje, sposób adresowania pamięci, a także wprowadzono potokowe przetwarzanie danych. W 1985 roku pojawiła się 32-bitowa kość Intel 80386, znana też pod nazwą i386DX. W układzie tym wprowadzone zmiany były znacznie istotniejsze. {o pierwsze - zastosowano 32-bitową magistralę adresową oraz 32-bitową magistralę danych. Oczywiście rozszerzone zostały do 32-bitów też rejestrów ogólnego przeznaczenia, co w połączeniu ze zmianami w jednostkach wykonawczych spowodowało, że był to procesor 32-bitowy. Dodano także nowe rejestrów kontrolne. Istotną zmianą było dodanie do procesora jednostki zarządzania pamięcią MMU (Memory Management Unit).

Następcą układów Intel 80386 stała się rodzina procesorów Intel i486 (na rynku pojawiły się 1989 rok), które pod względem modelu programistycznego były niemal identyczne z układami i386DX – dodano

w nim jedynie kolejne instrukcje. Spore zmiany nastąpiły natomiast w mikroarchitekturze układu – po raz pierwszy pojawiła się wspólna dla danych i instrukcji pamięć podręczna pierwszego poziomu (cache L1) o pojemności 8 KB oraz zintegrowany z procesorem koprocesor arytmetyczny FPU (Floating Point Unit), wykonujący operacje zmiennoprzecinkowe. Co ważne, procesory i486 jako pierwsze spośród układów x86 większość instrukcji wykonywały dokładnie w jednym taktie zegara.

Kolejną, piątą generację procesorów zgodną z architekturą x86 nazwano Pentium - od greckiej liczby pięć (pente). Pierwszy model, który pojawił się w 1993 roku składał się, mówiąc w pewnym uproszczeniu, z połączonych ze sobą dwóch układów i486. Taka konstrukcja pozwalała, w zależności od sytuacji, wykonywać dwa rozkazy równolegle. Pewnie niewielu użytkowników komputerów osobistych pamięta procesory Pentium 60 MHz ... Te pierwsze procesory zasilane były napięciem 5V. Kolejna grupa procesorów Pentium (P54C) pracowała przy mniejszym napięciu zasilającym (3,3V), a częstotliwość taktowania jądra oscylowała w granicach 200 MHz.

Procesory Pentium są 32-bitowe - rejestyry procesora mają 32-bity, jednakże wewnętrzne ścieżki mają nawet do 256 bitów szerokości, dzięki czemu prędkość wewnętrznych transferów jest większa niż mogłoby wynikać z owych 32-bitów. Pentium posiada dwa potoki przetwarzające instrukcje stałoprzecinkowe (U i V) oraz jednostkę zmiennoprzecinkową. Jeżeli jest to możliwe, procesor przetwarza w każdym cyklu dwie instrukcje (po jednej na potok). Potoki pracują wyłącznie w trybie synchronicznym, niemożliwe jest więc np. przewidywanie skoków przez jeden z nich, ponieważ zatrzymanie jednego z potoków prowadzi do zatrzymania drugiego. Jednak Pentium posiada dodatkowy moduł dynamicznego przewidywania rozgałęzień BPU (ang. *Branch Prediction Unit*). Moduł ten przewiduje rozgałęzienia, czyli "wkłada" do potoku te instrukcje, które jego zdaniem będą wykonywalne. Błędne przewidzenie rozgałęzienia powoduje, że jeden z potoków musi wstrzymać przetwarzanie, a więc i drugi potok nie może być realizowany.

Pentium może działać w systemach wieloprocesorowych, ale tylko z identycznym procesorem (*Dual system*). Montowany jest w gniazdach rozszerzeń typu Socket 7.

Istotna modyfikacja architektury x86 nastąpiła zaś w 1995 roku wraz z premierą układu Pentium MMX (oznaczenie kodowe P55C), kiedy to do listy rozkazów dodano zestaw 57 nowych instrukcji MMX (MultiMedia eXtension) oraz specjalny moduł wykonawczy do ich przetwarzania. Procesor Pentium MMX był jednostką typu SIMD (Single Instruction Multiple Data), czyli wykonuje operację na kilku grupach danych jednocześnie. Rozszerzenie spowodowało pojawienie się nowych instrukcji (i nowych wyprowadzeń) oraz nowych rejestrów wykorzystywanych do tego celu. Istotne zmiany procesora Pentium MMX w stosunku do jego poprzednika, to:

- podwójne napięcie zasilające (od wersji 166 MHz) - układy odpowiadające za współpracę z magistralami I/O zasilane są innym napięciem niż rdzeń procesora;
- zwiększoną pamięć podręczną L1 (ang. *first-level cache*) do 16kB - pracuje z poczwórną asocjacją;
- stos powrotu (ang. *return stack*) - przy przejściu do wykonywania podprogramu zapamiętywany jest adres powrotu do programu aktualnie wykonywanego;
- bufor zapisu jest zwiększony z 2 do 4 słów;
- nowa doskonalsza jednostka BPU (taka jak w Pentium Pro);
- możliwość równoległego wykonywania rozkazów (w sprzyjających warunkach 2 polecenia równolegle);

Modele Pentium MMX dzięki wykorzystaniu ww. nowinek (ale bez nowych instrukcji) osiągnęły wzrost mocy obliczeniowej do ok. 20 %.

Pentium MMX był ostatnim procesorem Intela wykorzystującym mikroarchitekturę CISC. Kolejne generacje układów zgodnych z x86 tego producenta tylko zewnętrznie są układami CISC, a wewnętrznie są już klasycznymi układami typu RISC. We wszystkich nowoczesnych procesorach rozkazy x86 przetwarza się bowiem na proste mikropolecenia zgodne z architekturą RISC. Co ciekawe, pierwszym układem zgodnym z architekturą x86, który wewnętrznie był procesorem RISC był NextGen Nx586 z 1994 roku, na bazie którego powstał w 1995 roku AMD K5.

Pierwszym układem Intela z wewnętrzną architekturą RISC był Pentium Pro, również z 1995 roku, który przeznaczony był głównie do serwerów i wydajnych stacji roboczych. Na jego bazie powstawały kolejno układy Pentium II (w 1997 roku) i Pentium III (1999), a później Intel Core (2006), Core 2 Duo (2006), a także współczesne Intel Core i3, i5 oraz i7. Przy czym wraz z kolejnymi generacjami procesorów pojawiały się kolejne modyfikacje (czy też raczej rozszerzenia) modelu programistycznego x86, np pojawienie się się rozkazów i jednostek SSE (Streaming SIMD Extension) w procesorze Pentium III, które są znaczco wydajniejsze niż MMX.

Pentium II to model, który pierwszy przełamał granicę taktowania 200 MHz (stąd już tylko krok do 1GHz). Tym co odróżniło go od poprzedników był zupełnie nowy image. Otóż procesor Pentium II pojawił się na pokładzie specjalnej karty o 242 końcówkach, którą wkłada się do złącza krawędziowego o nazwie Slot 1 (patrz rysunek dla Pentium 3). Dodatkową zmianą było oddzielenie pamięci podręcznej L2 (ang. *Second Level Cache*) od procesora, a stało się to dlatego, że taki sposób produkcji był znacznie tańszy. Pamięć L2 miała rozmiar 256/512 kB, taktowana była częstotliwością równą połowie taktu zegara procesora i znajdowała się na płytce wraz z procesorem.

Pentium II przetwarzał dane w trzech równoległych dwunastostopniowych potokach, a jądro procesora, jak wspomnieliśmy - jest wykonane w architekturze RISC. W praktyce instrukcje z modelu x86 rozkładane są na proste mikrooperacje i grupowane w centralnym zbiorniku (ang. *Instruction Pool*). Dzięki temu zbiornikowi pobieranie kolejnych kodów x86 z pamięci operacyjnej jest niezależne od ich wykonania. Ze zbiornikiem instrukcji połączony jest układ dyspozytora (ang. *Dispatcher*), który kieruje mikroinstrukcje do właściwych jednostek wykonawczych (ang. *Execute*). Dyspozytor kieruje do wykonania te mikrokody, które aktualnie nie czekają na żadne wyniki pośrednie z innych operacji. Wykonane instrukcje RISC trafiają znowu do zbiornika instrukcji (nie wiadomo, czy układ przewidywania rozgałęzień dobrze określił czy dana instrukcja rzeczywiście jest potrzebna). Instrukcja uznana za wykonaną zostaje "wyrzucona" ze zbiornika i przechodzi na "emeryturę" (ang. *retire*) i dopiero w tym momencie wyniki jej działania są zapisane.

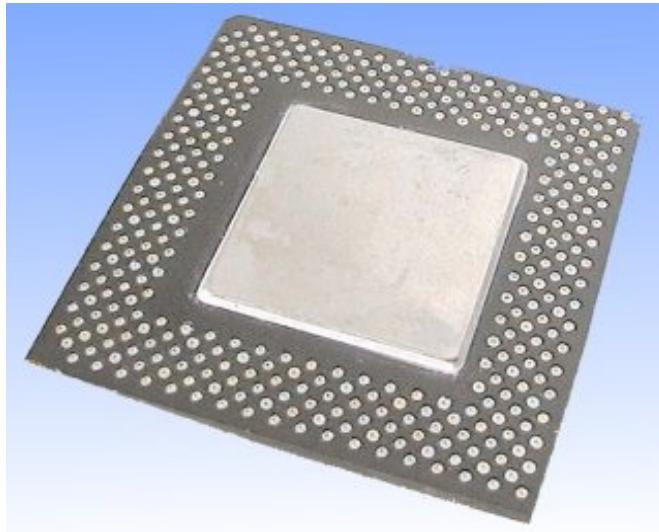
Następny procesor z serii Pentium III (nazwany roboczo Katmai), tak jak jego poprzednik, miał architekturę 32-bitową. Ponadto posiadał zintegrowaną 512 kB pamięć podręczną drugiego poziomu, wykorzystywał poszerzoną 256-bitową szynę BSB (ang. *Back Side Bus*) zapewniającą szybszy transfer wewnętrzny danych. Procesor zasilany był napięciem 1.65/1.7 V (zależnie od modelu). W Pentium III rozszerzono listę rozkazów - dodano ponad 70 rozkazów określanych jako ISSE (ang. *Internet Streaming SIMD Extentions*), czyli wykorzystanie znanych z MMX technik przetwarzania kilku danych z użyciem jednego rozkazu. Co ważne, ISSE potrafi operować także na danych w formacie zmiennoprzecinkowym, co wykorzystywane jest do obliczania obrazów 3D, rozpoznawania mowy, itp...



Procesor Pentium III

Równolegle z procesorami Pentium II firma Intel wprowadziła do sprzedaży ich tańszy odpowiednik - serię nazwaną Celeron. Pierwsze procesory nazywano Celeron A Medocino (określenie "medocino" identyfikuje strukturę krzemową). Wyposażone były w zintegrowaną pamięć podręczną L2 o skromnym rozmiarze 128 kB, która była taktowana z pełną prędkością zegara procesora, czyli z prędkością cache L1. Medocino współpracuje z magistralami 66/100MHz. Taka konstrukcja procesora zapewniła mu dużą wydajność, przy stosunkowo niskim koszcie.

Dodatkowo, Celeron A jako pierwszy dał możliwość "podkręcania" (ang. *overclocking*) częstotliwości pracy magistrali systemowej współpracującej z tym procesorem, a więc częstotliwości pracy procesora (znane są przypadki podkręcania Celeron'a 333 do ok. 700 MHz). Sprawiło to, że Celeron A stał się w swoim czasie rynkowym przebojem i idealną "*maszyną do overclockingu*".

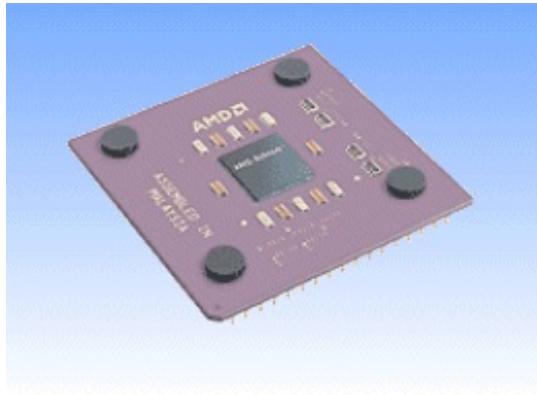


Procesor Celeron

W drugiej połowie lat 90 istniała silna konkurencja na rynku procesorów zgodnych z architekturą x86. Najważniejsi gracze to były firmy VIA (specjalizująca się w wyjątkowo tanich układach) oraz AMD (konkurująca nie tylko ceną ale też wydajnością swoich rozwiązań). AMD między innymi produkowała procesory Duron - opracowane głównie z myślą o klientach, którzy potrzebują dużej mocy obliczeniowej za niewielką cenę. Ograniczenie kosztów możliwe stało się dzięki zmniejszeniu ilości pamięci cache poziomu L2 do 64 KB. Pamięć cache poziomu L1 w procesorze Durom wynosi 128 KB, a więc po raz pierwszy w historii procesor rodziny x86 posiada mniejszy cache poziomu L2 niż poziomu L1. Nie powoduje to jednak znacznego spadku wydajności dzięki zastosowaniu specjalnej organizacji pamięci podrzcznej. Wykluczająca się architektura dostępu do pamięci cache (ang. *exclusive cache*) to rozwiązanie, które pozwala procesorowi używać sumarycznej pojemności pamięci cache L1 i L2 tak, jakby to była pamięć poziomu L1 o średnim czasie dostępu 1.5 cyklu zegara.

Zaimplementowana w Duronie interpretacja rozkazów 3DNow! i *Enhanced 3DNow!*, które wspomagają operacje multimedialne, pozwalała na równoległe przetwarzanie kilku liczb arytmetyki stałoprzecinkowej jak i zmiennoprzecinkowej, a więc zwiększenie szybkości przetwarzania danych - była to realna alternatywa dla wprowadzanych przez Intel równolegle instrukcji SSE.

Innym procesorem AMD był Athlon Thunderbird - jego budowa wewnętrzna była niemal identyczna jak ta, która charakteryzuje procesor AMD Duron. Najbardziej istotną różnicą jest rozmiar pamięci cache L2 działający z pełną szybkością jądra układu, który wynosi 256 KB. Pamięć ta pracuje w trybie *exclusive L2 cache memory*. Procesory Athlon współpracowały z 200/266 MHz magistralą systemową, dzięki czemu możliwe jest dostarczenie do procesora 1.6/2.13 GB danych. Ten wydajny interfejs magistrali zewnętrznej Thunderbirda - EV6 - pochodził od procesorów Alpha firmy DEC.



Procesor Athlon Thunderbird

W 2000 roku firma Intel postanowiła odejść od wywodzącej się od Pentium Pro wewnętrznej architektury RISC. Nowa wewnętrzna architektura o nazwie NetBurst zadebiutowała w procesorach Pentium 4. Teoretycznie mikroarchitektura NetBurst została zaprojektowana pod kątem osiągania bardzo dużych częstotliwości zegara taktującego (między innymi wydłużono potok wykonawczy do 20 etapów, a w kolejnych odsłonach tej architektury – np. w jádrze o kodowej nazwie Prescott – liczył on nawet 31 etapów), w praktyce okazało się, że w seryjnie produkowanych procesorach nigdy nie udało się uzyskać taktowania wyższego niż 4 GHz.

Procesor Pentium 4 był pierwszym od dłuższego czasu, w pełni nowym opracowaniem - a nie modyfikacją poprzednich. Firma Intel - czując na plecach konkurencję AMD, postanowiła kompletnie przeprojektować jądro procesora. Wprowadzony do sprzedaży w końcówce 2000 roku PIV był procesorem 32 bitowym, mogącym pracować z wyjątkowo wysokimi częstotliwościami zegara - w zasadzie limitowanymi jedynie poprzez ograniczenia w możliwości odprowadzenia ciepła.

W styczniu 2004 roku zadebiutowały procesory P4 z zupełnie nowym jádrem "Prescott". Produkowane w technologii 90 nm nowe jádro w pełni uzasadniałoby nazwanie tego procesora P5 - lecz ze względów marketingowych Intel nie zdecydował się na taki krok. Podstawową zmianą było rozszerzenie listy rozkazów procesora o instrukcje 64-bitowe (lista instrukcji x86-64), co spowodowało, że był to pierwszy procesor Intela przeznaczony na masowy rynek, który możnaby nazwać 64-bitowym. Pozostałe zmiany były mniej rewolucyjne. Poprawiono kolejkowanie rozkazów, co pozwoliło na zwiększenie szybkości przetwarzania, poprawiono wewnętrzną architekturę, co spowodowało dalsze możliwości wzrostu częstotliwości zegara (do 3.8 GHz), oraz wprowadzono kolejną rewolucję - technologię HT (nie we wszystkich modelach).

P4 posiadał nowy mechanizm mający na celu poprawienie wydajności układu. Jest to 8 kB bufor śledzenia wykonywania instrukcji (ang. *Execution Trace Cache*), przechowujący kod x86 w postaci mikrooperacji (nie ma konieczności odkodowywania instrukcji, gdyż są już odkodowane). Ponadto ALU pracuje dwa razy szybciej niż reszta procesora, a więc np. 2.8 GHz dla procesora 1.4 GHz. Mechanizm ten nosi nazwę *Rapid Execution Engine* i umożliwia zakończenie operacji na danych stałoprzecinkowych w połowie cyklu zegarowego.

Najmniej zmienioną częścią P4 Intela była jednostka zmienoprzecinkowa. Tak jak dla P3 wykorzystywane są dwa kanały do obliczeń zmienoprzecinkowych. Poszerzony został zestaw instrukcji SSE (nazywane SSE2, oraz dla jáder Presscot SSE3), umożliwiając wykonywanie operacji na liczbach 128 bitowych o podwójnej precyzji oraz na 128 bitowych liczbach stałoprzecinkowych.



Procesor Pentium 4

Dość szybko okazało się, że bez możliwości dalszego przyspieszania zegara mikroarchitektura NetBurst zaczęła wyraźnie przegrywać pod względem wydajności z konkurencyjnymi procesorami Athlon firmy AMD. Mimo to Intel konsekwentnie „obstawał” przy swojej technologii i ją rozwijał. Produkcji procesorów z architekturą NetBurst zaniechano dopiero w 2008 roku. W międzyczasie Intel wrócił do sprawdzonej architektury Pentium III, którą rozwinał i unowocześnił. Poligonem doświadczalnym dla niej były mobilne procesory Intel Core oraz Intel Core Duo, bazujące na jądrze o kodowej nazwie Yonah, na których przetestowano założenia nowej mikroarchitektury Intel Core Microarchitecture – m.in. współużytkowaną pamięć cache oraz strukturę wielordzeniową, wytwarzaną w jednym kawałku krzemiu. Układy te stały się podstawą do zaprojektowania procesorów Intel Core 2 Duo (nazwa kodowa Conroe), a później architektury Nehalem oraz kości Intel Core i3, i5 i i7 z najnowszym jądrem Sandy Bridge.

ARCHITEKTURA X86-64

W 2003 roku, wraz z wprowadzeniem na rynek przez firmę AMD układów Opteron (serwery) oraz Athlon 64 na rynku zadebiutowało kolejne rozszerzenie programowej architektury x86 – tym razem do 64-bitów. Co ciekawe, było to pierwsze w historii rozszerzenie modelu programowego, które opracowane zostało przez firmę AMD, a nie jak dotąd przez Intela – choć, gwoli ściśleści, trzeba zaznaczyć, że obie firmy nad nim pracowały wspólnie, jednak to AMD zdecydowała się jako pierwsza je zaimplementować w swoich produktach.

Athlon 64 i serwerowe Opteron wykorzystywały mikroarchitekturę K8 znaną też pod nazwą Hammer – wymienione procesory wykorzystywały jądra odpowiednio o kodowych nazwach Clawhammer i Sledgehammer. Mikroarchitektura Hammer wykorzystywała 64-bitowy model programistyczny o nazwie AMD64, nazywany też x86-64, a w wypadku układów Intela znany jako EM64T, który jest prostym rozwinięciem dotychczas stosowanej w procesorach x86 32-bitowej architektury IA-32.

Rozwinięcie modelu programistycznego x86-64 polega na rozszerzeniu zestawu instrukcji x86 o rozkazy operujące na 64-bitowych operandach. Zwiększoną została liczba rejestrów ogólnego przeznaczenia z ośmiu do szesnastu. Dodatkowo liczba 128-bitowych rejestrów SSE używanych w instrukcjach SIMD również zwiększoną została z ośmiu do szesnastu. Powiększenie liczby widocznych dla programisty rejestrów pozwoliło na około 5-15% zwiększenie wydajności.

Jedną z przyczyn, która spowodowała pojawienie się potrzeby rozszerzenia możliwości architektury x86 było coraz większe zapotrzebowanie programów na dostępną w systemie pamięć operacyjną. 32-bitowa architektura x86 pozwala na zaadresowanie 4 GB pamięci, z czego dla aplikacji - w zależności od systemu operacyjnego - dostępny jest najczęściej 2 GB lub 3 GB. Architektura x86-64 teoretycznie mogłaby zdefiniować 64-bitową przestrzeń adresową - lecz w praktyce została ona ograniczona do 48 bitów. W praktyce i to okazało się zbyt poważnym wyzwaniem technologicznym, więc w aktualnych implementacjach szyna danych została jeszcze bardziej ograniczona – ma 40-bitów. Pozwala ona obsłużyć 1 TB fizycznej i 280 TB wirtualnej przestrzeni adresowej, co jednak na dzień dzisiejszy jest ciągle raczej ograniczeniem wirtualnym niż rzeczywistym.

W architekturze x86-64 przewiduje dwa główne tryby pracy – tryb Long i tryb Legacy. Pierwszy z nich jest podstawowym trybem działania procesora w architekturze x86-64. Jest to kombinacja 64-bitowego trybu pracy i zgodnego wstecz trybu 32-bitowego. W trybie Long 64-bitowy system operacyjny może uruchamiać zarówno programy 64-, jak i aplikacje 32-bitowe oraz programy 16-bitowe – możliwość uruchamiania tych ostatnich zależy jedynie od mechanizmów zawartych w systemie operacyjnym, a nie od samego procesora. Drugi z trybów – tryb Legacy, jest używany przez 16-bitowe i 32-bitowe systemy operacyjne – w tym 32-bitową wersję systemu Windows 7. Jak można się domyślić mogą być w nim używane jedynie programy 16- i 32-bitowe – nie ma w tym trybie możliwości uruchomienia programów 64-bitowych. Procesor zgodny z architekturą x86-64 po włączeniu zasilania jest uruchamiany w trybie Legacy, a mówiąc ściślej w trybie rzeczywistym 8086, i do trybu Long musi zostać przełączony przez uruchomiony program (system operacyjny).

Oczywiście również Intel dysponuje obecnie procesorami 64-bitowymi zgodnymi z architekturą x86-64. Są to układy ze wspomnianą wyżej technologią EM64T (Extended Memory 64 Technology). Projekt wdrożenia przez Intela 64-bitowych rozszerzeń x86-64 znany pod kodową nazwą Yamhill ogłoszony został w lutym 2004 roku na konferencji IDF. Pierwszy procesor Intel'a z rozszerzeniami EM64T pojawił się w czerwcu 2004 roku – był to serwerowy Xeon z rdzeniem Nocona, który obsługiwał 64-bitowe instrukcje nieoficjalnie. Procesory przeznaczone dla komputerów osobistych z rdzeniem Prescott, produkowane od lutego 2004 roku, miały tę technologię zablokowaną. Oficjalna premiera technologii EM64T nastąpiła w 2005 roku, wraz z premierą procesorów Pentium 4 z jądrem Prescott-2M. Trzecim producentem procesorów zgodnych z programową architekturą x86-64 jest tajwańska firma VIA – procesory począwszy od modelu VIA Nano (nazwa kodowa Isaiah) z 2008 roku.

Wprowadzenie rozszerzeń x86-64 to nie ostatnia ważna zmiana w modelu programowym x86. Od czasu jej wprowadzenia pojawiły się bowiem kolejne rozszerzenia modelu programowego, jak następne wersje SSE (ostatnia z nich to SSE 4.2), rozbudowujące listę instrukcji oraz, przede wszystkim, wektorowe rozszerzenie AVX (Advanced Vector Extensions). W AVX wprowadzono 256-bitowe rejestrze, które są dwa razy większe niż wykorzystywane w rozszerzeniach SSE. Nowych rejestrów jest w sumie 16 (osiem dotychczasowych rejestrów SSE wydłużono ze 128 do 256 bitów oraz dodano osiem nowych) i noszą one nazwy YMM0...YMM15. Dodano 19 rozkazów działających wyłącznie na rejestrach YMM oraz czteroargumentowe rozkazy akumulujące wyniki mnożenia wektorów liczb zmiennoprzecinkowych (12 instrukcji), a także sześć rozkazów wspomagających szyfrowanie AES. W sumie zestaw rozkazów w najnowszych procesorach zgodnych z architekturą x86 liczy obecnie 678 instrukcji.

Implementacje tej architektury w procesorach to przede wszystkim w przypadku firmy Intel seria Core 2, ale wcześniej debiutowała w procesorze Pentium IV - w którym zwiększo rozmiar pamięci podręcznej L2 z 512 KB do 1 MB oraz wydłużono potok wykonawczy. W późniejszych wersjach dodano obsługę EDB (seria J) oraz technologię oszczędzania energii (stepping E0).

Nowy rdzeń Prescott wykonany w technologii 0,09 mikrona, 1 MB pamięci podręcznej L2 technologia oszczędzania energii: Enhanced SpeedStep Technology (tylko stepping E0) poprawione bezpieczeństwo dzięki Execute Disable Bit (tylko wersja J). Wprowadzono technologię HyperThreading – (dwa logiczne procesory)zwiększa płynność pracy systemu przy działających równocześnie kilku aplikacjach. Daje to wzrost wydajności o ok. 10-20%

Natomiast jeśli chodzi o sam procesor, to pierwsza wersja Prescotta z technicznego punktu widzenia

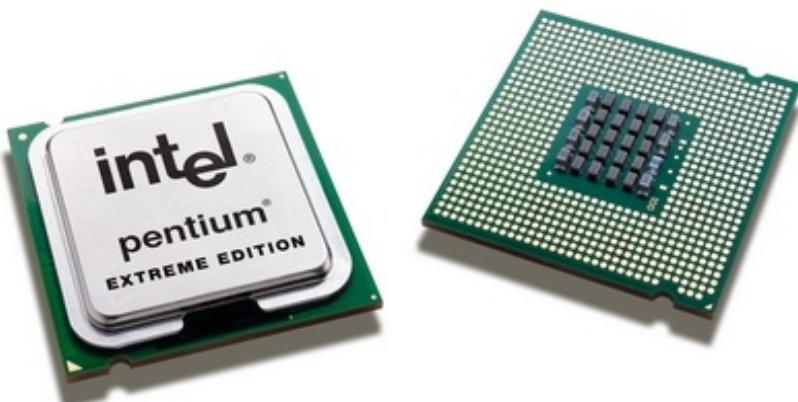
okazała się klapą. Przejście na technologię 90 nm spowodowało liczne problemy, z którymi Intel nie zdołał się na czas uporać. W efekcie Prescott bardzo się grzał, nie będąc jednocześnie specjalnie szybszym od swojego poprzednika opartego na jądrze Northwood - nawet pomimo dwukrotnego zwiększenia pamięci cache.

Na płytach z Chipsetami do tego procesora po raz pierwszy zastosowano nową magistralę PCI-Express. Z myślą o zwiększonym poborze energii wprowadzono nowe, beznóżkowe gniazda LGA775.

Nowszą serią były procesory rodziny Core / Centrino. Były to pierwsze produkty wielordzeniowe przeznaczone do zastosowania w urządzeniach przenośnych. Z punktu widzenia użytkownika dwa rdzenie w notebooku oznaczają "tylko" dalszy, choć istotny wzrost szybkości laptopów. Dla inżynierów projekt stanowił nie lada wyzwanie. To, co od biedy ujdzie w stacjonarnym peccie, w urządzeniu mobilnym może być wadą nie do zaakceptowania: kilka watów poboru mocy więcej w "zwykłym" procesorze da się zrekompensować odpowiednim chłodzeniem; w notebookowym układzie taka różnica przełoży się na fiasko całego przedsięwzięcia. Intelowi udało się pogodzić ogień z wodą, i seria Core 2/Centrino stała się rynkowym sukcesem.

Oprócz zmiany nazwy CPU z Pentium na Intel Core wprowadzona została nowa numeracja modeli układu. Zamiast trzycyfrowego kodu procesor oznaczany jest literą symbolizującą klasę poboru mocy i czterocyfrową liczbą będącą wskaźnikiem wydajności.

Następcą Core był Core 2 Duo. Mimo zbliżonej nazwy do Core Duo - to jest zupełnie nowy procesor, przeznaczony nie tylko do Laptopów, lecz także do systemów stacjonarnych.



Procesor Core 2 Duo

W tych procesorach zastosowano dość krótki potok wykonawczy, który składa się z 14, a nie 31 etapów, jak w Pentium 4. Obniżyło to maksymalną częstotliwość pracy i ograniczyło zużycie energii, a procesor się nie przegrzewa. Ponieważ jego wydajność zależy od liczby instrukcji wykonywanych w każdym takcie zegara sterującego i od częstotliwości, bardzo skuteczne okazało się zastosowanie czwartej jednostki wykonawczej.

Aby została wykorzystana, należało zastosować układ niekolejnych pobrań z pamięci (Memory Disambiguation), dzięki czemu dane do przetwarzania są w pamięci podręcznej i na czas trafią do potoku wykonawczego. Poważną modyfikacją jest wykonywanie wszystkich instrukcji SSE i SSE2 w jednym takcie zegara, zamiast w dwóch. Wobec braku zintegrowanego kontrolera pamięci bardzo istotną funkcję pełni współdzielona przez rdzenie pamięć podręczna drugiego poziomu. W zależności od wykonywanej aplikacji może być w całości przydzielona jednemu rdzeniowi lub w dowolnych proporcjach obu. Pamięć ta połączona jest z rdzeniami 256-bitową szyną danych, co pozwala jednocześnie przesyłać 4 bajty danych.

W tańszych procesorach Core 2 Duo E6400 (2,13 GHz) i E6300 (1,86 GHz) pamięć cache L2 była ograniczona do 2 MB. W szybszych wersjach procesorów: Core 2 Due Extreme X6800 (2,93 GHz), Core 2 Duo E6700 (2,66 GHz) oraz Core 2 duo E6600 (2,4 GHz) miała wielkość 4 MB. Pamięć podrzczna o wielkości 4 MB składa się z około 270 milionów tranzystorów co stanowi ponad 90 procent łącznej liczby tranzystorów w układzie. Jednak dla kosztów wytwarzania układów istotniejsze jest to, że cache zajmuje jedynie około połowy powierzchni struktury krzemowej, która przy produkcji w procesie technologicznym 65 nm zajmuje jedynie 143 mm².

W uzyskaniu dużej wydajności przez nowe procesory pomagają także inne usprawnienia. Pamięć podrzczna L1 procesorów przechowuje instrukcje w postaci wstępnie zdekodowanej. Bufory dla pobieranych danych są większe, więc mogą przechowywać więcej informacji. Instrukcje x86 rozkładane są na tzw. makro operacje, dzięki czemu mogą być łączone w celu lepszego wykorzystania przepustowości połączeń. Przykładem takiego połączenia jest para Cmp+Jmp (porównaj i skocz), która bardzo często występuje w kodzie programów.

Bardzo istotną modyfikacją jest wprowadzenie układu śledzącego wskaźnik stosu (Extender Stack Pointer). Wszelkie obliczenia na tym rejestrze adresowym nie zajmują jednostek obliczeniowych procesora, a jednocześnie odpowiedni adres jest dostępny zawsze wtedy, gdy jest potrzebny bez konieczności oczekiwania na wyliczenie go.

AMD Athlon 64 to pierwsze procesory 64-bitowe (AMD64) przeznaczone dla wydajnych stacjonarnych zestawów komputerowych. Procesory te zależnie od modelu, mają od 512 do 1MB wbudowanej pamięci podrzcznej L2 oraz na gnieździe Socket 754 i Socket 939. Taktowanie procesora było od 1800(2800+) do 2400MHz(4000+) lub 2600MHz(4000+) dla AM2.

Zwiększoną przestrzeń adresową pojedynczego procesu umożliwia procesowi odwoływanie do 256 TB pamięci wirtualnej. Procesory AMD64 mogą zaadresować do 1TB fizycznej pamięci operacyjnej. W celu maksymalnego skrócenia dostępu do pamięci operacyjnej procesory tej firmy mają zintegrowany kontroler pamięci. Układ oznaczony symbolem Athlon 64 i Sempron 3300+ na gnieździe Socket 754. Podobnie jak procesory Intela obsługują instrukcje multimedialne SSE3. Procesor jest wykonany w technologii 90nm pracujący z częstotliwością 2.0 GHz i pamięcią podrczną L2 128Kb. Procesor AMD ma specjalną szynę do komunikacji z pamięcią RAM a z resztą urządzeń takich jak kontrolery PCI-E , podłączony jest przez magistralę HyperTransport pracującą z częstotliwością do 2.6 GHz(dla wersji 3.0). Dla przykładu HyperTransport 2.0 pracuje przy 1.4 GHz i ma przepływność 22.4GB/s.

Pełne wykorzystanie mocy obliczeniowej tych procesorów jest możliwe dopiero po zainstalowaniu 64 bitowego systemu operacyjnego

Opracowana była również wersja FX procesorów AMD - 64-bitowych dla stacji roboczych, przeznaczona dla gier lub bardziej skomplikowanych obliczeń graficznych. Procesory FX były porównywalne pod względem wydajności z Intel Core 2 Duo, przy zazwyczaj niższej cenie.

Firma AMD także wprowadziła na rynek procesory wielordzeniowe (tak jak Core 2) - np. dwurdzeniowy procesor AMD Athlon 64 X2, składał się z dwóch jednostek obliczeniowych połączonych w jednym układzie, wykonujących obliczenia na dwóch strumieniach danych.

Równolegle rozwijana była seria AMD Operton - produktów firmy AMD przeznaczony dla stacji roboczych oraz serwerów. Zastosowana w nich technologia AMD64 z Architekturą Direct Connect zwiększała całkowitą wydajność systemu, eliminując tradycyjne wąskie gardła charakterystyczne dla dotychczasowych systemów. Dotychczas magistrala FSB ograniczała i przerywała przepływ danych. Przerwy w przepływie danych oznaczają mniejszą skalowalność systemu. W architekturze Direct Connect nie ma magistrali FSB. Zamiast tego rdzenie procesora, kontroler pamięci oraz układ We-Wy są połączone bezpośrednio ze sobą i komunikują się z częstotliwością zegara procesora.

WSPÓŁCZESNE PROCESORY DO KOMPUTERÓW OSOBISTYCH I PRZENOŚNYCH

Jak już wspomnieliśmy, współcześnie na rynku liczą się dwie firmy: Intel i AMD. W przypadku firmy Intel podstawową rodziną procesorów dla typowych zastosowań jest Core - występująca w trzech seriach (Core i3, i5 oraz i7). Są to wielordzeniowe (od 2 do 4 rdzeni) procesory 64 bitowe, zazwyczaj (choć nie zawsze) mające wbudowany prosty układ graficzny. Pełne oznaczenie procesora, oprócz serii, składa się z 4 cyfr i 1 lub dwóch liter. Pierwsza cyfra oznacza generację procesora, druga i trzecia oznacza relatywną wydajność procesora w danej serii. Innymi słowy - im wyższa trzecia cyfra, tym procesor wydajniejszy w serii wyznaczonej przez cyfrę drugą. Czwarta cyfra oznacza energooszczędność procesora, i w wersjach dla urządzeń stacjonarnych zawsze wynosi 0, ogólnie - im wyższa wartość, tym procesor jest bardziej energooszczędny. Litery mają przypisane następujące znaczenie:

- M - dla urządzeń mobilnych (Mobile)
- U - niskie zużycie energii (Ultra-low power), osiągane przez obniżenie napięcia
- Y - bardzo niskie zużycie energii (Extremely low power)
- H - grafika wysokiej jakości (High performance graphics) - oznacza wbudowany w procesor układ graficzny (kartę graficzną)
- R - grafika wysokiej jakości/do urządzeń stacjonarnych oparty na BGA1364
- K - brak blokady mnożnika (Unlocked)
- X - możliwość zwiększania mnożnika bez ograniczeń
- S - optymalizacja wydajności (Performance-optimized lifestyle)
- T - optymalizacja zużycia energii (Power-optimized lifestyle)
- MX - Mobile Extreme Edition
- MQ - czterordzeniowy procesor dla urządzeń mobilnych (Quad-Core mobile)

W założeniach, seria i3 miała być serią najmniej wydajną, i5 średkową, natomiast i7 najbardziej wydajną. Lecz ze względu na wewnętrzne różnice w każdym procesorze każdej serii przyjęcie założenia, że np i7 jest zawsze szybszy niż i5 jest błędne - są procesory i5 o zdecydowanie wyższej wydajności niż najsłabsze modele i7, przy porównywaniu należy uwzględnić wszystkie parametry procesora, a nie tylko jego serię.

Oprócz produktów z rodziny Core, Intel oferuje również rozwiązania serwerowe, czyli rodzinę Xeon, składającą się z czterech serii (od najbardziej wydajnej): E7, E5, E3, Phi - oznaczanej wg zasad zbliżonych, lecz nie identycznych, do rodziny Core. Inną rodziną procesorów firmy Intel jest Atom - budżetowe rozwiązanie o bardzo niskim zużyciu energii, przeznaczone do netbooków, tabletów czy smartfonów. Są to wyraźnie mniej wydajne procesory, często ciągle 32 bitowe, ale za to nie wymagające aktywnego chłodzenia.

Firma AMD oferuje zarówno procesory klasyczne (seria FX do komputerów ogólnego przeznaczenia oraz Operon do rozwiązań serwerowych) oraz zintegrowane z dość silnymi kartami graficznymi jednostki nazywane APU. Wydaje się, iż od jakiegoś czasu AMD pogodziło się z rolą "tego drugiego" producenta układów w architekturze x86-64, w związku z tym w dziedzinie klasycznych procesorów konkurowie głównie ceną, nie mając ambicji zaprezentowania najszybszych procesorów na rynku. W zamian za to stawia na nowe pomysły - tym jest właśnie APU i wewnętrzne elementy jego architektury. W najnowszej generacji, zwanej kodowo Kaveri, takim innowacyjnym elementem jest obsługa HSA. HSA to sposób konstruowania sprzętu komputerowego, w którym procesory każdego rodzaju wchodzące w skład jednego chip-a są równorzędne: mają równy dostęp do wspólnej pamięci operacyjnej oraz mogą sobie nawzajem przekazywać zadania. Na razie mamy na myśli rdzenie procesora oraz układ graficzny; w przyszłości będą mogły do nich dołączyć inne typy maszyn obliczeniowych. To znaczące odejście od tradycyjnych systemów, w których centralny procesor (CPU) jest „szefem” układu, procesor graficzny ma własny obszar pamięci, a komunikacja między nimi wiąże się z tłumaczeniem adresów i przekazywaniem obszarów pamięci pod kontrolę jednego, to drugiego procesora. Tutaj mamy zaimplementowane rozszerzenie architektury NUMA do standardu zwanego [hUMA](#) - czyli pamięć jest bezpośrednio wzajemnie połączona pomiędzy wszystkie jednostki wykonawcze każdego typu implementowane w procesorze. Ponadto - w odróżnieniu od grafiki wbudowanej w procesor przez firmę Intel, w przypadku AMD APU zintegrowane układy graficzne są zdecydowanie bardziej wydajne, pozwalające w wielu wypadkach nawet na uruchamianie współczesnych gier.

Ze względu na bardzo szybki postęp w tej dziedzinie, zamieszczanie w podręczniku opisów konkretnych modeli współczesnych procesorów jest z góry skazane na szybkie przedawnienie. Uważamy, że zdecydowanie lepiej będzie polecić Wam kilka źródeł w internecie, gdzie zawsze znajdziecie najbardziej aktualne informacje:

- www.benchmark.pl - strona poświęcona głównie testom sprzętu komputerowego, zawierająca także ciekawe opisy natury ogólnej
- www.chip.pl - strona czasopisma komputerowego, w której zawsze możecie znaleźć aktualne rankingi podzespołów PC
- www.notebookcheck.pl - kolejna strona poświęcona testom sprzętu komputerowego, w tym wypadku głównie przenośnego - także zawierająca często aktualizowane zestawienie procesorów
- www.pclab.pl - kolejna strona poświęcona testom sprzętu komputerowego

ARCHITEKTURA ARM

Architektura ARM (Advanced RISC Machine, pierwotnie Acorn RISC Machine) – 32-bitowa oraz 64-bitowa architektura i model programowy procesorów, będąca od początku architekturą typu RISC. Mimo iż nie widzicie tego na co dzień, procesory z tą architekturą są jednymi z najczęściej stosowanych procesorów na świecie. Używa się ich w dziesiątkach tysięcy rozwiązań wbudowanych (dyskach twardych, telefonach komórkowych, routerach, kalkulatorach, zabawkach dziecięcych, itp). Obecnie zajmują one ponad 75% rynku 32-bitowych CPU dla systemów wbudowanych. Od opisywanej wcześniej architektury x86 różni się bardzo wieloma względami technicznymi oraz fundamentalnie innym podejściem biznesowym. Stojąca za projektem ARM firma ARM sama nie produkuje własnych procesorów. Zamiast tego tworzy ich projekty i sprzedaje je innym firmom wg dwóch schematów:

- gotowe do implementacji projekty elektroniki (jądro Cortex) do osadzenia w swoich układach. Z tej pierwszej możliwości korzystają takie marki, jak Samsung (seria układów Exynos), Nvidia (seria układów Tegra), Broadcom (seria układów BCM, które są montowane w Nokiach z Symbianem), ST-Ericsson (seria NovaThor, często spotykana w smartfonach ze średniego segmentu) i Texas Instruments (seria OMAP).
- licencję na tworzenie układów podobnych do swoich i kompatybilnych z wybranym zestawem instrukcji. Z tej drogi, znacznie trudniejszej, w tym momencie korzysta jedynie Qualcomm (sprzedający układy z rdzeniami Snapdragon i Krait) i Apple (architektura Swift, która zadebiutowała wraz z iPhone'em 5), choć wiadomo też, że Nvidia od jakiegoś czasu konstruuje coś własnego, co może trafić do kolejnych generacji Tegr.

Projektowanie pierwszego procesora ARM rozpoczęło się w 1983 roku, jako projekt rozwojowy brytyjskiej firmy Acorn Computers Ltd. Grupa inżynierów kierowana przez Rogera Wilsona i Steve'a Furbera rozpoczęła projektowanie jądra będącego ulepszoną wersją procesora MOS 6502 firmy MOS Technology. Pierwsza wersja testowa, nazywana ARM1, opracowana została w 1985 roku, a rok później ukończono wersję produkcyjną ARM2. ARM2 wyposażony był w 32-bitową szynę danych, 26-bitową przestrzeń adresową oraz w szesnaście 32-bitowych rejestrów. Był to w tym czasie najprostszy szeroko stosowany 32-bitowy mikroprocessor, zawierający tylko 30 tysięcy tranzystorów. Prostota wynikała głównie z braku mikrokodu i, jak w większości procesorów w tym czasie, braku cache. ARM2 miał z tego powodu bardzo niski pobór mocy i jednocześnie szybkość przetwarzania większą od procesora Intel 80286. Następna wersja ARM3 produkowana była z 4KB cache, co jeszcze bardziej poprawiło wydajność.

W późnych latach osiemdziesiątych firma Apple Computer rozpoczęła współpracę z Acorn Computers w projektowaniu nowszej wersji jądra ARM. Projekt był na tyle istotny, że Acorn wydzielił grupę projektową tworząc w 1990 roku nową firmę o nazwie Advanced RISC Machines (ARM Ltd.). Wynikiem tej współpracy był procesor ARM6, udostępniony w roku 1990. Firma Apple użyła opartego na ARM6 procesora ARM610 w palmtopie (PDA) o nazwie Apple Newton. Jądro procesora ARM6 zawiera około 35 tysięcy tranzystorów i jest tylko niewiele większe od jądra ARM2 (30 tysięcy

tranzystorów). Dzięki swej prostocie jądro ARM może być łączone z dodatkowymi blokami funkcjonalnymi, tworząc w jednej obudowie, mikroprocesor dostosowany do konkretnych wymagań. Jest to możliwe, gdyż podstawą działalności ARM Ltd. jest sprzedaż licencji na zaprojektowane jądra. Dzięki temu powstały także mikrokontrolery oparte na architekturze ARM.

Firma DEC zakupiła licencję na architekturę ARM i na jej podstawie zaprojektowała procesor StrongARM. Przy częstotliwości 233 MHz procesor ten pobierał tylko 1W mocy (najnowsze wersje StrongARM pobierają znacznie mniej). Projekt ten został następnie przejęty przez firmę Intel, na podstawie umowy procesowej między obiema firmami. Dla Intel był to szansa na zastąpienie przestarzałej architektury i960 nową architekturą StrongARM. Na podstawie StrongARM Intel zaprojektował bardzo wydajny mikroprocesor o nazwie XScale.

Zgodnie z założeniami architektury RISC, rozkazy procesorów ARM są tak skonstruowane, aby wykonywały jedną określona operację i były przetwarzane w jednym cyklu maszynowym. Interesująca zmiana w stosunku do innych architektur jest użycie 4-bitowego kodu warunkowego na początku każdej instrukcji. Dzięki temu każda instrukcja może być wykonana warunkowo. Ogranicza to przestrzeń dostępną, na przykład, dla instrukcji przeniesień w pamięci, ale z drugiej strony nie ma potrzeby stosowania instrukcji rozgałęzień dla kodu zawierającego wiele prostych instrukcji warunkowych. Inną unikatową cechą zestawu instrukcji procesora ARM jest łączenie operacji przesunięcia i obrotu bitów w rejestrze z instrukcjami arytmetycznymi, logicznymi, czy też przesłania danych z rejestru do rejestru. Przedstawione cechy powodują, że typowy program zawiera mniej linii kodu niż w przypadku innych procesorów RISC. W rezultacie jest mniejsza liczba operacji pobrania/zapisania argumentów instrukcji, więc potokowość jest bardziej efektywna. Pomimo że procesory ARM są taktowane zegarem o stosunkowo niskiej częstotliwości są konkurencyjne w stosunku do znacznie bardziej złożonych procesorów.

Zestaw instrukcji i architektury ARM doczekały się już ósmiu generacji. W tym momencie na rynku dominują procesory kompatybilne z architekturą ARMv7 lub jej rozszerzeniem, nazywanym czasem ARMv7s, ARMv8 jest wdrażane do produkcji i na rynku powinno się pojawić już niedługo. ARM, tworząc nową rodzinę procesorów i związany z nią zestaw instrukcji, określa tylko, co będą umiały, ale nie wskazuje dokładnie, w jaki sposób ma to zostać osiągnięte. Dlatego w ramach jednej rodziny ARM projektuje kilka różnych rdzeni, różniących się wydajnością, możliwościami i zastosowaniem. Pierwszym przedstawicielem układów ARMv7 był rządzenie Cortex-A8, kojarzony głównie z pierwszą generacją Samsungów Galaxy S (miały one chip o nazwie Hummingbird, który składał się z jednego rdzenia Cortex-A8 taktowanego z częstotliwością 1 GHz i układu graficznego PowerVR SGX540) i z tym, że przyniósł spory wzrost wydajności w porównaniu z poprzednikiem. Po nim powstał Cortex-A9, spotykany teraz w większości smartfonów w konfiguracjach dwu- lub czterordzeniowych. Główną różnicą w stosunku do Cortex-A8 było to, że A9 umożliwiała tworzenie układów wielordzeniowych oraz nauczyła się wykonywać niektóre instrukcje poza kolejnością (ang. Out-of-Order Execution), co dało kilkunastoprocentowy wzrost wydajności. ARM zaprojektowało też rządzenie Cortex-A5 (spotykane na przykład w HTC Desire X) i Cortex-A7, które są kompatybilne z instrukcjami ARMv7s, ale mniej wydajne (na przykład nie umieją wykonywać instrukcji poza kolejnością) i przystosowane do tańszych i mniejszych układów scalonych.

Najwydajniejszym i najnowszym rdzeniem należącym do rodziny ARMv7 (z rozszerzeniami ARMv7s) jest Cortex-A15 Choć nadal należy on do siódmej generacji procesorów ARM, to poczyniono w nim bardzo dużo zmian w porównaniu z Cortex-A9. Cortex-A15 to rządzenie trójpotokowe, w którym całkowicie zmieniono i poszerzono część zajmującą się dekodowaniem i kolejkowaniem instrukcji, zwiększo liczbe jednostek wykonawczych, rozbudowano część procesora odpowiedzialną za dostarczanie danych i komunikację z pamięcią, itp.

A gdzie w tym wszystkim znajdują się procesory Apple i Qualcomm? Architektury Apple Swift i Qualcomm Krait to coś pomiędzy Cortex-A9 a Cortex-A15. Również są one trójpotokowe, tak jak Cortex-A15 (i w przeciwieństwie do dwupotokowego Corteksa-A9), ale nie mają aż tak rozbudowanych jednostek wykonawczych i układów wejścia/wyjścia, przez co mają mniejszą wydajność, ale też mniejsze zapotrzebowanie na energię, i są zbudowane z mniejszej liczby tranzystorów.

Wewnętrznie wszystkie te jednostki są zdecydowanie mniej skomplikowane niż współczesni reprezentanci architektury x86, i też mniej wydajne. Jednak stosunek stopnia skomplikowania (i co za tym idzie - ceny) do wydajności zdecydowanie przemawia za ARM - co dostrzegają też producenci tacy jak AMD, opracowując wersje procesorów ARM dla serwerów (np. AMD Operton A1100).

ZAMIAST PODSUMOWANIA

Jak widzicie - świat procesorów jest całkiem skomplikowany. Co więcej - rozwiązania techniczne w nich zmieniają się szybciej niż w innych dziedzinach techniki. Rewolucja porównywalna z wprowadzeniem układów CommonRail w samochodach, w dziedzinie konstrukcji procesorów trafia się raz na dwa / trzy lata. Z tego względu nie bardzo wiadomo, jak będzie rozwijał się świat procesorów. Z jednej strony - mamy procesy utrzymujące przy życiu skomplikowaną architekturę x86 w postaci istniejącego oprogramowania, z drugiej - rosnącą rolę zastosowań kieszonkowych, przenośnych i innych. Jedyne co możemy wam doradzić - to na bieżąco uzupełniać wiedzę o procesorach i kierunkach ich rozwoju korzystając z sieci internet.

PAMIĘĆ OPERACYJNA

Elektroniczne pamięci półprzewodnikowe, które są stosowane w systemach komputerowych, zazwyczaj należą do jednej z dwóch grup:

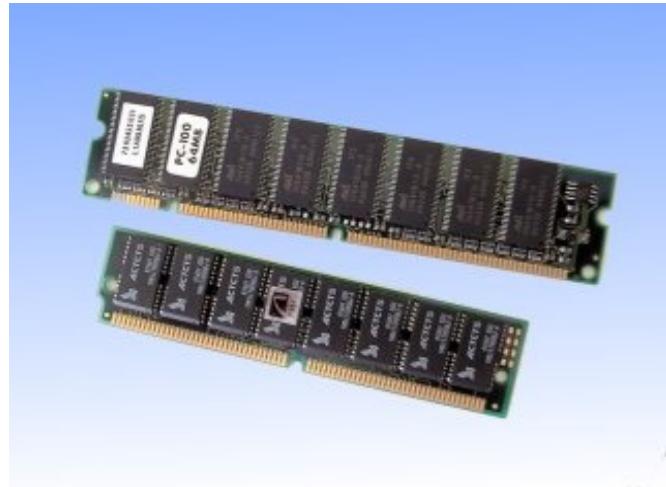
- DRAM (ang. Dynamic Random Access Memory) - ten rodzaj pamięci o dostępie swobodnym przechowuje każdy bit danych w oddzielnym kondensatorze. Ze względu na to, że nośnikiem informacji są kondensatory (a dokładniej tranzystory polowe Denarda), które samoistnie rozładowują się. Odświeżanie polega na odczytaniu i zapisaniu tej samej informacji. Ponadto odczyt z pamięci DRAM jest niszczący (informacja jest kasowana w wyniku rozładowania kondensatora przez współpracujący z nim tranzystor), dlatego też należy powtórnie zapisać odczytane dane tak, aby nie uległy zmianie. Zaletą tego typu pamięci jest natomiast niski koszt i możliwość uzyskania wysokiego stopnia upakowania (dużej ilości bitów pamięci na jednostkę powierzchni). Pamięci DRAM są podstawowym typem stosowanym jako pamięć operacyjna.
- SRAM (ang. Static Random Access Memory), przechowuje dane tak długo, jak długo włączone jest zasilanie, w odróżnieniu od pamięci typu DRAM, która wymaga okresowego odświeżania. Każdy bit przechowywany jest w pamięci SRAM w układzie zbudowanym z czterech tranzystorów, które tworzą przerzutnik, oraz z dwóch tranzystorów sterujących. Taka struktura umożliwia znacznie szybsze odczytanie bitu niż w pamięci typu DRAM, oraz w przeciwieństwie do pamięci DRAM nie wymaga odświeżania. Pamięci SRAM wykorzystywane są w szybkich pamięciach podręcznych cache wbudowanych w procesor lub instalowanych na płytach, ponieważ nie wymagają one dużych pojemności (gęstość danych w SRAM jest 4 razy mniejsza niż w DRAM), ale prędkość dostępu jest około 10 razy większa od DRAM

Ponieważ zazwyczaj pamięć cache jest wbudowana w procesor, natomiast pamięć operacyjna jest wybierana spośród dostępnych na rynku modułów - to w dalszej części lekcji skupimy się na typach pamięci DRAM.

FPM RAM

FPM RAM (ang. *Fast Page Mode RAM*) jest najstarszym rodzajem pamięci. Jest to pamięć typu asynchronicznego, w której sygnały sterujące matrycą komórek pamięci (sygnały te to: *RAS: Row Adress Strobe* - wybór wiersza matrycy pamięci oraz *CAS: Column Adress Strobe* - wybór kolumny matrycy pamięci) generowane są niezależnie od taktów zegara. Tak więc, informacja na wyjściu ukazuje się po czasie wynikającym z konstrukcji układu. Dodatkowo, występują problemy ze zsynchronizowaniem się taktów zegara systemowego i taktów pamięci

Termin "Fast" odnosi się do faktu, iż pamięci te umożliwiają szybszy dostęp do danych znajdujących się na jednej stronie pamięci. Niegdyś pamięci te były montowane "na pokładzie" 386-óstek i 486-óstek w postaci 72-pinowych modułów SIMM (ang. *Single In-line Memory Module*).



Porównanie pamięci typu SIMM i DIMM

SDRAM

Pamięć SDRAM (ang. *Synchronous DRAM*), podobnie jak pamięć typu FPM, jest pamięcią typu DRAM. Pamięć ta pracuje z częstotliwością zewnętrznej magistrali systemowej (a więc synchronicznie) i charakteryzuje się czasem dostępu rzędu 10 ns. SDRAM-y wyróżnia ponadto wysoka teoretyczna przepustowość danych - 800 MB/s dla kości typu PC-100 i 1064 MB/s dla PC-133. Pamięci SDRAM są wykonywane w postaci 168-pinowych modułów DIMM (ang. *Dual In-line Memory Module*), obecnie zasilanych napięciem 3,3 V.

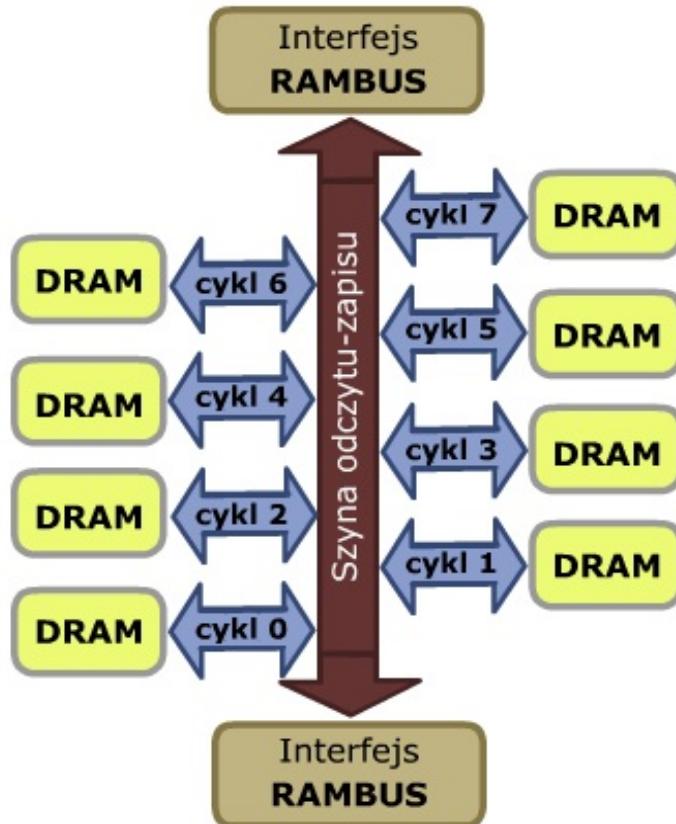
DDR SDRAM

Pamięć DDR SDRAM (ang. *Double Data Rate DRAM*) jest ulepszoną wersją swojej poprzedniczki. Ulepszenie polega na możliwości przesyłania danych na obydwu zboczach sygnału zegarowego. Charakteryzują się bardzo dużą przepustowością - 2.1 GB/s dla DDR SDRAM pracujących efektywnie z częstotliwością 266 MHz.

Podwojenie częstotliwości pracy nie jest jedyną nowinką związaną z pamięciami DDR. Pamięci tego typu posiadają usprawniony mechanizm synchronizacji oraz buforowania danych. Pamięć DDR wykonana jest w postaci 184-pinowych modułów DIMM.

RDRAM

Pamięci RDRAM (ang. *Rambus Direct RAM*) są układami podobnymi do pamięci DRAM. W układach tych matryca pamięci jest podzielona na 8 niezależnych bloków. Każda z części jest odczytywana z pewnym opóźnieniem, wynikającym z częstotliwości zegara. W jednym takcie zegara jest więc odczytana informacja z pojedynczego bloku. Przy kolejnych cyklach pobierane są kolejne dane i dopiero po odczytaniu wszystkich bloków, dane wysyłane są na zewnątrz w postaci pojedynczego pakietu.



Pobieranie danych z poszczególnych banków przez pamięci Rambus

Ważną zaletą pamięci RDRAM jest fakt, że umożliwiają one duże transfery - 1,6 GB/s dla pamięci współpracującej z 16-bitową szyną danych z częstotliwością 400 MHz (efektywnie 800 MHz - informacje przesyłane są na obu zboczach sygnału zegarowego). Podwojenie kanału Rambus (do którego wsparcie oferuje chipset i840) daje przepustowość 3,2 GB/s. Olbrzymia przepustowość pamięci Rambus spowodowała wydłużenie czasu dostępu do danych.

Z wytwarzaniem pamięci RDRAM wiąże się wiele problemów technologicznych, jak np. eliminacja efektu linii długiej, zakłócającego poprawność przesyłania informacji. Wymusza to rezygnację z możliwości zainstalowania więcej niż dwóch modułów pamięci RDRAM na płycie głównej. Pamięć RDRAM wytwarzana jest w postaci modułów RIMM (ang. *Rambus In-line Memory Module*).

DDR₂ RAM

DDR2-Double Data Rate 2 to układ pamięci RAM następca DDR. Zasada działania jest taka sama jak pierwowzoru. Natomiast wzrosła przepustowość. W chwili obecnej dostępne są pamięci DDR2 400(PC 3200), DDR2 533 (PC4200), DDR2 667(PC5300) DDR2 800 (PC6400).



Pobieranie danych z poszczególnych banków przez pamięci Rambus

Pamięci te przy odpowiednim parowaniu(dobraniu dwóch takich samych modułów) mogą pracować w trybie Dual-Channel. jest to zwiększenie szerokości z 64-bitowej na 128-bitową szyny dostępu do pamięci RAM. Obsługę taką zapewniają odpowiednie chipsety płyt głównych. Uzyskuje się teoretycznie zdwojoną przepustowość MB/s przy tej samej częstotliwości taktowania pamięci w stosunku do pojedynczego modułu o tej samej łącznej pojemności.

DDR₃ RAM

DDR3 jest kolejnym rozszerzeniem standardu DDR RAM. Obecne układy pamięci **DDR3** SDRAM są produkowane w procesie produkcyjnym 90 nanometrów, a standardowe napięcie zasilania wynosi dla nich 1,5 V. Pamięci **DDR2** wymagają standardowo 1,8 V. Moduły pamięci DDR3 mają 240 wyprowadzeń, czyli tyle samo ile DDR2. Jednak kontroler pamięci przystosowany do obsługi DDR2 nie potrafi obsługiwać DDR3, a kilka wyprowadzeń modułu DIMM zmieniło swoje znaczenie. Z tego powodu niemożliwe jest włożenie niezgodnych modułów – np. modułów DDR3 w złączach DIMM dla DDR2.

Ogólnie pamięć DDR3 jest bardziej wydajna niż DDR2, przy czym ze względów marketingowych na sprzedawanych w sklepach modułach podaje się szybkość sygnału DDR a nie częstotliwość zegara. Tak więc oferowano pamięci DDR - 266, 333 i 400 MHz, DDR2 - 533, 667, 800 i 1066 MHz a obecnie DDR3 - 1333 MHz i szybsze. Do oznaczenia standardu używa się natomiast jeszcze bardziej imponujących wartości podających przepustowość pamięci. Wyliczamy je według wzoru - (64 bity * 2 * częstotliwość pracy (MHz))/8. Tym samym pamięci DDR2 800 MHz to PC2-6400 (przepustowość 6400 MB/s), zaś DDR3 1600 MHz to PC3-12800.

Błędem byłoby przyjęcie, że teoretyczne uzyskiwane przez moduły RAM przepustowości są jednoznaczne z ich "wydajnością". Dzieje się tak za sprawą czasów dostępu (CL - CAS Latency) czyli czasu jaki mija między wysłaniem przez kontroler pamięci żądania dostępu do jej określonej kolumny a odczytaniem danych z tej kolumny. Im krótszy czas dostępu (niższa wartość CL), tym szybciej procesor uzyskuje dostęp do wymaganych, przechowywanych w pamięci danych. Uzyskanie wyższych częstotliwości taktowania odbywa się w kolejnych generacjach DDR kosztem wydłużenia czasów dostępu.

O ile w popularnych DDR standardowa wartość CL wynosiła 3, to w DDR2 wzrosła ona już do 5 podczas gdy w DDR3 wahala się w graniach od 7 do 9. Co za tym idzie w przypadku zaspokojenia zapotrzebowania procesora na przepustowość pamięci, o realnej wydajności systemu zadecydują właśnie czasy dostępu. Oznacza to, że w takim przypadku dalsze podnoszenie przepustowości nie przełoży się wzrostem wydajności.

Pamiętajmy też, że procesor nie posiada bezpośredniego połączenia z pamięcią RAM, ale komunikacja odbywa się za pomocą płyty głównej. Tak więc może ona stanowić (i stanowi) wąskie gardło w tym systemie wymiany danych. W przypadku systemów z procesorami Intel, komunikacja na płycie odbywa się poprzez dość prymitywną (co nie oznacza, że nieskuteczną) magistralę FSB. Kontroler pamięci w tym przypadku został zintegrowany w mostku północnym płyty. Inaczej pracują płyty dla procesorów AMD gdzie kontroler znajduje się w samym rdzeniu procesora zaś miejsce FSB, pisząc w wielkim skrócie, zajęło łącze HT o nieporównywalnie większych możliwościach.

PAMIĘCI MASOWE

DYSKI TWARDE

Dyski twarde HDD (ang. *Hard Disk Drive*) zostały tak nazwane z powodu swej sztywnej konstrukcji. Dyski twarde nie zawsze były takie "twarde". Kiedyś, przed przenoszeniem dysku z miejsca na miejsce, trzeba było zaparkować głowice, czyli uruchomić specjalny program, który zajmował się przemieszczeniem głowic poza obszar magnetyczny dysku.



Dysk twardy firmy Quantum

Dzisiaj dyski operacje takie wykonują automatycznie, ponadto są bardzo odporne na wstrząsy. Dyski twarde zawierają w swojej obudowie kilka, a nawet kilkanaście talerzy (standardowo 3 talerze magnetyczne). Talerze wirują prędkością 3600-7200 obrotów na minutę (ang. RPM - *Rounds Per Minute*), a niektóre dyski SCSI kręcą się z prędkością 15 000 RPM (250 obrotów na sekundę). Wewnątrz pyłoszczelnej obudowy dysku twardego znajdują się (oprócz głowicy i talerzy): układy sterowania silnikiem napędu dysków, silnikiem przesuwu głowic (służącym do pozycjonowania) oraz głowicami zapisu/odczytu, a także inne układy sterowania i kontroli. Dzięki dużej prędkości w ruchu obrotowym wytwarza się poduszka powietrzna pod głowicą zapisu/odczytu, dlatego łatwo może ona być utrzymywana w stałej odległości od talerza (głowica nie dotyka dysku podczas pracy!). Dzięki dużej prędkości obrotowej możliwe jest również uzyskiwanie dużych prędkości transmisji danych.

Najważniejszymi parametrami dysków twardych są:

- szybkość transmisji (transfer) danych;
- prędkość obrotowa (5400 RPM, 7200 RPM, 15 000 RPM);
- Średni czas dostępu (ang. *average access time*) wyrażany w ms (np. 10ms). Na tę wielkość składają się:
 - średni czas wymagany do umieszczenia głowic nad odpowiednim cylindrem (ang. *average seek time*);
 - opóźnienie rotacyjne związane z umieszczeniem głowicy nad wybranym sektorem;
- pojemność (na dzień dzisiejszy sięgająca terabajtów danych);

Należy tu zauważyć, że prędkość dysku zależy także od rodzaju złącza (interfejsu).

Na dzień dzisiejszy standardów jest kilka, przy czym najważniejsze z nich to:

SATA, SATA-II, SATA-III – to obowiązujący od kilku lat standard w przypadku komputerów osobistych. SATA (ang. *Serial Advanced Technology Attachment*) jest szeregowym interfejsem umożliwiającym podłączanie różnorakich urządzeń zewnętrznych. Pierwsza wersja oferuje przepustowość do 1,5 Gbit/s (ok 179 MB/s), druga 3 GBit/s, trzecia 6 GBit/s. Ponadto w tym standardzie istnieje także kilka typów złączy. Oprócz klasycznego wtyku 7-pinowego, możliwe jest stosowanie standardu eSATA do podłączania dysków na zewnątrz obudowy komputera, przy ograniczeniu długości kabla do ok. 2 metrów. Rozwinięcie standardu eSATA to xSATA - pozwala na stosowanie kabli do 8 metrów długości. Inne zastosowanie ma złącze mSATA, opracowane głównie w celu miniaturyzacji samej wtyczki, i wykorzystywane w rozwiązańach przenośnych (małe dyski, netbook-i, itp.).

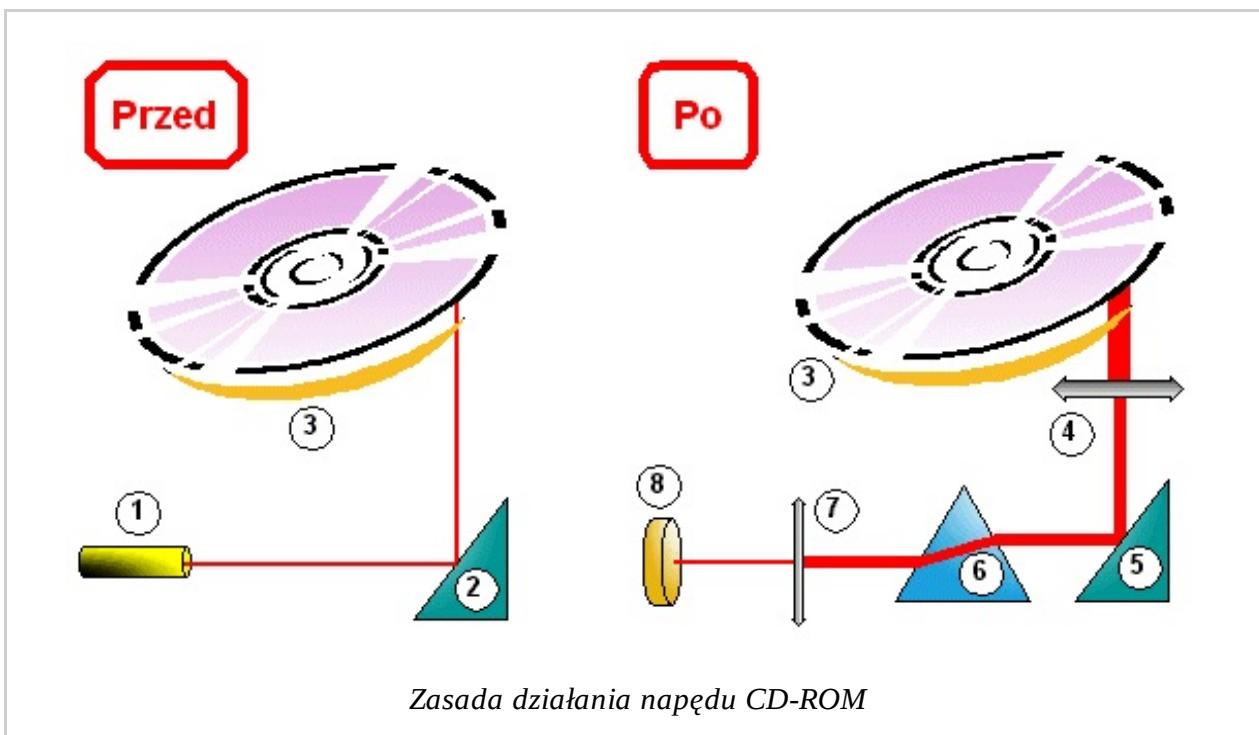
SAS – Serial Attached SCSI – następca standardu SCSI. Używany głównie w dyskach serwerowych. Fizycznie SAS jest kompatybilny z SATA. Dyski SATA można podłączać do kontrolera SAS, ale zgodności w drugą stronę już nie ma. Osiągane prędkości transferu to 12 Gbit / s.

NAPĘDY OPTYCZNE

Prace nad napędem CD-ROM rozpoczęte zostały w roku 1978. Pierwszy napęd powstał dzięki firmie Phillips, ale odtwarzacz CD-ROM, jaki znamy w dzisiejszej postaci zaprojektowany został przez team (ang. "team") firm Phillips i Sony. W połowie lat 80 minionego wieku, napędy te weszły do szerszej dystrybucji, zaś lata 90 przyniosły im niezwykłą popularność głównie dzięki stosunkowo niskiej cenie oraz dużej pojemności jaką oferowały płyty CD-ROM.

Zasada działania napędu CD-ROM jest prosta (patrz rysunek poniżej). Otóż:

1. *Dioda laserowa* (1) emituje na lustro (2) promień podczerwony o niskiej energii.
2. *Silnik servo*, przesuwając lustro (2) na rozkaz mikroprocesora, ustawia promień na właściwej ścieżce płyty (3).



Zasada działania napędu CD-ROM

3. Po odbiciu promienia przez płytę (3), jego światło jest zbierane i skupiane przez pierwszą soczewkę (4) znajdującą się pod płytą, odbijane od lustra (5) i kierowane do układu pryzmatów (6).

4. *Układ pryzmatów* (6) kieruje promień na następną soczewkę skupiającą (7).
5. Druga soczewka kieruje promień światła na *fotodetektor* (8), który zamienia go na ciąg impulsów elektrycznych.
6. Impulsy elektryczne są dekodowane przez mikroprocesor i wysyłane do komputera.

Zagłębiania wytrawione na płycie CD-ROM (ang. *pit*) sprawiają, że światło odbite od pit'a różni się intensywnością (energią) od tego, które na pit'a nie trafiło. Zmiana energii wiązki lasera traktowana jest więc jako pojawienie się informacji i przekształcana przez fotodetektor na impulsy elektryczne.

Ponieważ przekłamanie informacji o pojedynczym bicie może być groźne np. dla działania programu typu *.exe, stosuje się zaawansowane techniki wykrywania i korekcji błędów. Dzięki nim prawdopodobieństwo niewykrycia błędu jest rzędu 10^{-25} . Sama korekcja wymaga zaś 288 B na każde 2048 B danych.

Najważniejszymi parametrami napędów CD-ROM są :

- *Szybkość transmisji* - określa jak szybko może odbywać się komunikacja pomiędzy jednostką centralną a urządzeniem. Dla napędów CD-ROM przyjęło się używać oznaczeń 1x, 2x, 8x, itp. Są to oznaczenia właśnie szybkości transmisji. Symbol 1x określa przepływ danych przez 75 sektorów danych po 2048 B w ciągu sekundy (format CD), co daje szybkość 150 kB/s. Naturalnie symbole 2x, 4x, itd. są wielokrotnościami wymienionej prędkości.
- *Czas dostępu* (ang. *access time*) - opóźnienie pomiędzy momentem zlecenia odczytu, a chwilą uzyskania danych (typowa wartość 100 ms). Należy pamiętać, że inny jest czas dostępu do danych położonych blisko środka, a inny przy brzegu płyty.

Napęd CD-ROM można podłączyć do komputera korzystając z interfejsów, z których najpopularniejsze to SCSI/ASPI oraz IDE/ATAPI.

DVD

Skrót DVD *Digital Versatile Disk* - dysk zawierający różne dane. W domyślnym znaczeniu chodzi o dane cyfrowe, przede wszystkim cyfrowo zapisane multimedia. Obecnie jest najpopularniejszym nośnikiem optycznym. Pojedynczy dysk DVD jest w stanie pomieścić nawet do 15GB, oczywiście jeśli to będzie płytka dwuwarstwowa dwustronna. Najczęściej jednak dysk ten jest jednowarstwowy jednostronny o pojemności 4.7GB. Podobnie jak w przypadku płytki CD podawane są pojemności w minutach - w takim przypadku pojemność obliczana jest dla filmu i ścieżki dźwiękowej o bardzo dobrej jakości - dla płytki jednowarstwowej (4.7GB) to 133 minuty. Wymiarami zewnętrznymi płytka DVD nie różni się od CD. Grubość 1.2 mm, średnica 120 mm (mini 80mm). Podobnie są również pod względem kolejności i układu warstw. Laser wykorzystywany do pracy w tym napędzie pracuje wykorzystując światło o długości 650nm

Cieńsza jest natomiast warstwa z poliwęglanu. Związane jest to z technologia wykonywania nośników dwustronnych. Są one składane „plecami” do siebie i wymiar grubości 1.2 mm zostaje przez to zachowany.

FORMATY DVD

DVD-ROM - CZYLI PŁYTKA TYLKO DO ODCZYTU

Może ona zawierać różne dane, trzy najpopularniejsze formaty to DVD-ROM (dane komputerowe bez determinowania ich treści), DVD-VIDEO (materiał video w formacie akceptowalnym przez każdy odtwarzacz video) oraz DVD-AUDIO (najmniej popularny z wymienionych - zawiera muzykę o jakości

wyższej niż CD)

DVD-R

Płytki jednorazowego zapisu. Wyróżnia się dwa, niezgodne ze sobą, formaty: -R oraz +R (na szczęście aktualnie praktycznie wszystkie napędy DVD-RW umożliwiają odczyt - zapis w nich obu). Pierwszy zapisywany typ nośnika w technologii DVD. Stanowią odpowiednik płytki CD-R. Obecnie prędkość zapisu takich płyt osiąga wartość 18x

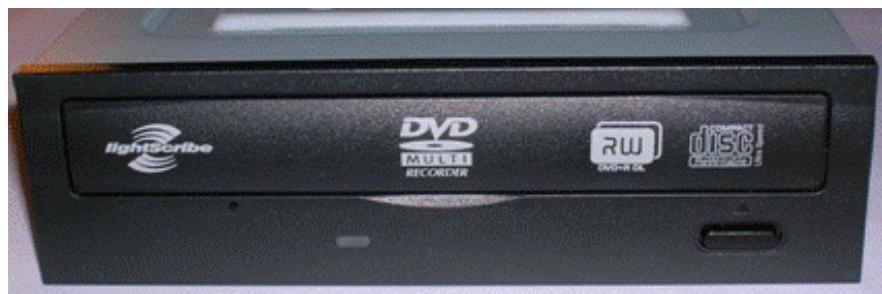
DVD-RW

Płytki wielokrotnego zapisu. Istnieją dwa formaty podobnie jak poprzednie. DVD-RW /+RW . Płytki takie mogą być nagrywane do ok. 1000 razy (oczywiście rozwiązań firmowych, i te zapisy muszą być dokonane w krótkim czasie).

DVD-RAM

Płytki ta po pierwszym formatowaniu służy jak dysk twardy. Dane odczytywane jak i zapisywane na niej są w czasie rzeczywistym. Standard ten popularny jest Japonii, w innych rejonach świata przegrał z konkurencyjnym DVD-RW. Cechuje go wysoka niezawodność i bezpieczeństwo danych. Trwałość tego medium szacowana jest na ok. 100 000 cykli. Pierwsza generacja jednostronnej płytki miała 2.58 GB potem doprowadzono ją do typowej pojemności 4.7 GB. Doskonale sprawdzają się tam gdzie stosowano do tej pory napędy taśmowe oraz napędy MO. Stosowany system plików to UDF lub FAT32. **Waga** - płytki DVD RAM nie da się odczytać w standardowym napędzie DVD

Nowością wprowadzoną przy uatrakcyjnianiu nagrywarek DVD jest technologia **LightScribe** - jest to możliwość tworzenia za pomocą lasera nagrywarki etykiet na dyskach CD i DVD. Obraz powstaje na górnej stronie płytki. Wadą tej technologii jest stosunkowo długi czas wypalania.



Typowa nagrywarka DVD z funkcją LightScribe

Praktyczne stosowanie technologii LightScribe wymaga również stosowania przeznaczonych do tego płyt i oprogramowania.

HD DVD

Pierwotna nazwa AOD (Advanced Optical Disk) - standard opracowywany wspólnie przez NEC i Toshiba. Dysk HD DVD jest bardzo zbliżony do poprzednika-DVD. Płytki są o średnicy 120mm składają się z dwóch warstw o grubości 0.6 mm każda. Płytki mogą być jednowarstwowe, dwuwarstwowe, jednostronne lub dwustronne. Zasadniczą różnicą jaka została wprowadzona to

zastosowanie światła laserowego o długości 405nm - koloru niebieskiego, a mniejsza długość światła oznacza możliwość gęstszego upakowania danych (zmniejsza się rozmiar najmniejszego możliwego do odróżnienia punktu). Prosta zmiana długości światła pozwoliła uzyskać 15GB na jedną warstwę tłoczoną - dla warstwy nagrywanej do 20GB. Standard przewiduje stosowanie jednego systemu plików - UDF. Napędy HD-DVD zachowują wstępna zgodność - większość z nich pozwala na odczyt płyt poprzednich generacji (DVD i CD). W format HD-DVD mogą być wbudowane zabezpieczenia przed kopiowaniem i nieautoryzowanym dostępem.

Pierwszą nagrywarką komputerową HD DVD jest Toshiba SD-903A. Wypala dyski jedno i dwuwarstwowe z prędkością 1x. To daje czas 30GB w ok. 2 godz.

BLUE **R**AY

Nowoczesny nośnik danych oraz napęd nad którym prace prowadzą firmy Sony, Philips Hitachi, Sharp, Samsung, TDK. Jest to konkurencyjna technologia wobec HD DVD. Płytki Blue Ray są w stanie pomieścić w jednej warstwie 25GB danych. Zatem dla płytki dwuwarstwowej to już 50GB danych. Osiągnięcie tak ogromnej pojemności na krążku wymagało wprowadzenia zupełnie nowej technologii. Skonstruowano układ optyczny który wypala zmniejszone pole (pit) w warstwie światłoczułej płytki. Płytnka w stosunku do poprzednich rozwiązań charakteryzuje się następującymi cechami:

- zmniejszona grubość warstwy ochronnej do 0,1 mm . Przy czym została ona pokryta materiałem odpornym na zabrudzenia.
- zmieniona apertura numeryczna soczewki do wartości 0,85.

Długość wiązki zastosowanego lasera ma wartość 405 nm. Jest to kolor niebieski- stąd to właśnie pochodzi nazwa całego standardu- Blue-ray. Ponadto moc lasera potrzebna do nagrania zmniejszona została o jedną czwartą w porównaniu do potrzebnej do nagrania w standardzie DVD.

Dyski BD postanowiono w związku ze zwiększoną wrażliwością na zabrudzenia np. z odcisków palców, pakować w kasety.

Napęd Blue-ray powszechnie zastosowano w SONY PlayStation 3.

PAMIĘCI **F**LASH PODŁĄCZANE POD PORT **USB**

Jest wiele różnych nazw do określenia tego rodzaju urządzenia. Najczęściej można spotkać się z określeniem:

- PenDdrive
- Memory Stick
- USB Flash Drive



Typowa pamięć pendrive

Wspólną cechą wszystkich urządzeń typu PenDrive jest obecność złącza USB. Ma dwojaką rolę, zasilania i interfejsu dla danych. Pamięci tego typu charakteryzują się dobrymi parametrami. Brakiem części ruchomych a przez to dużą trwałością urządzenia. Średni czas pracy obliczono na 10 lat. Ścisłej zależy on od liczby dokonywanych zapisów i odczytów. Jedną z najważniejszych cech jest szybkość zapisu i szybkość odczytu. Nowoczesne pamięci mają średni transfer rzędu kilkudziesięciu MB/s. Przy czym należy zauważyć ze transfer ten zależy od jakości zastosowanej pamięci, wersji łącza USB aż po wielkości pliku kopiowanego. Urządzenia wyprodukowane na złączu USB Low Speed (USB v.1.0) to zaledwie 1.5Mb/s.



Przykład popularnego PenDrive'a

Przypadku Urządzeń z określeniem USB 2.0 Full Speed to 12 Mb/s. Najszybsze to urządzenia z nazwą Hi-Speed to maksymalnie 480Mb/s. Przy kopiowaniu małej liczby plików prędkość kopowania ma drugorzędne znaczenie. Im plik większy tym transfer kopowania wyższy.

Pamięci PenDrive mają szerokie zastosowania. Od zwykłego zastępowania zwykłej dyskietki podczas przenoszenia plików z jednego komputera do innego, aż po wyrafinowane urządzenia mogące przechowywać dane w zaszyfrowanej postaci, czy też po napędy symulujące dysk twardy komputera. Stosowane są również w połączeniu z odtwarzaczami mp3, mp4.

Dyski SSD

Jak już wspomnieliśmy w poprzednim rozdziale, w tym przypadku nazwa jest nieco myląca. Dysk SSD w zasadzie nie jest żadnym dyskiem, w skrócie można powiedzieć, że jest to urządzenie pamięci masowej, w którym rolę nośnika informacji pełnią półprzewodnikowe pamięci flash - składa się wyłącznie z układów pamięci flash oraz elektroniki pomocniczej. Klasyczne dyski twarde to skomplikowane mechanicznie urządzenia, co czyni je bardzo wrażliwymi na urazy mechaniczne. Jeśli dysk twardy spadnie ze stołu, jedyne co będziemy mogli z nim zrobić, to zamieść szczątki do śmieci. W takiej samej sytuacji dysk SSD prawdopodobnie nie ulegnie żadnemu uszkodzeniu. Z tej zalety korzystają przede wszystkim użytkownicy komputerów przenośnych. To głównie tam dyski twarde narażone są na urazy mechaniczne. Brak elementów ruchomych przekłada się nie tylko na wytrzymałość mechaniczną dysków SSD, ale także na ich w zasadzie bezgłośną pracę. No i trzecia zaleta - wysoka sprawność energetyczna. Dysk SSD zazwyczaj pobiera podczas pracy ok 1 W, co jest wartością dziesięć razy niższą niż typowy pobór prądu dysku HDD. Do innych zalet należy brak nagrzewania się, niska waga i potencjalnie małe rozmiary, czy też wysoka odporność na zakłócenia elektromagnetyczne.

Z punktu widzenia bardziej "informatycznego" dyski SSD różnią się diametralnie od rozwiązań mechanicznych w jednym parametrze - czasie dostępu. W praktyce taki dysk jest pamięcią o dostępie swobodnym - stałym dla każdego zapisanego bitu. Nie bez znaczenia jest też wartość czasu dostępu - co najmniej rzad wielkości niższy niż w przypadku klasycznych rozwiązań.

Innym parametrem jest czas odczytu i zapisu danych. I tu ujawniają się pierwsze wady pamięci półprzewodnikowych. W ich przypadku występuje wyraźna asymetria w przypadku obu tych czasów - zapis jest zawsze wielokrotnie wolniejszy niż odczyt, co więcej - odczyt odbywa się szybciej niż w klasycznych dyskach, lecz zapis już niestety jest wolniejszy. Z zapisem na dysk jest związana jeszcze jedna wada urządzeń półprzewodnikowych - ograniczona liczba cykli (ilości zapisów pojedynczej komórki).

Oba wspomniane parametry (szybkość zapisu i ilość cykli) różnią się znaczco w przypadku dwóch technologii budowania pamięci RAM - MLC (ang. Multi Level Cell) oraz SLC (ang. Single Level Cell). Pamięci typu MLC są tańsze w produkcji oraz mają większą pojemność, natomiast SLC są i droższe, i mają mniejsze upakowanie informacji (mniejszą pojemność), w zamian wytrzymując 10x więcej cykli zapisów, na dodatek ze znacznie większą prędkością.

Dyski SSD podłącza się do jednostek je wykorzystujących za pomocą tych samych interfejsów co HDD.

KARTY GRAFICZNE

Podstawowym zadaniem karty graficznej jest przechowywanie informacji o tym jak powinien wyglądać obraz na ekranie monitora i odpowiednim sterowaniu monitorem. Pierwsze karty graficzne potrafiły jedynie wyświetlać znaki alfabetu łacińskiego ze zdefiniowanego w pamięci karty generatora znaków - tryb tekstowy. Kolejna generacja kart graficznych potrafiła już wyświetlać w odpowiednim kolorze poszczególne punkty (piksele) - tryb graficzny. Nowoczesne procesory graficzne udostępniają wiele funkcji ułatwiających i przyśpieszających pracę programów. Możliwe jest narysowanie odcinka, trójkąta, wieloboku, wypełnienie ich zadanym kolorem lub wzorem, tzw. akceleracja 2D. Większość kart na rynku posiada również wbudowane funkcje ułatwiające tworzenie obrazu przestrzeni trójwymiarowej, tzw. akceleracja 3D. Niektóre posiadają zaawansowane algorytmy potrafiące na przykład wybrać tylko widoczne na ekranie elementy z przestrzeni.

Przegląd kart graficznych rozpoczęmy od rozwiązań w zasadzie już historycznych:

MDA

Prekursorem kart graficznych, instalowanych w komputerach rodziny PC, był sterownik, który pojawił się w komputerach IBM PC w roku 1981, o nazwie **MDA** (ang. *Monochrome Display Adapter*). Karta pracowała tylko w trybie tekstowym o rozdzielczości 25 linii x 80 znaków [2]. Wyposażona w 4KB pamięci oferowała częstotliwość odchylania pionowego 50 Hz.

CGA

Następcą sterownika MDA była karta **CGA** (ang. *Color Graphics Adapter*) - opracowana przez IBM w 1982 roku karta jako pierwsza oferowała możliwość korzystania z trybu graficznego. Były to jednak tylko dwie rozdzielczości: 640x200 oraz 320x200 punktów. Karta oferowała 16 KB pamięci - większa rozdzielcość wyświetlana była jedynie w trybie monochromatycznym, zaś niższa "aż" w 4 kolorach. Tryb tekstowy możliwy był również w dwóch wariantach: 80 znaków x 25 linii, bądź 40 znaków x 25 linii, niestety matryca znaku miała rozmiary 8x8 pikseli. Karta oferowała maksymalną częstotliwość odświeżania pionowego 60 Hz.

Karta wykorzystywała spakowaną (ang. *packed*) metodę odwzorowania pamięci - w danym bloku pamięci RAM każdemu pikselowi obrazu odpowiadał fragment bajtu, zawierający numer koloru tego punktu (np. 1 bit - 2 kolory, 2 bity - 4 kolory, itd.).

HERKULES

Karta Hercules pojawiła się w tym samym czasie co karta CGA. Była ona wyposażona dodatkowo w złącze równoległe, umożliwiające podłączenie drukarki.

Karta oferowała możliwość pracy w rozdzielczości 720x348 punktów (zarówno w trybie tekstowym, jak i graficznym), ale jedynie w trybie monochromatycznym. Wyposażona była w 64 KB pamięci. Znaki w trybie tekstowym wyświetlane były na podstawie matrycy 9x14 punktów. Karta nie miała możliwości współpracy z IBM-BIOS, gdyż nie została wyprodukowana przez IBM.

Aby umożliwić szybszy dostęp do danych pamięć została podzielona była na dwie strony graficzne, natomiast każda ze stron - na cztery banki.

EGA

Karta EGA (ang. *Enhanced Graphics Adapter*) to kolejny etap rozwoju CGA. Karta oferowała wyświetlanie obrazu w rozdzielcości 640x350 punktów przy 16 kolorach (wybieranych z palety 64 kolorów). Zaopatrzona była w 256 KB pamięci. Rozdzielcość w trybie tekstowym wynosiła 80x43, przy matrycy znaku 8x14.

Sterownik EGA składał się z czterech głównych bloków funkcjonalnych [1]:

- *Układ sekwencyjny* - generuje sygnał zegarowy; przesyła dane pomiędzy pamięcią obrazu, układem graficznym i układem określania atrybutu; odpowiada za wybór lokalizacji wyświetlanych znaków.
- *Układ graficzny* - przekazuje dane pomiędzy pamięcią obrazu, układem graficznym i układem określania atrybutu.
- *Układ sterowania atrybutem* - służy do zmiany kolorów zapisanych w pamięci obrazu na indeksy kolorów zdefiniowanych w rejestrach wzorców kolorów.
- *Układ sterowania wyświetlaczem* - odpowiada za zachowanie zależności czasowych podczas wyświetlania obrazu oraz wyświetla kurSOR.

Pamięć wideo opisywanej karty podzielona jest na cztery 64KB obszary (rozwiązań to jest wykorzystywane również w kartach VGA). Trzy kolory podstawowe (RGB) przyporządkowane są do kolejnych obszarów, zaś czwarty z obszarów zawiera informacje o intensywności z jaką ma być wyświetlony dany kolor. Tak więc jeden piksel zawiera swoje składowe w czterech blokach pamięci. Dzięki takiej strukturze 256 KB zajmuje 64 KB przestrzeni adresowej. Rozwiązań to nosi nazwę metody płatowej (ang. *planar, bit maped*), a jego wadą jest to, że utrudniony zostaje dostęp do danych.

VGA

Karta VGA (ang. *Video Graphics Array*) to kolejny standard firmy IBM, opracowany z myślą o aplikacjach graficznych. Sterownik ten jest w stanie emulować wszystkie dotychczas opisane standardy.

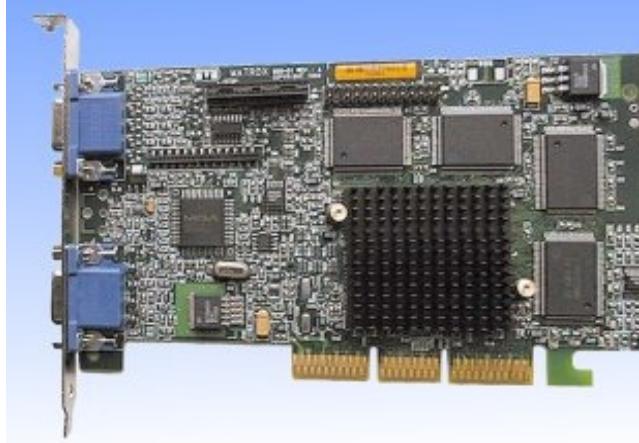
Cechą wyróżniającą kartę VGA jest fakt, że wysyła ona do monitora sygnały analogowe (poprzedniczki operowały na sygnałach cyfrowych), dzięki czemu zwiększo ilość wyświetlanych kolorów. Zajmuje się tym wyspecjalizowany układ przetwornika cyfrowo-analogowego (DAC - ang. *Digital Analog Converter*), który jest w stanie przedstawić każdą z barw w postaci określonej liczby poziomów (np. 64). Standardowy sterownik VGA umożliwiał wyświetlanie 25 wierszy znaków w 80 kolumnach (matryca znaku 9x16). Znak może być wyświetlany w jednym z 16 kolorów, natomiast kolor tła dla każdego znaku może być inny.

W trybie graficznym karta VGA umożliwia wyświetlenie obrazu o rozmiarach 640x480 punktów przy 16 kolorach (wybranych z palety 256 kolorów). Maksymalną liczbę kolorów - 256 - osiągnąć można przy rozdzielcości 320x200 punktów.

SVGA

Karty SVGA (ang. *Super VGA*) są rozszerzeniem techniki VGA. Sterowniki SVGA wykorzystują tzw. technikę stronicowania, polegającą na kojarzeniu z niewielkim obszarem pamięci (oknem), w przestrzeni adresowej, różnych fragmentów większego obszaru pamięci (stron lub banków). Tak więc, zapis/odczyt adresu położonego wewnątrz okna, powoduje zapis/odczyt odpowiadającego mu bajtu w banku. Rozmiar banku i okna wynosi zwykle 64 KB. Aby dostać się do pamięci spoza bieżącego banku, należy zmienić zawartość rejestru sterującego położeniem banku (ang. *Bank Start Adress*). Wszystko to po to, aby efektywniej wykorzystać 128 KB przestrzeni adresowej, którą ma

zarezerwowaną procesor na pamięć obrazu.



Karta graficzna SVGA firmy Matrox

Dzięki takim, i wielu innym innowacjom, możliwe jest korzystanie z dużej pojemnościowo pamięci, co umożliwiało kartom graficznym osiąganie rozdzielczości 1280x1024 i wyższych, przy palecie kolorów 16.7 mln. (true color).

Rozwiązania współczesne

Współczesna karta graficzna to wysoko-wydajnościowy system mikroprocesorowy, zwykle składający się z pięciu komponentów: interfejsu systemowego, pamięci graficznej, procesora graficznego GPU (Graphics Processing Unit), bufora ramki i tzw. RAMDAC (Random Access Digital/Analog Converter).

Interfejs systemowy znajduje się najbliżej płyty głównej. Obecnie stosuje się złącze PCI Express. Za jego pośrednictwem nieobrobione dane są ładowane do pamięci graficznej, w której przechowywane są wszelkie obiekty graficzne i tekstury. Jej pojemność waha się od 256 do nawet ponad 2 GB. Informacje są następnie odczytywane przez procesor graficzny, który przygotowuje do wyświetlenia na ekranie pozycje, ruchy i fakturę wszelkich obiektów widocznych w trójwymiarowej scenie. Po obróbce gotowy obraz jest umieszczany w buforze klatek i wędruje do RAMDAC – układu, który przetwarza cyfrowe obrazy, nadając im formę analogową, odpowiednią dla wyjścia VGA, lub przesyła je do cyfrowych wyjść DVI, HDMI lub DisplayPort. W przypadku większości wyjść cyfrowych istnieje możliwość prostej konwersji sygnału poprzez zakupienie odpowiedniego kabla. Wyjście DVI jest wyjściem mieszanym - to znaczy oprócz sygnałów cyfrowych istnieje także wyprowadzenie sygnałów analogowych, dzięki czemu po zastosowaniu prostej przejściówki można kartę podłączyć do odbiornika obsługującego analogowy standard VGA.



Współczesny procesor graficzny - jak widzicie potrzebuje on chłodzenia równie wydajnego jak procesor główny

Centralnym elementem każdej karty graficznej jest procesor graficzny (ang. Graphics Processing Unit, GPU). Jako pierwsza tego terminu użyła firma NVIDIA wprowadzając na rynek karty graficzne GeForce 256. To przełomowe wydarzenie dla akceleracji grafiki w systemach domowych miało miejsce 31 sierpnia 1999 roku, wcześniej systemy takie były dostarczane wyłącznie jako specjalizowane karty profesjonalne. Głównym zadaniem GPU było wykonywanie obliczeń potrzebnych do uzyskania akcelerowanej grafiki 3D, co spowodowało odciążenie procesora CPU z konieczności wykonywania tego zadania. W tej sytuacji mógł on zająć się innymi obliczeniami, co skutkowało zwiększeniem wydajności komputera podczas renderowania grafiki. Nowoczesne procesory graficzne wyposażone są w szereg instrukcji, których nie posiada procesor komputera, i w całości są wielkimi procesorami typu SIMD. Ponadto są wyposażone w dodatkowe bloki specjalizowane, np do kompresji / dekompresji obrazu. W tym przypadku w tym przypadku rola procesora w komputerze ograniczona jest do dostarczenia danych, a pracochłonnym dekodowaniem i wyświetleniem filmu zajmuje się procesor karty graficznej. Współczesne układy GPU radzą sobie także z konwersją wideo, czyli zmianą formatu filmu. Prawdziwym wyzwaniem dla współczesnych kart są jednak wymagające gry komputerowe.

W przypadku gier zadania karty składają się na tzw renderowanie obrazu, czyli przekształcenie wektorowego, uproszczonego opisu sceny na jej obraz widziany z konkretnego punktu. Zazwyczaj te zadanie jest wykonywane dwuetapowo:

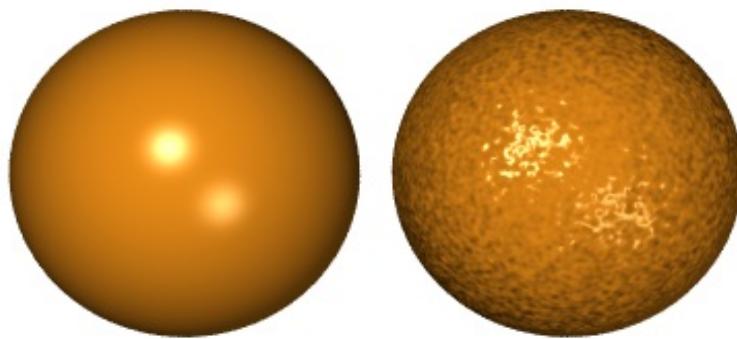
- W pierwszym etapie wykorzystywane są tzw. shadery, w grafice trójwymiarowej odpowiadające za cieniowanie obiektów. Shader ozwala na dużo bardziej skomplikowane modelowanie oświetlenia i materiału na obiekcie niż standardowe modele oświetlenia i teksturowanie. Jest jednak dużo bardziej wymagający obliczeniowo i dlatego dopiero od kilku lat sprzętowa obsługa shaderów jest obecna w kartach graficznych dla komputerów domowych. Shader jest niewielkim procesorem istniejącym w ramach GPU. Każda karta graficzna ma kilka, kilkanaście, na nawet kilkaset shaderów w różnych wariantach. Jeśli monitor ma na przykład wyświetlacz pędzący samochód wyścigowy, to odpowiada za to Vertex Shader, który błyskawicznie oblicza nową pozycję pojazdu w każdym obrazie i dodatkowo dba o właściwe oświetlenie obiektu, kiedy w scenie występuje wiele źródeł światła. Pixel Shader z kolei odpowiada za zmiany powierzchni i koloru obiektu. Kiedy w danej scenie samochód szybko się zbliża, uaktywnia się natomiast Geometry Shader, który błyskawicznie zaopatruje pojazd w nowe krawędzie i narożniki. O ile samochód jest wyświetlany jako obiekt o niewielkich rozmiarach, daleko w tle, to do jego

wyświetlenia wystarczy praktycznie sam obrys. Jeśli jednak jego obraz wypełnia cały ekran, to realistyczny wygląd może mu nadać tylko mnóstwo drobnych szczegółów. W stosunku do standardowych modeli oświetlenia, stosowanych do generowania grafiki w czasie rzeczywistym, shadery dają możliwość uwzględnienia między innymi:

- refrakcji
- odbić lustrzanych
- mapy przemieszczeń
- kiedy shadery wykonają swoją pracę, to obiekt musi zostać "oklejony" kolorami i tak zwanymi teksturami, za co odpowiada jednostka teksturowująca. Procesor graficzny zapisuje w pamięci karty graficznej możliwie najbardziej kompletne tekstury już na początku sceny.

Do najważniejszych funkcji współczesnych akceleratorów graficznych można zaliczyć:

- **Anisotropic Filtering** (Filtrowanie anizotropowe) - jest to technika wyostrzania tekstur (czyli rysunków jakimi pokryte są ściany obiektów) w trójwymiarowej grafice komputerowej, które znajdują się w dalszej odległości od kamery (lub postaci sterowanej przez gracza).
- **Bump Mapping** (mapowanie wypukłości) – w grafice 3D technika teksturowania, która symuluje niewielkie wypukłości powierzchni, bez ingerencji w geometrię obiektu trójwymiarowego.



Kula prezentowana bez (po lewo) i z (po prawo) mapowaniem wypukłości

- **HDR** (High Dynamic Range) - technologia generowania sceny w grafice trójwymiarowej, której efektem jest renderowanie świata z realistycznym oświetleniem. Największa różnica dostrzegana jest w bardzo ciemnych lub bardzo jasnych fragmentach sceny, gdzie symulowane jest natężenie światła wykraczające poza zakres możliwy do osiągnięcia na ekranie monitora (np. efekt oślepienia po spojrzeniu na słońce).
- **Shader**

Poza tym karty graficzne oferują inne sprzętowe efekty, jak mgła, przezroczystość (dodatkowy kanał Alpha).

Możliwości GPU nie ograniczają się jedynie do grafiki 3D. We współczesnych procesorach graficznych umieszczone elementy ALU mogące pracować jako potężne jednostki przeliczania równoległego. W niektórych zastosowaniach, np. symulacjach finansowych, karty graficzne uzyskują wyniki nawet 150 razy szybciej niż procesory CPU. Pozwalają na to shadery dające się dowolnie programować. Czynią one z chipów graficznych tzw. GPU ogólnego przeznaczenia (GPGPU, General Purpose GPU). Tutaj prekursorem znów była firma nVidia, która jako pierwsza udostępniła biblioteki do programowania

swoich kart graficznych, znane ogólnie pod nazwą CUDA (Compute Unified Device Architecture), której elementem jest środowisko programistyczne języków C i C++. Umożliwia ona tworzenie podzielnych na wątki aplikacji i ich wykonywanie przez jednostki GPU. ATI rozwija podobny projekt o nazwie CTM (Close to the Metal), pozbawiony jednak wygodnego środowiska C++.

W testach syntetycznych okazuje się, że wydajność GPU, dzięki masowej równoległości przetwarzania (kilka set programowalnych "procesorków" - shaderów w jednym GPU pracuje równolegle) przewyższa o rząd wielkości wydajność najlepszych procesorów.

Aktualnie na rynku liczą się jedynie dwaj producenci procesorów graficznych - nVidia oraz AMD, który przejął firmę ATI. Znów - podobnie jak w przypadku procesorów - opis konkretnych nowych rozwiązań w podręczniku jest z góry skazany na permanentną nieaktualność, więc po informacje o najnowszych rozwiązaaniach odsyłamy Was do internetu.

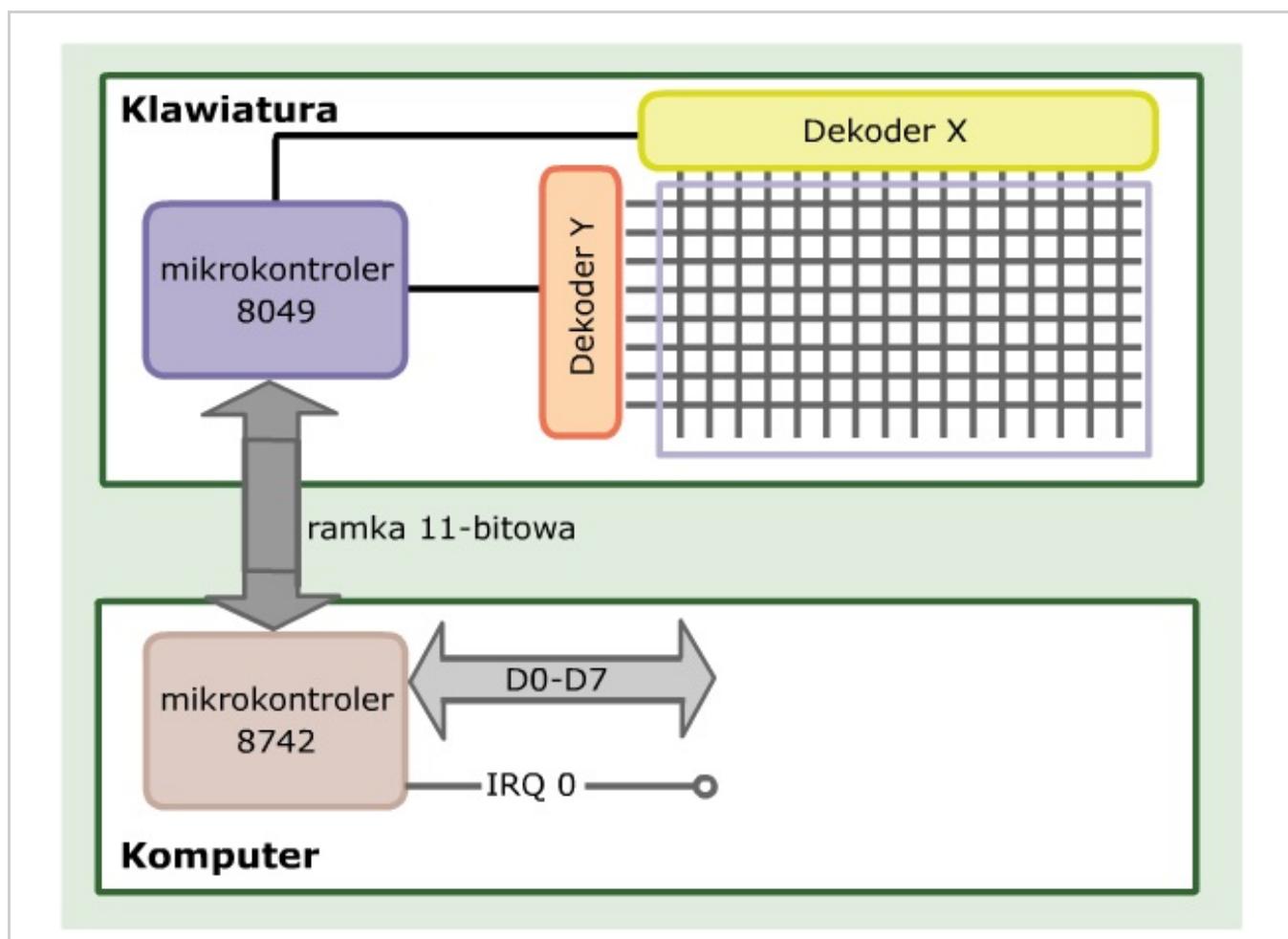
URZĄDZENIA WEJŚCIOWE

KLAWIATURA

Wynalezienie klawiatury było dla komputera było tym, czym był pierwszy lot dla kosmonautyki - biorąc pod uwagę fakt, że wcześniej wprowadzanie danych do komputerów było możliwe jedynie przy pomocy kart dziurkowanych. Historycznie klawiatury podzielić można na trzy grupy: model XT, model AT, i współczesna klawiatura 101, 102 lub 104- klawiszowa PS/2 (wprowadzona na rynek przez firmę IBM).

Klawiaturę XT i AT charakteryzowała mniejsza liczba klawiszy (83), inne ich rozmieszczenie, ponadto klawiatura XT zapewniała jedynie jednokierunkową wymianę informacji pomiędzy klawiaturą a komputerem.

Zasada działania klawiatury jest stosunkowo prosta. Klawiatura posiada swój mikrokontroler jednoukładowy, który nadzoruje siatkę połączeń linii X i kolumn Y, w węzłach której umieszczone są poszczególne klawisze. Układ wysyła, z odpowiednią częstotliwością, impulsy kolejno wszystkimi liniami X i bada, czy nie pojawiły się one na którejś z linii Y. Dzięki takiemu rozwiązaniu można jednoznacznie określić, które klawisze zostały naciśnięte, a które zwolnione. Informacja o mapie klawiatury jest przekazywana szeregowo do komputera. Dane przesyłane są synchronicznie, a 11-bitowa ramka ma stałą budowę i zawiera: 1 bit startu (zawsze 0), 8 bitów danych, 1 bit kontroli parzystości, bit stopu (zawsze 1). Pojawienie się przychodzącego z klawiatury znaku, komputer sygnalizuje przerwaniem IRQ1.



Schemat działania klawiatury

Należy pamiętać, że każdemu klawiszowi przyporządkowany jest unikalny numer (ang. *scan code*), a naciśnięcie klawisza powoduje przesłanie do komputera tzw. kodu naciśnięcia (ang. *make code*), który to mieści się we wspomnianej 11-bitowej ramce. Ponadto do interpretacji znaczenia klawisza (grupy klawiszy) potrzebna jest cała mapa zestyków klawiatury.

Dzięki ww. możliwości programowania klawiatury i kontrolera oraz istnieniu unikatowych numerów przypisanych każdemu klawiszowi (ang. *scan code*) możliwe jest dostosowanie klawiatury do używania znaków narodowościowych (np. polskich)

Układ znaków na klawiaturze jest zależny od przyjętego standardu. W Polsce stosuje się dwa standardy - pierwszy to tzw. układ programisty, gdzie polskie litery można uzyskać poprzez wybranie znaku "bez ogonka" z jednocześnie wciśniętym prawym przyciskiem ALT, drugi - to tzw. układ maszynistki, zgodny z wcześniejszym standardem maszyn do pisania (QQWERTZ).

Jako ciekawostkę możemy powiedzieć, że taka a nie inna kolejność i ułożenie liter na klawiaturze miały sprzyjać jak ... najwolniejszemu pisaniu (sic!). Zarówno QWERTY jak i QWERTZ powstały w dawnych czasach (sporo przed komputerami), i były opracowywane tak, by utrudniać szybkie pisanie, od którego mogło dojść do zacięcia się klawiatury. Optymalny pod względem szybkości pisania układ Dvoraka - niestety nie przyjął się na prawach standardu

Mysz

Myszkę wynalazł w 1963 roku Douglas Engelbart, pracujący ówcześnie w Stanford Research Center. W latach 70 rozwijaniem technologii "myszatych" zajęła się firma Xerox, ale pierwsza mysz wyprodukowana na masową skalę pojawiła się w latach 80, wraz z upowszechnieniem się komputerów Apple Macintosh i ich graficznego interfejsu użytkownika.



Mysz firmy Microsoft

Jak działa mysz, wiedzą niemal wszyscy. Przesuwanie małym urządzeniem, wyposażonym w przycisk(i), powoduje przekazanie do komputera informacji o zmianie położenia myszy i w rezultacie ruch kurSORA na ekranie. Kliknięcie w przycisk(i) myszki powoduje modyfikację parametrów używanego programu.

Ze względu na sposób, w jaki myszy odczytują swoją pozycję, urządzenia te możemy podzielić na trzy grupy: mechaniczne, optomechaniczne i optyczne. Modele mechaniczne i optomechaniczne działają

według podobnych zasad:

- **Model mechaniczny.** Wewnątrz myszy znajduje się kula wykonana ze specjalnego materiału. Przesuwanie myszy po płaskiej powierzchni powoduje obroty kulki, które zamieniane są na składowe ruchu w kierunkach X i Y. Za proces ten odpowiada para, umieszczonych prostopadle do siebie przetworników (zestawu rolek, diody i fotodetektora), które stykają się z kulką.
- **Model optomechaniczny.** Jedyną różnicą odróżniającą ten model od modelu mechanicznego jest fakt, że zmiana pozycji kulki odczytywana jest za pomocą detektorów optycznych, a nie rolek.
- **Model optyczny.** Tak jak w poprzednim przypadku czujniki optyczne używane są do odczytu zmiany pozycji, jednakże nie kulki, a przesuwania się podłożu. We wczesnych modelach do działania takiej myszy niezbędna była specjalna podkładka pokryta siatką punktów kontrolnych, na których podstawie sensory optyczne odczytują aktualne położenie urządzenia. Współczesne rozwiązania mogą odczytywać poruszenie myszy po dowolnym podłożu, byle nie było to lustro czy też biała kartka

Ważnym parametrem myszy, jest **rozdzielcość myszy** - liczba impulsów wysyłanych w trakcie przemieszczania się urządzenia na odcinku jednego cala (zazwyczaj 300 lub 400 dpi). Naturalnie, im rozdzielcość jest większa, tym lepiej dla użytkownika.

Większość myszy komunikuje się z komputerem wykorzystując złącze PS/2 lub USB. Możliwa jest również bezprzewodowa komunikacja pomiędzy komputerem a myszą chociażby z wykorzystaniem np. standardu IrDA (złącze podczerwieni) czy Bluetooth.

SKANER

Zadaniem skanera jest konwersja dokumentów i zdjęć z postaci papierowej do cyfrowej oraz przesłanie ich, z użyciem sterownika TWAIN (ang. *Technology Without An Interesting Name*), do komputera. Typowym zastosowaniem skanerów jest również rozpoznawanie wyrazów - system OCR (ang. *Optical Character Recognition*). Działanie OCR opiera się na odpowiednio "wytrenowanych" do tego celu sieciach neuronowych, a uzyskane wyniki są, w porównaniu z początkami tej technologii, zaskakująco dobre.

Podstawowym parametrem skanera jest rozdzielcość, na którą składają się:

- **rozdzielcość sprzętowa** - zależy od jakości wykonania fizycznych komponentów skanera (lustra, układu skupiającego). Najczęściej spotykane rozdzielcości to 600x600 dpi (DPI - *Dots Per Inch* - punkty na cal) i 1200x1200 dpi, ale profesjonalne skanery potrafią pracować nawet przy rozdzielcości 14000x5600 dpi.
- **rozdzielcość interpolowana** - jej wartość wynika z zastosowanych algorytmów interpolacji, mających na celu zwiększenie "wyostrzenie" szczegółów obrazu. Typowa wartość 19 200 dpi. Należy jednak pamiętać, że takie obliczenie niekoniecznie musi odpowiadać zeskanowanemu oryginałowi

Kolejnym, bardzo istotnym, parametrem skanera, jest poprawna interpretacja barw oryginału. Standardowo informacja o barwie każdego zeskanowanego punktu zapisana jest na 24 bitach. Profesjonalne skanery odczytują informację o barwie z większą dokładnością. Posługując się tzw. wewnętrznym kodowaniem, uzyskują głębię koloru (np. 36 bitów), po czym z zeskanowanego zbioru wyliczany jest najlepszy (wg programu, który dokonuje tych obliczeń) zakres barw i zapisywany w formacie RGB 24-bitowym.

Innym parametrem, który ma wpływ na jakość skanowanego obrazu, jest zakres gęstości optycznej (D), czyli zdolność prawidłowej prezentacji najjaśniejszych i najciemniejszych partii obrazu.



Skaner płaski

Wszystkie opisane powyżej własności skanera związane są z jego wnętrzem, to ono w największym stopniu decyduje o jakości skanowanych dokumentów i fotografii.

Konstrukcje skanerów podzielić możemy, ze względu na wykorzystane elementy światłoczułe (zamieniające sygnał świetlny na elektryczny), na dwie grupy:

1. Urządzenia wykorzystujące układy typu CCD (ang. *Charge Coupled Device*). Konstrukcje takie zawierają lampa z zimną katodą, której rozgrzewanie, w początkowej fazie skanowania, wymaga dostarczenia prądu o większym natężeniu niż urządzenia drugiego typu. Skanery te mają znaczną wagę. Jest to spowodowane faktem, że głowica skanująca, prócz elementów CCD i lampy, zawiera lustro i układ skupiający.
2. Urządzenia wykorzystujące układy typu CID (ang. *Contact Image Sensor*). Konstrukcja zawiera diody LED wymagające skromnego poboru prądu. Główica skanująca nie zawiera układu lustra i układu skupiającego, dzięki czemu skanery tego typu są bardzo lekkie. Charakteryzują się jednak gorszymi parametrami jakościowymi od swoich starszych braci. Zaletą tych urządzeń jest przystępna cena.

Niezależnie od typu, działanie skanera jest bardzo podobne: podczas skanowania pod dokumentem przemieszcza się głowica skanera (najczęściej o długości równej szerokości skanowanego dokumentu), której lampa (lub diody LED) oświetlają dokument, który odbija światło. Natężenie światła odbitego od elementu ciemniejszego jest inne, niż natężenie światła odbitego od elementu jaśniejszego. Światło odbite (o różnym natężeniu) kierowane jest do komórek światłoczułych (w przypadku elementów CCD wymagany jest specjalny układ lustra-soczewek skupiających). Dla każdego punktu skanowanego dokumentu istnieje zbiór elementów światłoczułych, które zajmują się "obróbką" tego sygnału.

Elementy światłoczułe zamieniają informację o natężeniu światła na impulsy elektryczne, które są przesyłane do komputera. W komputerze informacje o poszczególnych częściach dokumentu są zbierane i generowany jest obraz dokumentu jako całości.

Skanery komunikują się z komputerem poprzez magistrale SCSI, FireWire lub przez łącza USB.

URZĄDZENIA WYJŚCIOWE

MONITOR

Współcześnie jako urządzenia wyświetlające obraz z komputera stosuje się głównie monitory LCD i ich pochodne, niemniej niniejszy opis byłby niepełny, jeśli nie uwzględnimy w nim rozwiązań praktycznie już historycznego - monitorów CRT. Prototypem tych monitorów była, zaprezentowana w 1897 roku, lampa obrazowa typu CRT (ang. *Cathode Ray Tube*), tzw. oscyloskop. Stosowany później kineskop CRT był wciąż lampą elektronową pokrytą od wewnętrz warstwą luminoforu, na którym elektrony rysują obraz z określona częstotliwością, powiedzmy 50 razy na sekundę. Wspomnieć należy, że kluczową rolę w rozwoju kineskopów do monitorów był moment wprowadzenia, na początku lat dziewięćdziesiątych, monitorów kolorowych. Wydarzenie to wzbudziło powszechnie zainteresowanie tym typem urządzeń.

Jedną z właściwości monitorów jest wielkość plamki (ang. *dot pitch*), czyli odstępu między dwoma sąsiednimi punktami obrazu. Pojęcie plamki wiąże się jednak ściśle z konstrukcją maski kineskopu. Dla masek perforowanych, w których każdy piksel składa się z trzech punktów luminoforu (tzw. triady) w kolorach podstawowych RGB ułożonych w kształt trójkąta, wielkość plamki liczona jest jako odległość między punktami luminoforu tego samego koloru. W przypadku masek szczelinowych i kratowych, o wielkości plamki decyduje szerokość jednej triady luminoforu.



Monitor kineskopowy

Istotnym parametrem monitora są również jego wymiary, w tym głównie przekątna obrazu, która opisuje wielkość przekątnej ekranu (np. 17 cali). W ich konstrukcji uwzględnione były różnice w odległości pomiędzy działem elektronowym a środkiem ekranu, oraz działem a krawędzią kineskopu - kształt plamki na krawędzi zbliżony jest do elipsy, zaś w środku do okręgu. Wyrzutnia elektronowa w połączeniu z układem ogniskującym, zapewnia stały kołowy przekrój plamki obrazu, niezależnie od miejsca padania wiązki elektronowej na luminofor. Gwarantowało to wszędzie jednakową ostrość obrazu. Dodatkowo nowe wysokonapięciowe katody (wyrzutnie elektronowe) charakteryzują się mniejszymi wymiarami i zmniejszonym poborem prądu.

Kolejnym parametrem monitorów jest częstotliwość odświeżania (pionowa), która określa, ile razy w

ciągu sekundy tworzony jest obraz. Według normy TCO '99 ergonomiczne minimum, powyżej którego oko ludzkie nie dostrzega migotania, to 85 Hz.

Konwergencja (zbieżność kolorów) stanowi ważny parametr wyświetlanego obrazu. Konwergencja określa precyzję, z jaką wyświetlane są kolorowe punkty. Każdy piksel tworzony jest z trzech barw składowych (czerwonej, zielonej, niebieskiej). Jeżeli miejsce padania na luminofor którejś z wiązek elektronowych, odpowiedzialnych za rysowanie barw, jest przesunięte względem pozostałych dwóch, to obraz punktu ulega rozmyciu.

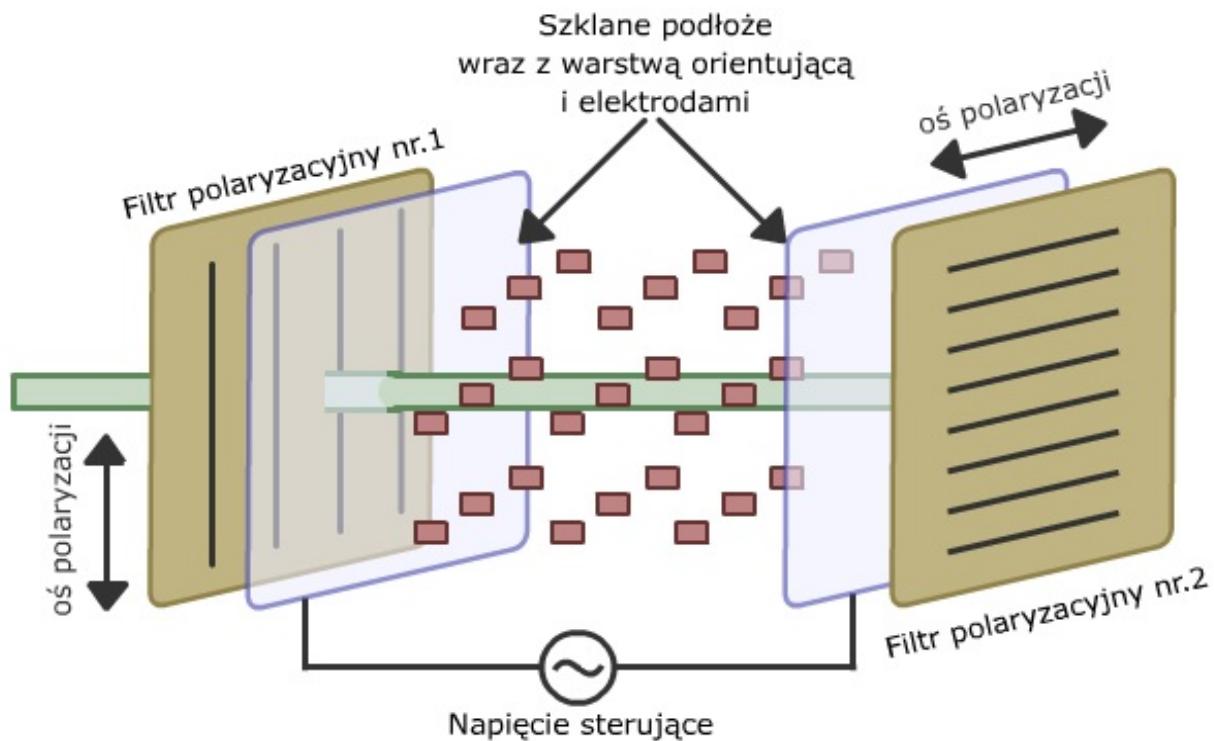
MONITOR LCD

Głównym składnikiem wyświetlacz LCD (ang. Liquid Crystal Display) jest pewna substancja, która wykazuje własności zarówno cieczy jak i ciała stałego. Mowa tu o odkrytym w 1888 roku przez austriackiego botanika Friedrich'a Rheinitzera ciekłym krysztale.

Kolejna ważna data związana z panelami LCD to rok 1973. To wtedy, za sprawą firmy Sharp, na rynek trafił pierwszy seryjnie produkowany kalkulator zawierający wyświetlacz ciekłokrystaliczny.

Ogólna zasadę działania paneli LCD opisać można następująco (patrz rysunek poniżej):

1. Lampa fluorescencyjna emituje światło.
2. Światło przechodzi przez filtr polaryzacyjny nr 1.
3. Spolaryzowane światło dociera do substancji ciekłokrystalicznej, której długie molekuły wcześniej, w procesie produkcyjnym, zostały odpowiednio ułożone wewnętrz każej z komórek matrycy LCD (o rozmiarze np. 1024 x 768 punktów).
 - W przypadku braku zasilania molekuły ciekłego kryształu układają się w położeniu "skręconym", wymuszonym przez warstwę orientującą znajdująca się wewnętrz każej z komórek. Powoduje to zmianę polaryzacji światła o 90 stopni.
 - Pod wpływem napięcia sterowania cząsteczki ciekłego kryształu układają się równolegle do linii pola elektrycznego.
4. Światło po przejściu przez warstwę ciekłokrystaliczną pada na filtr polaryzacyjny nr 2, którego osi polaryzacji jest zmieniona o 90 stopni w stosunku do filtra nr 1. Zależnie od braku/występowania napięcia sterującego możliwe są dwa rezultaty działania filtra:
 - Brak napięcia powoduje przepuszczenie światła przez filtr - odpowiada to emisji światła, a więc zapaleniu się piksela.
 - Przyłożenie napięcia sterowania powoduje wyłumieniu światła przez filtr nr 2 - piksel czarny.



Budowa i działanie pojedynczej komórki LCD wykonanej w technologii Twisted Nematic - przypadek występowania napięcia sterującego.

Ad. 2

Polaryzacja światła z użyciem filtru powoduje zorientowanie fali świetlnej tak, aby była zorientowana w jednej płaszczyźnie, którą wyznacza oś filtra.

Ad. 4

Sterowanie wartością napięcia sterującego pozwala na zmianę przestrzennego ułożenia cząsteczek ciekłego kryształu. Dzięki temu możliwe staje się odchylenie płaszczyzny spolaryzowanego światła o kąt różny od 90 stopni (np. 25 stopni), a więc uzyskanie różnych odcieni szarości. Kolorowe wyświetlacze zawierają dodatkową warstwę, w której skład wchodzą barwne filtry RGB (ang. Red Green Blue). Dzięki złożeniu jednego piksel z trzech różnokolorowych komórek matrycy, możliwe staje się wyświetlenie dowolnego koloru.

Główne rodzaje wyświetlaczy LCD:

1. Wyświetlacz pasywne **DSTN** (ang. *Dual Scan Twisted Nematic*). Cechy charakterystyczne to:
 - duża bezwładność - czas potrzebny na ustalenie się wymaganego napięcia sterującego migracją cząsteczek ciekłego kryształu jest duży, co prowadzi do długiego czasu odświeżania obrazu;
 - wzajemne oddziaływanie na siebie ścieżek przewodzących - powoduje to ograniczenie palety barw możliwych do wyświetlania oraz powstawanie przesunięć obrazu przy dużych kontrastach;
2. Matryca aktywna **TFT** (ang. *Thin Film Transistor*). W roku 1970 po raz pierwszy zastosowano tranzystory wbudowane w ekran ciekłokrystaliczny. Każda komórka matrycy sterowana jest cienkowarstwowym tranzystorem TFT, który reguluje napięcie na elektrodach. Dzięki takiej konstrukcji wyeliminowany został niemal zupełnie niekorzystny efekt wzajemnego wpływu ścieżek przewodzących na siebie.
3. Wyświetlacz **IPS** (ang. *In-Plane Switching*). Ten opracowany przez Hitachi w roku 1995

standard stworzył ekran o kącie widzenia przekraczającym 60 stopni. Odmienne niż przy dwóch poprzednich panelach częsteczki ciekłego kryształu rozmieszczone są w taki sposób, że przyłożenie napięcia sterowania powoduje wyświetlenie piksela. Wyświetlacze tego typu w przeciwieństwie do matryc TFT (złożonych z oddzielonych komórek) zbudowane są jako jednolite struktury, co zapewnia lepszą jakość obrazu. Wadą tych wyświetlaczów jest stosunkowo długi czas reakcji na zmianę obrazu - duży czas potrzebny do wytworzenia odpowiedniego napięcia sterującego ułożeniem częsteczek ciekłego kryształu w komórce.

Wyświetlacze LCD dzięki wielu swym zaletom, do których na pewno należy zaliczyć dużą żywotność (ok. 60 tys. godzin), niewielką grubość oraz ciągle malejącą cenę, aktualnie praktycznie wyparły już klasyczne monitory CRT (kineskopowe). Ponadto, ponieważ są to urządzenia z samej natury swojego działania cyfrowe (w przeciwieństwie do analogowych monitorów CRT), to wraz z ich upowszechnieniem coraz bardziej popularny staje się nowy standard połączenia monitora z kartą graficzną, nazwany DVI, w którym obraz przesyłany jest cyfrowo, co pozwala na uzyskanie jego wyższej jakości oraz zwalnia nas z korygowania ustawień monitora. Nie bez znaczenia są też inne cechy LCD, takie jak brak szkodliwego migotania, czy też zdecydowanie niższy poziom promieniowania elektromagnetycznego.

W tym miejscu należałoby jeszcze wspomnieć o monitorach LED - które tak naprawdę nie są nową technologią, od LCD różnią się jedynie sposobem podświetlania ekranu. W przypadku LCD źródłem podświetlenia są (najczęściej dwie) świetłówki, w przypadku monitorów LED - matryca diód LED. To drugie rozwiązanie pozwala na wygaszanie części elementów matrycy w przypadku ciemnych scen, i uzyskiwanie dzięki temu głębszej czerni, oraz jest bardziej efektywne energetycznie.

DRUKARKA

Drukarka, w przeciwieństwie do skanera, zamienia postać dokumentu z cyfrowej na papierową. Drukarki podzielić możemy na trzy grupy w zależności od techniki jaką wykorzystują do drukowania:

1. **Drukarki igłowe.** Technika druku opiera się na dwóch elementach: głowicy (z igłami) i taśmie barwnej. Igły głowicy uderzając w taśmę barwiącą powodują przeniesienie barwnika na papier. Drukarki tego typu niestety są bardzo głośne i oferują słabą jakość wydruku, i mają w zasadzie jedną zaletę - są w stanie drukować na kalce w więcej niż jednej kopii. Dlatego znajdzicie je jeszcze stosowane jedynie do wydruku faktur, potwierdzeń, itp...
2. **Drukarki atramentowe.** Wykorzystują atrament jako nośnik informacji. Atrament, który znajduje się w zbiornikach, jest doprowadzany do głowicy i wypychany przez dysze. Istnieje wiele rodzajów drukarek atramentowych, wykorzystujących różne techniki do nakładania na papier atramentu. Najpopularniejsze z nich to:
 - technika termiczna - tutaj kropelka atramentu jest "wypluwana" na papier na skutek rozprężenia pęcherzyka powietrza (w każdej z dysz znajduje się specjalny "grzejniczek", który w krótkim czasie potrafi rozgrzać się do wysokiej temperatury. W wyniku takiej reakcji wytwarza się para wodna, której pęcherzyk wypycha kropelkę atramentu z dyszy).

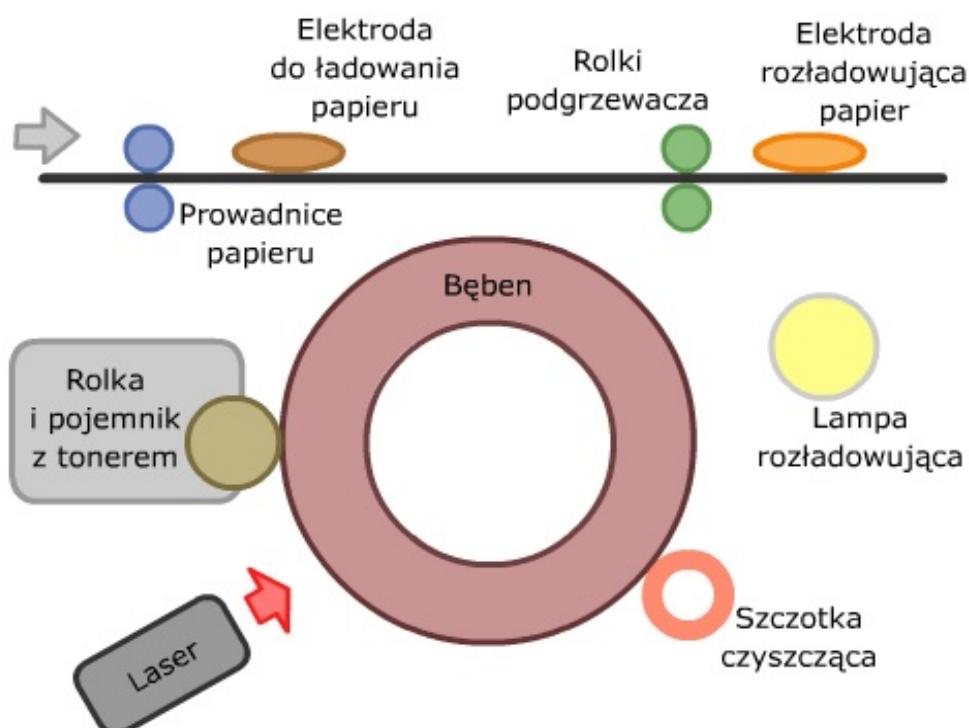


Drukarka atramentowa firmy HP

- technika piezoelektryczna - w tym wypadku kropelka jest wypychana przez kryształki, które pod wpływem przyłożonego napięcia zmniejszają swoją objętość, co powoduje zwiększenie ciśnienia w dyszy i w rezultacie "wyplucie" kropelki atramentu.

3. **Drukarki laserowe.** Zasadę działania drukarki laserowej opiszymy w punktach i zilustrujemy (patrz rysunek 2.16.):

- Za pomocą specjalnej rolki, toner jest rozkładany w tych miejscach, które wcześniej oświetlił promień lasera - ujemnie naładowany toner jest przyciągany przez dodatnio naładowane punkty bębna.
- Naelektryzowany toner przyciągany jest przez papier (również naelektryzowany, lecz przeciwnie), po czym zostaje utrwalony przez zespół wałków rozgrzanych do 200 stopni Celsiusza (w temperaturze tej toner topi się).



- W trzecim etapie bęben jest oczyszczany z resztek barwnika i rozładowywany. Jest gotowy do kolejnego wydruku.

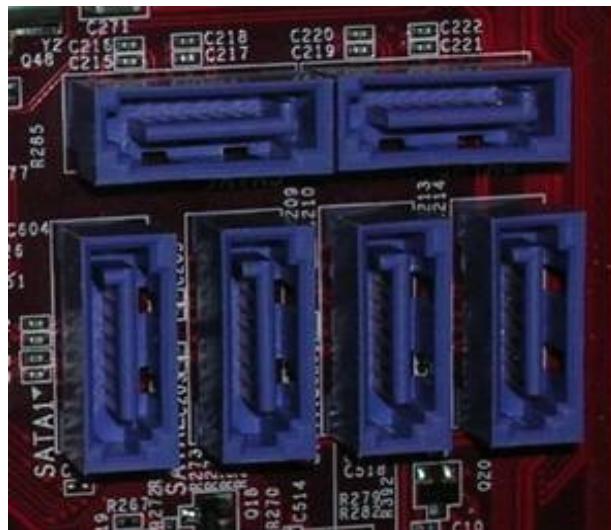
Najważniejszym parametrem drukarki jest maksymalna rozdzielcość, z jaką jest w stanie drukować (rozdzielcość ta może być inna dla wydruku kolorowego, a inna dla czarno-białego). Typowe rozdzielcości popularnych drukarek atramentowych to 600x300 dpi, 600x600 dpi. Oczywiście drukarki laserowe (ale i atramentowe również) oferują znacznie większe rozdzielcości. Drugim, ważnym czynnikiem jest czas drukowania dokumentów (drukarki atramentowe ok. 4-6 stron na minutę, drukarki laserowe 10 i więcej).

Na zakończenie warto wspomnieć, że drukarki komunikują się z komputerem wykorzystując porty szeregowe USB, lub też korzystając bezpośrednio z interfejsów sieciowych.

STANDARDY INTERFEJSÓW

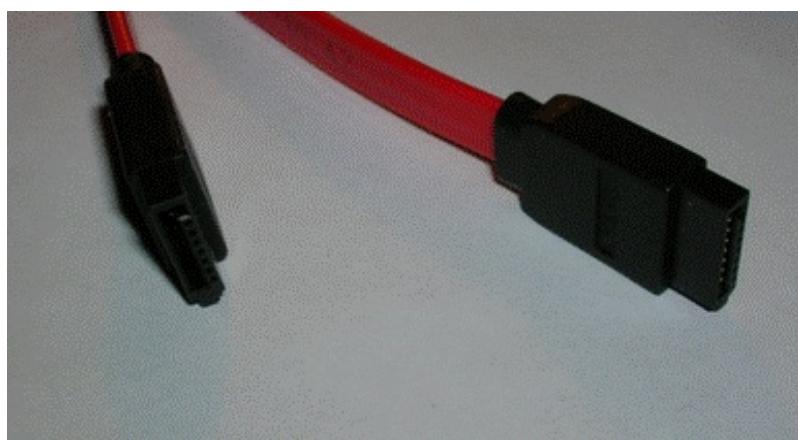
SERIAL ATA (SATA)

Jest to nowy interfejs do podłączania urządzeń wewnętrznych takich jak dyski twarde HDD i napędów optycznych np. DVD RAM. Standard ten został opublikowany w roku 2001. Wypiera on powoli standard ATA który osiągnął swoje maksimum jako medium transmisyjne na poziomie 133MB/s w trybie Ultra DMA. Standard SATA umożliwiał transfer z prędkością 180 MB/s (1,5 GBit/s)



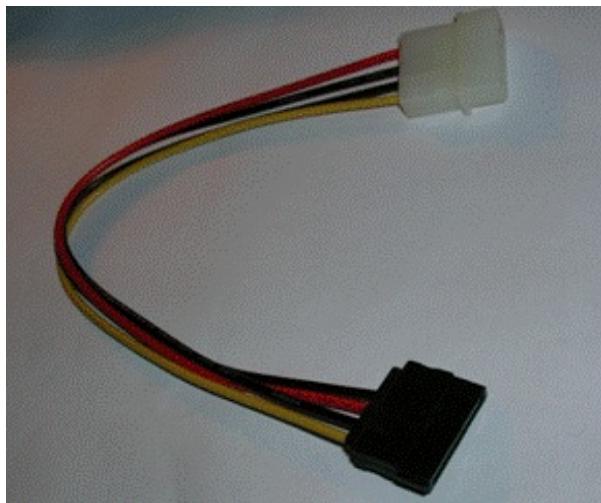
Gniazda kontrolera SATA na płycie głównej

Każde urządzenie Sata podłączane jest osobnym kablem do portu. Standard przewiduje podłączanie „na gorąco”. Kabel sygnałowy zakończony jest wtyczką z siedmioma stykami o różnej długości. Podobnie jak w przypadku USB. Najpierw załączone zostaje zasilanie, następnie styki sygnałowe. Natomiast złącze zasilające urządzenia SATA ma 15 styków.



Wtyczka złącza Sata

Sata on the Go jest to fizycznie wyprowadzony port z tyłu obudowy komputera. Rozwiążanie takie wprowadziła jako pierwsza na swoich płytach głównych firma ASUS. Podłączanie zewnętrznego dysku odbywa się podczas pracy komputera.



Przejściówka z wtyczki MOLEX do zasilania dysku SATA

Współcześnie na rynku dostępne są kolejne wersje standardu SATA: **SATA 2** pozwalająca na transfer danych do 3 GBit/s (ok 358 MB/s, w praktyce nie więcej niż 300 MB/s). Ponadto w tej wersji standardu wprowadzono NCQ (Native Command Queuing) - czyli kolejkowanie rozkazów. Prowadzi to do wzrostu wydajności oraz lepszego rozłożenia obciążenia. Specyfikacja SATA II wprowadziła także Backplane- zewnętrzną macierz dyskową (o dużej liczbie dysków) podłączoną bezpośrednio do kontrolera SATA. W kolejnej wersji, **SATA III** poza dalszym przyspieszeniem transferu do ok 6 GBit/s (751 MB/s), wprowadzono nowe polecenie przesyłania strumieniowego, umożliwiające izochroniczny transfer danych, przydatny głównie podczas odtwarzania strumieni multimedialnych.

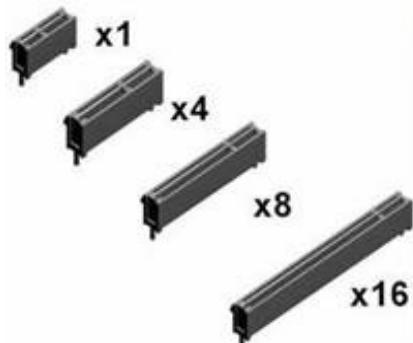
PCI EXPRESS



gniazdo PCI Express x16. poniżej x1

Jest to szeregowa szyna systemowa, służąca głównie do podłączania urządzeń (kart rozszerzeń) do płyty głównej. Standard ten powstał jako następca starszego PCI, i umożliwia przesyłanie dwustronne

danych (zarówno od, jak i do podłączonego urządzenia). Ze względu na połączenie bezpośrednio na płycie, osiągane prędkości były znacznie wyższe niż kablowe łącza szeregowe, dlatego też początkowo PCI Express był wykorzystywany głównie do podłączenia karty graficznej. Podstawowa wersja oznaczona została jako PCI Express 1x (dla obustronnej przepustowości 200MB/s). Zgodnie ze specyfikacją dostępne są gniazda z prędkościami 2x 4x 8x 16x 32x.



Porównanie gniazd PCI Express

Wcześniej stosowany standard - AGP - dziś już przeszedł do historii.



Port AGP x8

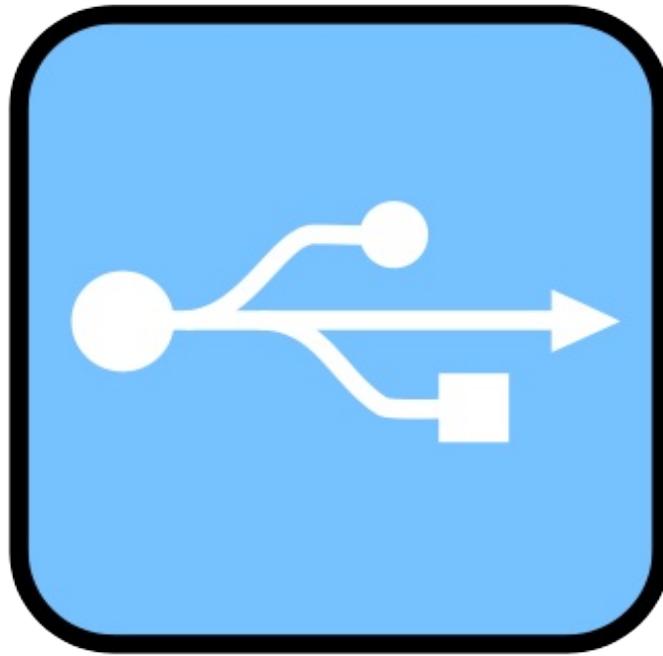
THUNDERBOLT

W 2009 roku firma Intel we współpracy z Apple zaprezentowała połączenie łącza PCI Express z DisplayPort (interfejsem używanym do podłączania cyfrowych monitorów). W efekcie otrzymano bardzo szybki (10 Gbit/s z możliwością dalszego przyspieszania do 100 Gbit/s) interfejs służący do podłączania zewnętrznych napędów czy kamer, ale także kart graficznych, kontrolerów RAID czy monitorów. Standard miał zastąpić starsze magistrale, takie jak USB czy FireWire - jednakże ze względu na początkowy brak wsparcia standardu przez producentów sprzętu (poza Apple) nie upowszechnił się zbytnio. Teraz powoli sytuacja się zmienia, jednak wydaje się iż USB 3.0 jest wystarczający do większości zastosowań.

USB

Standard USB (ang. *Universal Serial Bus*) to rodzaj sprzętowego portu komunikacyjnego komputerów, zastępującego stare porty szeregowe i porty równoległe. Został opracowany przez firmy Microsoft, Intel, Compaq, IBM i DEC. Na dzień dzisiejszy jest to podstawowy standard podłączania urządzeń

peryferyjnych do komputerów osobistych, na tyle uniwersalny, że można go wykorzystać do podłączenia np.: kamery wideo, aparatu fotograficznego, telefonu komórkowego, modemu, skanera, klawiatury, przenośnej pamięci itp.



Logo USB, źródło - Wikipedia.

Standard wspiera łącze typu hot-plug (możliwość włączania / wyłączania urządzenia w trakcie pracy systemu), oraz kilkanaście profili różnych urządzeń. Sama architektura USB jest architekturą klient / serwer, co oznacza, iż zawsze w komunikującej się parze musi istnieć strona nadziedzna (tu nazywana host USB) oraz podrzędna. Host może zarządzać wieloma kontrolerami, a każdy może mieć więcej niż jeden port - co daje możliwość tworzenia mini sieci. Każde urządzenie komunikuje się z hostem korzystając z wydzielonych logicznych kanałów (16 wejściowych, 16 wyjściowych), w sumie teoretycznie host może obsługiwać do 127 urządzeń.

Poprzez złącze USB może także być doprowadzone zasilanie do urządzenia (do 500 mA w standardzie USB 1.1). Urządzenia podłączane w ten sposób mogą być automatycznie wykrywane i rozpoznawane przez system, przez co instalacja sterowników i konfiguracja odbywa się w dużym stopniu automatycznie.

USB występuje w kilku podstawowych wersjach:

- USB 1.1 - Urządzenia spełniające warunki tej specyfikacji mogą pracować z szybkością (Full Speed) 12 Mbit/s (1,5 MB/s) i (Low Speed) 1,5 Mbit/s (0,1875 MB/s). Prędkość tego portu nie pozwalała na wygodną pracę z napędami dyskowymi czy kartami sieciowymi, więc w praktyce jego zastosowanie było dość ograniczone.
- USB 2.0 - Urządzenia zgodne z tą specyfikacją mogą pracować z maksymalną szybkością 480 Mbit/s (60 MB/s). Rzeczywista szybkość przesyłu danych zależy od konstrukcji urządzenia, i jest zazwyczaj znaczco niższa, ze względu na wykorzystanie transmisji typu half-duplex na jednej parze przewodów.
- USB 3.0 - Urządzenia zgodne z tą specyfikacją mogą pracować z szybkością 5 Gbit/s (640 MB/s). Rzeczywista przepustowość łącza danych wynosi 4 Gbit/s, co przy zastosowaniu kodowania 8b/10b pozwala uzyskać transfer rzędu 400 MB/s. Nowy standard oprócz standardowych przewodów (dla kompatybilności w dół z USB 2.0 i 1.1) do szybkich transferów

- wykorzystuje dwie dodatkowe, ekranowane pary przewodów, co umożliwia pracę w trybie pełnego duplex-u.
- USB 3.1 - Wersja przyszłościowa, do powszechnego użytku ma wejść w 2015 roku. Prędkość maksymalna będzie wynosiła do 10 Gbit/s, dodatkowo zwiększo połów mocy przez urządzenie zasilane do 100W. Standard 3.1 będzie kompatybilny z USB 3.0 i 2.0.

SLI

nVidia Scalable Link Interface to technologia firmy nVidia. Jest to połączenie dwóch kart graficznych tejże firmy w celu uzyskania wzrostu wydajności. Tutaj złączenie kart dokonuje się specjalnym mostkiem wewnętrz komputera. Karty muszą być identyczne, oraz płyta główna musi wspierać obsługę tego rozwiązania. Do pełnego wykorzystania możliwości złącza wymagany jest dość wydajny procesor, oraz dwa gniazda PCI-E 16x na płycie głównej.

CROSSFIRE

Jest to innowacja Firmy ATI. Wzrost wydajności kart graficznych uzyskuje się poprzez połączenie mocy obliczeniowej dwóch kart graficznych. Jest one odpowiedzią na SLI firmy nVidia. Aby skorzystać z tego rozwiązania potrzebna jest płyta główna wyposażona w specjalizowany chipset Radon Xpress 200 CrossFire Edition, oraz dwie karty graficzne ATI mogące obsługiwać tą technologię. Fizycznie połączenie jest wykonywane (podobnie jak SLI) za pomocą kabla na zewnątrz obudowy.

IEEE -1394 (FireWire 400)

FireWire jest znakiem handlowym firmy Apple - komputery tej firmy standardowo są wyposażane w tą magistralę szeregową. Natomiast nazwa IEEE-1394 pochodzi z nr normy amerykańskiego komitetu normalizacyjnego IEEE(The Institute of Electrical and Electronic Engineers)

Ujęte w dokumentach normalizacyjnych cechy magistrali 1394 obejmują szereg zagadnień jak m.in. prędkości transmisji, topologie i protokoły. Transfer odbywa się w trybie asynchronousnym. Ale standard przewiduje możliwość wynegocjowania minimalnej gwarantowanej przepustowości.

Zdefiniowane są trzy przepustowości:

- 98.304Mb/s (S100)
- 196.608Mb/s (S200)
- 393.216Mb/s (S400)

Wartości w nawiasach są zaokrąglonymi podawanymi często w prasie oraz informacjach na urządzeniach zgodnych z FireWire. I tak na przykład kontrolery PCI pracują w zakresie S400, kamery zazwyczaj w S100. FireWire jest magistralą, co oznacza że teoretycznie do jednego złącza można podłączyć aż do 63 urządzeń. W praktyce rzadko spotyka się połączenie więcej niż dwóch urządzeń, tym bardziej, iż cała magistrala pracuje zawsze z prędkością najwolniejszego urządzenia.

Fire Wire 800

To nowsza i szybsza wersja interfejsu FireWire znana też jako IEEE-1394b. Jak nazwa wskazuje, można nim przesyłać dane z prędkością dochodzącą do 800MB/s (w rzeczywistości 768 Mb/s). W stosunku do

wcześniej wersji standardu liczba urządzeń nie uległa zmianie, podobnie jak topologia połączeń. Wprowadzono natomiast możliwość korzystania ze światłowodów, co wydłużyło dozwoloną długość połączenia do ok. 100 metrów.



Kabel Fire Wire

WI-FI (WIRELESS FIDELITY)

Skrót ten oznacza technologię tworzenia sieci bezprzewodowych za pomocą fal radiowych o wielkiej częstotliwości. Potocznie nazywane skrótem WLAN (od słów Wireless Local Area Network), z powodzeniem zastępują tradycyjne połączenia za pomocą kabli skrętkowych. Specyfikacjami sieci bezprzewodowych zajmuje się komitet IEEE. Więcej o sieciach bezprzewodowych napiszemy w trzeciej części podręcznika.

BLUETOOTH

Bluetooth (niebieski ząbek) to także standard komunikacji bezprzewodowej, tyle że nie nastawiony na tworzenie sieci komputerowej, a na współpracę z urządzeniami periferyjnymi. W 1994 roku firma Ericsson rozpoczęła prace nad nowym standardem łączenia telefonów komórkowych z innymi urządzeniami. Wspólnie z partnerami założono grupę zainteresowaną rozwojem standardu, i w 1999 roku opublikowano pierwszą jego specyfikację. W założeniach była to sieć bezprzewodowa krótkiego zasięgu o niskim poborze energii. Mechanizmy transportowe w niej zawarte były na tyle uniwersalne by można było przesyłać zarówno pliki jak i dane sterujące oraz strumienie multimedialne (głównie informacji głosowych).

Struktura łącz Bluetooth opiera się o tzw pikosieci - małe, wydzielone sieci składające się maksymalnie z 8 węzłów aktywnych, z których jeden pełni rolę mastera (jednostki nadzędnej). Dodatkowo w pikosieci może pracować do 255 uspionych węzłów, pozostających w trybie niskiego poboru mocy i oczekiwania na polecenia od master-a. Wiele pikosieci może równolegle egzystować w tym samym obszarze, bez zakłócania swojej pracy.

BT pracuje na ogólnie dostępnym paśmie 2.400 – 2.4835 GHz. Pasmo to podzielone jest na 79 kanałów. Podczas połączenia kanały zmieniane są okresowo z częstotliwością 1600 skoków na minutę. Skoki te wykonywane są według ścisłe ustalonego algorytmu.

Specyfikacja Bluetooth określa trzy klasy nadajników BT:

- Klasa 1 to maksymalna moc promieniowania do 100mW, co daje teoretyczny zasięg w otwartej przestrzeni do 100 metrów.
- Klasa 2 o mocy do 2.5 mW i zasięgu do 10 m jest najczęściej wykorzystywana (np standardowo wbudowywana w telefony komórkowe) .
- Klasa 3 to urządzenia promieniujące poniżej 1mW, przy zasięgu do 1 metra.



Adapter USB Bluetooth

Istnieje kilka wersji standardu Bluetooth różniących się głównie prędkością. Najczęściej spotykane rozwiązania to:

- Bluetooth 1.1 o prędkości do 124 kb/s - wystarczającej do strumieniowego przesyłania glosu,
- Bluetooth 2.0 o prędkości do 2,1 Mb/s,
- Bluetooth 3.0 o prędkości 24 Mb/s (wersja 3.1 jeszcze podnosiła prędkość do 40 Mb/s),

Zupełnie nowym jakościowo rozwiązaniem jest wchodzący właśnie standard Bluetooth 4.0 w połączeniu ze specyfikacją Low Energy (LE), nazywany handlowo także Bluetooth Smart - opracowywany początkowo przez firmę Nokia, potem w ramach grantu z programu ramowego Unii Europejskiej. W tym przypadku skupiono się nie na transferze danych, a na maksymalnym możliwym obniżeniu poziomu zużycia energii. Finalnie, istnieje możliwość pracy urządzeń zgodnych z BTLE przez kilkanaście miesięcy przy zasilaniu z baterii zegarkowej - co, w połączeniu z implementacją w najnowszych modelach smartphone, otworzyło zupełnie nowe obszary zastosowań, jak np. ubieralna elektronika czy też tak zwane beacons, wprowadzane aktywnie także przez polskie firmy na rynek światowy (np firma [Estimote](#)).

ZASADY DOBORU KONFIGURACJI SPRZĘTOWYCH

KOLEJNE ETAPY WYBORU ODPOWIEDNIEGO SYSTEMU KOMPUTEROWEGO

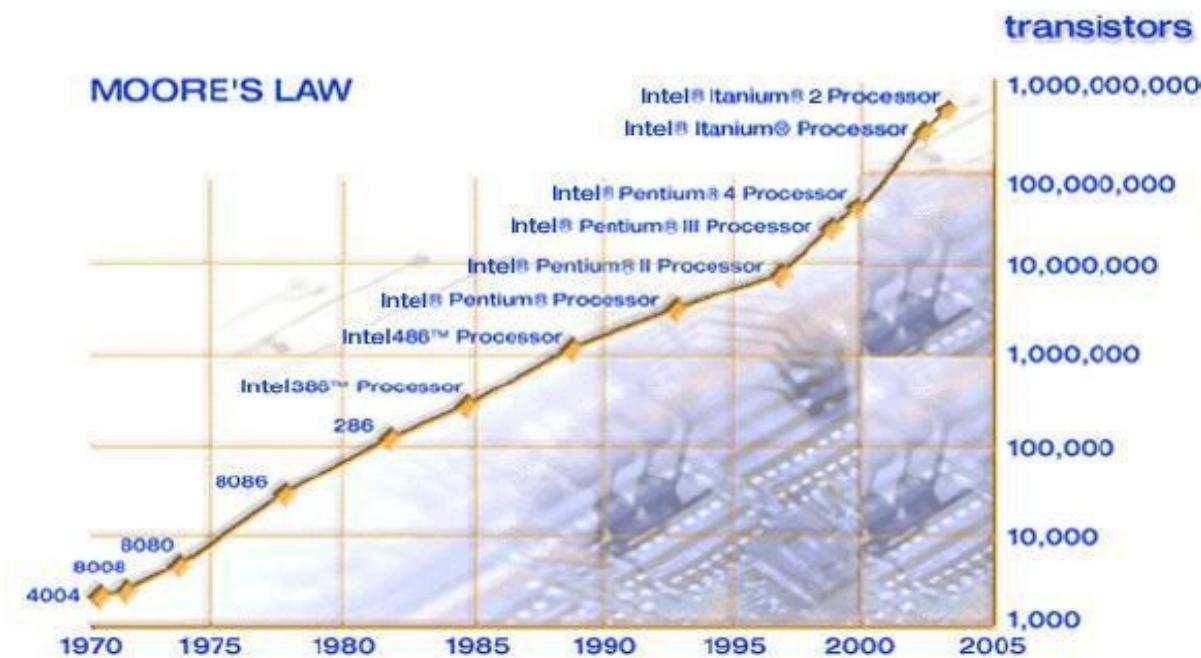
Zanim przystąpimy do analizy istotnych parametrów każdego elementu składowego systemu komputerowego, najpierw chcielibyśmy zaprezentować modelowy proces wyboru sprzętu komputerowego.

System komputerowy zazwyczaj służy do tego, by realizować określone zadania - inaczej będzie kolejnym drogim meblem w pomieszczeniu, ew. grzałką w zimne dni ;). Przez długi czas (szczególnie w naszym kraju) nie zwracano uwagi na fakt, iż zastosowanie systemów komputerowych w jakiejś dziedzinie nie jest wyznacznikiem nowoczesności, wymogiem czasów czy też "prawdą ekranu", lecz ma przynosić konkretne, wymierne korzyści. Ma skrócić czas wykonywania odpowiednich zadań, ograniczyć koszt pracy ludzkiej, zwiększyć bezpieczeństwo i dostępność danych, podnieść jakość produktu itp.

Jednocześnie, zastosowanie systemów komputerowych zazwyczaj pociąga za sobą dodatkowe koszty czy też utrudnienia. Wprowadzenie komputerów w okienkach bankowych lub pocztowych zwiększy czas obsługi pojedynczego petenta i w rezultacie wydłuży kolejkę, napisanie listu przy wykorzystaniu Worda zajmie więcej czasu niż napisanie go ręcznie, napisanie maila czy chat on-line także przebiega wolniej niż klasyczna rozmowa telefoniczna. Zakup sprzętu komputerowego wiąże się również z zakupem oprogramowania i ponoszeniem kosztów jego pielęgnacji (administracji, uaktualnień, modyfikacji).

Zatem, zamiast podejmować pochopnie decyzję o modernizacji, zastanówmy się czy nam się to opłaca. To jest zadanie bardziej dla menegera niż informatyka, niemniej jednak każdy powinien sobie zdawać sprawę, że komputer nie jest złotym środkiem, który sam z siebie sprawi że praca stanie się łatwiejsza, przyjemniejsza, a życie nabierze odcienia różowego.

Zakup sprzętu komputerowego jest inwestycją - taką jak każda inna, dodatkowo o bardzo kimż żądanym czasie amortyzacji. W latach 70 ubiegłego wieku zostało sformułowane heurystyczne prawo Moora, w oryginalnym sformułowaniu mówiące, że ekonomicznie optymalna liczba tranzystorów w układzie scalonym podwaja się co 18 miesięcy (obserwację tę przypisuje się Gordonowi Moore'owi, jednemu z założycieli firmy Intel). Termin ten jest też używany do określenia praktycznie dowolnego postępu technologicznego. "Prawo Moore'a" mówiące, że "moc obliczeniowa komputerów podwaja się co 18 miesięcy" jest nawet popularniejsze od oryginalnego prawa Moore'a.



Prawo Moora w odniesieniu do ilości tranzystorów w procesorze na przestrzeni ostatnich lat

Tak szybki, wykładniczy postęp w dziedzinie sprzętu komputerowego ma jeden poważny minus - zazwyczaj komputer staje się przestarzały po 18-48 miesiącach od daty wyprodukowania (w zależności od zastosowania) i w tym czasie powinien zwrócić się koszt poniesiony przy jego zakupie.

Pamiętajmy o tym przystępując do zakupów sprzętu komputerowego. Komputery mają wyjątkowo krótki czas użytkowania, zbliżony jedynie do telefonów komórkowych (większość tzw. trwałych dóbr ma większy czas życia i nie traci tak szybko na wartości). O ile jednak telefony są subsydiowane przez operatorów sieci komórkowych, o tyle w przypadku komputerów łatwo można doprowadzić do sytuacji, w której inwestycja w sprzęt komputerowy nie zwróci się nam, bądź w przypadku zakupów prywatnych - koszt posiadania nowoczesnego komputera będzie ogromny.

Przebieg procesu doboru sprzętu przedstawimy na prostym przykładzie odnoszącym się do zastosowań z rynku SOHO. Pierwszym krokiem przy wyborze sprzętu komputerowego powinno być jasne określenie jakie funkcje ma on spełniać, jakie zadania mamy zamiar przed nim postawić. Następnie powinniśmy oszacować wymaganą wydajność oraz wielkość zasobów (RAM, HDD) systemu komputerowego. Potem zestawiamy przykładową konfigurację, zwracając uwagę na ew. dodatkowe elementy komputera nie związane z wydajnością (monitor, drukarka, dodatkowy osprzęt) - nie zapominając także o koszcie niezbędnego oprogramowania. Następnym krokiem powinno być oszacowanie ceny takiego zestawu. Jeśli cena jest zbliżona do dostępnego budżetu - to super. Jeśli jest od niego znacznie niższa - może warto kupić system nieco wydajniejszy. Jeśli jest zbyt wysoka - musimy szukać oszczędności, w pierwszej kolejności w elementach peryferyjnych bądź wartościach dodanych (np. rezygnując ze sprzętu firmowego na rzecz sprzętu no-name), starając się nie zmieniać wydajności i mocy obliczeniowej systemu.

DEFINICJA WYMAGANEJ FUNKCJONALNOŚCI

Zadania stawiane przed systemem komputerowym można definiować na różnych poziomach dokładności. Optymalną sytuacją jest taka, w której jawnie możemy powiedzieć że potrzebujemy komputera tylko i wyłącznie do jednego celu. Rozważmy taką sytuację:

jestem programistą, będę tworzył aplikacje dla systemu Windows jako wolny strzelec (poza firmą), poza tym komputer będę wykorzystywał jedynie do przeglądania internetu. W swojej pracy wykorzystuję głównie środowiska do tworzenia oprogramowania firmy Borland, a na nowym komputerze chciałbym pracować z najnowszym Embarcadero Developer Studio. Komputer zmieniam średnio co 2 lata, pracuję w domu zazwyczaj ponad 8 godzin przed komputerem i jedynie ja będę korzystał z tego komputera. Do komfortowej pracy wymagam dwóch monitorów. W domu posiadam już stałe łącze internetowe, oraz router z możliwością podpięcia się do niego kablem Ethernet.

O SZACOWANIE ŻĄDANEJ WYDAJNOŚCI I WIELKOŚCI ZASOBÓW

Dzięki tak dokładnemu zdefiniowaniu zadań dla komputera, jesteśmy w stanie szybko i bezproblemowo wybrać komputer. Po prostu szukamy na sieci strony producenta naszego głównego programu, a następnie tam wyszukujemy wymagania sprzętowe dla niego. W omawianym przykładzie znajdziemy:

1 GB RAM (2 GB zalecane), 44 GB wolnego miejsca na dysku, DVD, GPU z obsługą DirectX 9 i PixelShader Level 2, procesor Pentium o zegarze co najmniej 1.6 GHz (2 GHz zalecane), monitor o standardowej rozdzielczości i Windows 7 albo 8.

Wymagania stawiane przez RAD Studio przewyższają te, których wymaga sam system operacyjny - lecz tu uwaga - ilość pamięci RAM powinniśmy zsumować, czyli określamy, że komputer musi mieć co najmniej 3 GB RAM. Przeglądanie sieci internet nie rozszerza wymagań sprzętowych (nawet systemu operacyjnego, nie mówiąc już o RAD Studio), więc możemy je pominąć. Przewidywany czas użytkowania komputera jest na tyle krótki, iż możemy się spodziewać, że pojawi się jedna / dwie nowe wersje naszego głównego programu, vide - musimy zapewnić pewien zapas mocy obliczeniowej i zasobów.

WSTĘPNA KONFIGURACJA

Decydujemy się więc na konfigurację zalecaną, czyli:

- komputer ma być stacjonarny (nie planujemy pracy poza domem.);
- względy prestiżowe czy też warunki przedłużonej gwarancji nie są zbyt istotne - więc będzie to komputer no-name (składak);
- procesor z serii profesjonalnej (więc Intel Core, Athlon FX, ale nie Celeron, Atom czy Sempron) o w zasadzie dowolnym zegarze (wszystkie sprzedawane współcześnie są szybsze od wymienionych w specyfikacji);
- płyta główna dobrana do procesora, bez specjalnych wymagań odnośnie udostępnianych interfejsów, z kartą sieciową i układem dźwiąkowym na płycie;
- ilość pamięci RAM - 8 GB, ze względu na przyszły rozwój oprogramowania;
- dysk twardy w rozsądnej cenie (pojemności współczesnych dysków znaczco przekraczają wymagania);
- napęd DVD-RW (przyda się do przygotowywania płyt instalacyjnych dla stworzonych programów);
- dowolna współczesna karta graficzna, z dwoma wyjściami DVI. Ze względu na wymaganie pracy dwumonitorowej rozwiązania graficzne wbudowane w płytę raczej odpadają;
- dwa monitory LCD o rozmiarze co najmniej 24 cale, z wejściami DVI (długotrwała praca przy komputerze męczy wzrok);
- obudowa, mysz, klawiatura - w zasadzie dowolne, byle trwałe;
- brak dodatkowych, specjalizowanych kart oraz urządzeń wejściowo / wyjściowych;
- system operacyjny + oprogramowanie antywirusowe.

ZAKUP

Mając tak przygotowaną listę możemy przystąpić do skalkulowania ceny w sklepie internetowym bądź w dowolnej firmie zajmującej się sprzedażą komputerów. Jeśli cena jest odpowiednia - dokonujemy zakupu i problem mamy z głowy :)

DEFINICJA WYMAGAŃ DLA SYSTEMU KOMPUTEROWEGO

Tak jasne i klarowne określenie wymagań dla systemu komputerowego, jak to które zaprezentowaliśmy w poprzednim segmencie, niestety rzadko zdarza się w praktyce. Zazwyczaj komputer nie jest wykorzystywany tylko i wyłącznie do jednego celu, lecz chcielibyśmy by był urządzeniem w pełni uniwersalnym. Niestety - uniwersalność kosztuje. Jeśli mamy nieograniczony budżet to wszystko jest ok. Zazwyczaj jednak chcielibyśmy optymalizować koszty, czyli uzyskać sprzęt o wymaganej przez nas funkcjonalności za jak najniższą cenę.

Musimy więc zdefiniować sobie jasno, czego od zestawu komputerowego oczekujemy. Lecz każdy z nas jest inny, każdy może potrzebować czegoś innego i nie istnieje jeden, uniwersalny zestaw cech, które moglibyśmy przejrzeć, przy niektórych z nich postawić znak plus, przy innych minus - i mieć gotowy przepis na zestaw komputerowy. Spróbujemy jednak choć zarysować podział zastosowań systemów komputerowych wraz z podstawowymi wymaganiami w stosunku do każdego z nich. Podstawowy podział klasycznych systemów komputerowych (pomijamy tutaj systemy osadzone oraz superkomputery) z punktu widzenia zastosowania przebiega pomiędzy zastosowaniami infrastrukturalnymi a osobistymi.

SYSTEMY KOMPUTEROWE W ZASTOSOWANIACH INFRASTRUKTURALNYCH

Takie zastosowania systemów komputerowych, przy których w normalnym trybie pracy nie siedzi żaden użytkownik, będącymi tutaj nazywali infrastrukturalnymi. Typowym przykładem takiego zastosowania są wszelkiego rodzaju serwery pracujące 24 h na dobę, dostępne jedynie zdalnie i realizujące jakąś usługę dla większej liczby użytkowników, jak np. dostarczanie poczty, stron www, aktualnego czasu, uwierzytelnianie użytkowników sieciowych, itp. Do tego grona zaliczymy również komputery w centrach danych, główne komputery przemysłowych systemów sterowania, systemy monitoringu i archiwizacji danych czy nawet komputery autonomicznych stacji meteorologicznych. W przypadku takich systemów możemy wyróżnić kilka charakterystycznych wymagań stawianych przed nimi.

WYMAGANIA NIEZAWODNOŚCIOWE

- Zazwyczaj głównym (a przynajmniej jednym z najważniejszych) wymaganiem jest **niezawodność**. Stosuje się różne miary niezawodności sprzętu komputerowego, najpopularniejszą z nich jest MTBF (ang. *Mean Time Between Failure*), wyrażający średni czas pomiędzy awariami danego urządzenia. MTBF jest określany na podstawie badań statystycznych przez producenta sprzętu. MTBF jest zazwyczaj podawany w godzinach, a jego najprostsze możliwe znaczenie fizyczne można przedstawić następująco: mamy 50% szans że system ulegnie uszkodzeniu po upłynięciu podanej liczby godzin. W przypadku, gdy MTBF podawany jest niezależnie dla każdego elementu systemu komputerowego, średni czas między awariami dla całego systemu jest najniższą wartością wśród podanych.
- Związana z niezawodnością jest również możliwość stosowania **redundancji**, czyli równoległej pracy dwóch lub więcej systemów komputerowych (w takiej samej konfiguracji i z takimi samymi danymi), z których jeden wykorzystuje rzeczywistą pracę, a drugi czeka w odwodzie. W przypadku uszkodzenia systemu głównego system zapasowy natychmiast przejmuje jego wszystkie zadania w sposób niezauważalny dla użytkownika. Redundancja może dotyczyć całych systemów komputerowych, jak i ich poszczególnych podzespołów, jak np. zasilacz czy pamięć masowa.
- Następną cechą, która powiązana jest z dwoma poprzednimi, jest tzw. **hot swap**, czyli możliwość wykonania naprawy systemu komputerowego (wymiany jego elementów) bez zatrzymywania całego systemu, w trakcie jego pracy.

Powyższe wymagania (a raczej poziom ich spełnienia) definiują nam finalnie stopień osiągalności systemu, czyli procent czasu, w którym realizuje on przedstawione przed nim zadania. Przeprowadzając analizę dotyczącą zadań (które muszą być spełnione, a które niekoniecznie), należy zawsze brać pod uwagę koszty ponoszone przez nas w związku z przestojem systemu komputerowego, oraz koszty poszczególnych wymagań. Wybór sprzętu o stosunkowo wysokim MTBF nie zwiększa znacząco kosztu całego systemu (maks. 2 razy), natomiast zastosowanie prawdziwej, pełnej redundancji czy urządzeń przystosowanych do hot swap jest zdecydowanie droższe - można powiedzieć że jest to inna klasa urządzeń. Pamiętajmy o tym zawsze dobierając sprzęt do zastosowań. Redundancja jest np. niezbędna w przypadku nadzorowania procesów rafineryjnych, gdzie każdy przestój instalacji generuje koszt rzędu milionów euro, lecz zazwyczaj chwilowa niedostępność strony internetowej średniej firmy jest akceptowalna.

WYMAGANIA WYDAJNOŚCIOWE

W następnej kolejności należy postawić wymagania wydajnościowe. Przy czym, analizując wymaganą wydajność sprzętu strukturalnego, zazwyczaj możemy pominąć aspekty związane z wydajnością podsystemu graficznego. Wydajność procesora (procesorów) w systemie infrastrukturalnym także jest analizowana nieco inaczej niż w systemach osobistych. Tutaj nie liczy się zazwyczaj prędkość wykonywania jednego programu, natomiast zależy nam na jak największej wydajności przy pracy z wieloma procesami (programami). Dodatkowo, tego typu systemy rzadko są kupowane jako systemy generalnego zastosowania, więc możemy oszacować wymaganą wydajność dzięki analizie oprogramowania, które na danym systemie będzie pracowało, oraz przewidywanego obciążenia tego systemu. Dlatego też ważne jest na tym etapie zdecydowanie, jakie oprogramowanie będzie zainstalowane na systemie komputerowym.

SYSTEMY KOMPUTEROWE W ZASTOSOWANIACH OSOBISTYCH

W tym konkretnym przypadku stwierdzenie "zastosowania osobiste" oznacza zarówno zastosowania domowe, jak i firmowe (jako komputer biurowy czy też stacja robocza). Wbrew pozorom, dobór konfiguracji takiego systemu jest trudniejszy niż w przypadku systemów infrastrukturalnych, a przynajmniej trudniejsze jest udowodnienie, iż wybrana konfiguracja spełni pokładane w niej nadzieje. Dzieje się tak, ponieważ od systemów osobistych wymaga się znacznie większej uniwersalności i najczęściej na tym etapie nie jesteśmy w stanie stwierdzić dokładnie, z jakiego oprogramowania będziemy korzystali. Dlatego też proponujemy Wam w tym przypadku podejście uproszczone, czyli odpowiedź na kilka prostych pytań, wybór głównej dziedziny zastosowań systemu komputerowego i w zależności od dokonanego wyboru - powiemy jak oszacować wydajność i jakimi przesłankami się kierować przy dokonywaniu wyboru. Oszacowanie wydajności jest zamieszczone w następnym segmencie tej lekcji.

- Jaki jest przewidywany czas korzystania z systemu komputerowego? Jest to o tyle istotne, iż w przypadku założenia, że ten czas jest dłuższy niż 2 lata, musimy liczyć się z tym, że pojawi się nowe oprogramowanie o większych wymaganiach sprzętowych. W takim przypadku albo wybieramy system komputerowy z zapasem, albo musimy liczyć się z jego późniejszą rozbudową. Jeśli musimy wybrać system komputerowy z zapasem mocy obliczeniowej, dobrym przybliżeniem jest kupienie komputera 1,5 - 2 razy szybszego, o 2 razy większej pamięci niż tego potrzebujemy na dziś.
- Czy przewidujemy rozbudowę komputera? Jeśli tak - to niestety od razu musimy zrezygnować z komputera przenośnego. Wymiana komponentów w laptopach, teoretycznie możliwa, w praktyce najczęściej jest zupełnie nieopłacalna, pomijając jedynie rozbudowę pamięci RAM. W przypadku komputera stacjonarnego powinniśmy natomiast zwrócić uwagę na dobranie odpowiedniej płyty głównej (zawierającej złącza interfejsów najnowszego standardu, z podstawką pod procesor z nowej linii produkcyjnej, z odpowiednią ilością wolnych gniazd pamięci, itp...). Powinniśmy też, by dobrać zasilacz z zapasem mocy oraz obudowę, która umożliwia nam włożenie do środka

dodatkowych urządzeń.

- Czy chcemy kupić sprzęt stacjonarny czy przenośny? Odpowiedź na to pytanie w oczywisty sposób definiuje nam, kiedy musimy wybrać laptopa (musimy mieć sprzęt przenośny). Lecz w przypadku, gdy decydujemy się na sprzęt wykorzystywany tylko i wyłącznie w jednym miejscu, czasem również warto rozważyć kupno notebook'a. Ten typ komputerów jest zazwyczaj mniejszy, cichszy, bardziej stabilny.... i droższy oraz mniej wydajny niż komputery stacjonarne.
- Do jakiej grupy zastosowań będziemy przede wszystkim wykorzystywać system komputerowy? Odpowiedź na to pytanie determinuje nam wymagania sprzętowe dla systemu komputerowego:
 1. zastosowania biurowe - edycja tekstu, obliczenia w arkuszach kalkulacyjnych, przygotowywanie prezentacji, proste bazy danych, itp...;
 2. stacja kliencka sieci komputerowej - korzystanie z internetu (www, ftp, poczta, RSS, komunikatory, itp);
 3. stacja dostępu do usług terminalowych;
 4. wyświetlanie multimedialnych - odtwarzanie muzyki i filmów;
 5. tworzenie i edycja grafiki;
 6. tworzenie i edycja multimedialnych;
 7. stacja do gier;
 8. system do nietypowych zastosowań (programowanie, CAD, sztuczna inteligencja, itd) - zastosowania nie wymienione wyżej, a wymagające specjalizowanego oprogramowania.
- Czy chcemy korzystać zawsze z najnowszego oprogramowania? Jeśli tak - to otrzymane na podstawie poprzedniego punktu wymagania sprzętowe pomóżmy przez 2 :)
- Czy system operacyjny ma dla nas znaczenie? A jeśli tak - to jaki system wybierzemy? Jeśli odpowiedź będzie pozytywna (system operacyjny ma dla nas znaczenie) i chcemy inny system niż Windows, to powinniśmy sprawdzić czy wszystkie komponenty wybrane przez nas będą pracowały pod wybranym systemem (np. Linux ma problemy z obsługą niektórych kart graficznych, kart sieci bezprzewodowej czy drukarek), bądź też po prostu wybrać sprzęt dedykowany (np. MacOS działa tylko na komputerach firmy Apple).
- Czy istotne dla nas są warunki gwarancji oraz obsługi informatycznej przez zewnętrzną firmę? Czy ważne są wzgłydy prestiżowe (szefowi firmy nie wypada pokazywać się z laptopem nieznanej marki ;)). Jeśli odpowiedź jest pozytywna, to powinniśmy ograniczyć się do sprzętu markowego (IBM, Dell, Siemens-Nixdorf, Acer, Toshiba itd...). Tylko firmy o ustabilizowanej pozycji na rynku są w stanie zapewnić długą i bezproblemową eksploatację sprzętu komputerowego.
- Jaka będzie przeciętna długość pracy przy systemie komputerowym? Odpowiedź na to pytanie jest istotna głównie przy doborze periferii. Ergonomia pracy przy komputerze jest jasna - monitor powinien być oddalony o co najmniej pół metra od oczu patrzącego. Tak więc jeśli będziemy długo pracowali przy komputerze, to nawet w przypadku laptopa warto dokupić zewnętrzny, duży monitor wraz z klawiaturą i myszką, a jeszcze lepiej - stację dokującą.
- Jakich urządzeń periferyjnych potrzebujemy i jak często będziemy z nich korzystać? Jeśli potrzebujemy drukarki, to zazwyczaj każdy model ma podawane dopuszczalne obciążenie w liczbie stron na miesiąc, a ponadto pamiętajmy, że drukarki laserowe nadal są droższe od atramentowych, co jest kompensowane przez niższy koszt wydruku jednej strony na drukarce laserowej. Jeśli będziemy często wykorzystywali skaner, to pamiętajmy, że tanie urządzenia są zazwyczaj znacznie mniej wydajne niż te klasy średniej, co oznacza że skanowanie jednej strony może zająć kilka minut, podczas gdy w sprécie profesjonalnym ten czas rzadko przekracza 1 minutę, itp...

OSZACOWANIE WYDAJNOŚCI

PRZYKŁADOWE OSZACOWANIE ŻĄDANEJ WYDAJNOŚCI SYSTEMU SERWEROWEGO

Przykład takiego oszacowania (za artykułem Piotra Wrzalika zamieszczonym w [PC Kurierze](#)).

Naszym celem jest dobranie platformy sprzętowej dla serwisu WWW, o zawartości tworzonej dynamicznie na podstawie bazy danych. Planowanie wydajności lub zdolności obsługi (*capacity*) jest procesem ciągłym i powinno się zacząć w pierwszym etapie realizacji serwisu WWW już wówczas można uzyskać pierwsze prognozy co do liczby wizyt w danym okresie czasu. To z kolei może posłużyć do stworzenia prostego i niedoskonałego jeszcze modelu analitycznego, który przy założeniu kilku empirycznych lub katalogowych wskaźników wstępnie przybliży zapotrzebowanie na wydajność.

Model uproszczony

Dla przykładu przeprowadźmy wstępna analizę dla następujących przewidywanych warunków:

- dziennie będzie składanych ok. 50 000 wizyt, każda potrwa średnio 15 minut;
- serwis będzie przygotowywany w języku PHP i w całości dynamiczny;
- użytkownik w czasie pojedynczej wizyty odwiedzi 20 stron.

Resztę parametrów możemy przyjąć domyślnie (a dokładniej - ustalić z projektantem systemu):

- średnia objętość kodu strony HTML - 7 kB;
- średnia objętość grafiki - 50 kB;
- średnia liczba grafik pobieranych z każdą stroną HTML - 4 (część wcześniej pobranych grafik przeglądarki zwykle przechowują w pamięci cache);
- stosunek liczby odwiedzin w okresie godzin szczytu (4 godziny - peak load) - 500 proc. obciążenia bazowego;
- średnia liczba skryptów PHP przetwarzana przez jeden rdzeń procesora - 20 skryptów/sekundę.
- średnie wymaganie na pamięć podczas wykonywania skryptu: ok 50 MB / skrypt.

Teraz możemy wyliczyć:

- średnia "waga" strony HTML - $7 + 4 * 50 = 207$ kB;
- średnia liczba użytkowników odwiedzających witrynę w tym samym czasie jest możliwa do obliczenia na podstawie całociowego obciążenia. Wg założeń - przez 4 godziny mamy obciążenie 5 razy większe niż bazowe, w pozostałym czasie (20 godzin) bazowe - i to składa się na 50 tys wizyt. A więc: $50\ 000 = 20*x + 4*5*x$, gdzie x - obciążenie w godzinach normalnej aktywności (bazowe);
- obciążenie normalne $x = 1250$ odwiedziny/godzinę;
- obciążenie szczytowe $5*x = 6250$ odwiedzin/godzinę;
- $1250 * 15$ minut / 60 minut = ok 313 użytkowników witryny jednocześnie w godzinach normalnej aktywności;
- $6250 * 15$ minut / 60 minut = ok 1563 użytkowników witryny jednocześnie w godzinach szczytowej aktywności;
- średnia liczba równocześnie wykonywanych skryptów ASP:
 - obciążenie normalne: $1250 * 20 / 3600$ daje ok. 7 skryptów/sekundę,
 - obciążenie szczytowe: $6250 * 20 / 3600$ daje ok. 35 skryptów/sekundę;
- średnia wymagana przepustowość serwera i sieci:
 - w godzinach normalnej aktywności: $1250 * 20 * 207$ kB = 5 054 MB/godz. = 1,4 MB/s,
 - o w godzinach szczytowej aktywności: 25 269 MB/godz. = 7 MB/s.

Wszystkie powyższe obliczenia są szacunkowe - więc często zaokrąglane w górę. Ponadto w tej analizie

w ogóle nie uwzględniamy czasu potrzebnego na odczyt danych z bazy, czy obsługę sytuacji błędnych oraz zadań administracyjnych (backup, mirroring, itp). Z tego względu, oraz ze względu na nieuniknione pomyłki w obliczeniach przyjmujemy jako docelowe obciążenie o 100 % wyższe niż maksymalne przewidywane w obecnej sytuacji, przy czym limit jest wyznaczany przez wymagania szczytowe.

Podsumowując obliczenia, możemy stwierdzić, że:

- potrzebujemy co najmniej 4 rdzeni procesora (35 skryptów / sek wymaga 2 rdzeni, zwiększone o 100 % daje 4);
- równoczesne wykonanie 70 skryptów wymaga ok. 3,5 GB RAM. Do tego należy dołożyć wymagania systemu operacyjnego (z zapasem bezpieczeństwa ok. 1 GB) - czyli potrzebujemy 4,5 GB RAM, w praktyce oznacza to co najmniej 8 GB (ze względu na typowe pojemności kości pamięci);
- Serwer musi być podłączony za pomocą łącz o przepustowości co najmniej 14 MB/s (112 Mbit/s) - więc potrzebujemy gigabitowego ethernetu;

W niniejszej analizie brak jest oszacowania na przestrzeń dyskową, wymagań odnośnie zasilania, redundancji, itp ... to powinno się znaleźć w Waszych projektach. Bardziej zależało nam na pokazaniu toku myślenia.

Model zaawansowany

W modelu uproszczonym brakuje dokładnej analizy wpływu pojedynczej wizyty na serwerze (i jej poszczególnych etapów) na ogólną wydajność systemu oraz określenia szczegółowego kosztu obsługi (*service cost*) związanego z wykorzystaniem procesora, pamięci, dysku i interfejsu sieciowego. Taka analiza jest niezbędna do zbudowania modelu wydajnościowego.

Metodologia zaawansowanego planowania wydajności powinna uwzględniać sześć kroków:

- analizę środowiska pracy serwisu;
- charakterystykę typów obciążenia;
- przygotowanie analitycznego modelu wydajnościowego;
- weryfikację modelu;
- prognozowanie obciążenia;
- zapewnienie odpowiedniej wydajności.

Do stworzenia modelu wydajnościowego niezbędna jest znajomość profilu przeciętnej wizyty użytkownika w naszym serwisie. Najlepszym źródłem tych danych są statystyki działania serwisu, jednak nie zawsze możemy sobie pozwolić na uruchomienie serwera na małą skalę, by takie statystyki zebrać i na ich podstawie przygotować się do zwiększonego obciążenia. W takiej sytuacji należy przygotować symulację przynajmniej kilkudziesięciu wizyt imitujących normalną pracę użytkownika w serwisie. Do stworzenia opisu profilu przeciętnej wizyty może nam posłużyć dowolny program analizujący logi serwera WWW, np. WebTrends Log Analyser.

WYMAGANIA WYDAJNOŚCIOWE DLA SYSTEMÓW KOMPUTEROWYCH W ZASTOSOWANIACH OSOBISTYCH

Tak jak wspominaliśmy w poprzednim rozdziale, trudno jest oszacować wymagania na moc obliczeniową dla komputerów. W zasadzie nie jest możliwe podanie dokładnych wyliczeń (jak to zrobiliśmy w paragrafie powyżej dla serwerów), jedyne co możemy zrobić, to podać ogólne wytyczne.

- Zastosowania biurowe. W tym przypadku wymagania odnośnie mocy obliczeniowej są w zasadzie definiowane poprzez system operacyjny, który chcemy wykorzystać. Oprogramowanie biurowe współpracuje nie obciąża komputera w stopniu znaczącym. W zasadzie nie obciążają go

wcale ... jedyne na co warto może zwrócić uwagę - to wielkość pamięci RAM - zazwyczaj warto mieć jej trochę więcej niż wynoszą oficjalne wymagania systemu operacyjnego. Do zastosowań biurowych możemy więc wybrać dowolny współczesny procesor (pod warunkiem że nie chcemy korzystać z Windows Vista jako naszego systemu operacyjnego ;)). Nawet najtańsze z Celeronów czy Semptronów mają wystarczającą moc obliczeniową. Podobnie w przypadku pamięci RAM. W zasadzie do zastosowań biurowych wystarczy minimum oferowane w danym czasie, co oznacza np. w początku 2007 roku 512 MB. Pojemność dysku twardego również nie ma specjalnego znaczenia, podobnie z pozostałymi podzespołami komputera (stosowanie zaawansowanego akceleratora graficznego do takich komputerów nie ma specjalnego sensu - wystarczy najtańszy układ). Reasumując, nawet najtańszy nowy komputer klasy PC dostępny na rynku może być z powodzeniem stosowany w biurach i domach do edycji dokumentów i wykonywania prostych obliczeń.

- Stacja kliencka sieci komputerowej - sytuacja w tym wypadku jest analogiczna do zastosowań biurowych.
- Stacja dostępu do usług terminalowych - podobnie jak w dwóch poprzednich przypadkach. Tutaj jedynie warto zwrócić uwagę na wydajną i szybką sieć komputerową.
- Wyświetlanie multimedialnych. Do tej pory panuje przekonanie, że wyświetlanie multimedialnych jest zadaniem obliczeniowo wymagającym. Jest w tym trochę racji. W porównaniu do zastosowań biurowych, proste wyświetlenie filmu zdecydowanie bardziej (o rząd większości) obciąża procesor niż edycja nawet skomplikowanego i dużego dokumentu Worda. Lecz musimy brać pod uwagę nieustanny rozwój i wzrost szybkości zarówno CPU, jak i pozostałych komponentów komputera. I podobnie jak w poprzednim przypadku możemy powiedzieć, że każdy współczesny procesor jest w stanie obsłużyć wyświetlanie dowolnych multimedialnych. Wielkość pamięci operacyjnej również nie ma w zasadzie znaczenia dla płynności wyświetlania filmów czy też odtwarzania muzyki - 512 MB w zupełności (z nadmiarem) wystarczy.
- Tworzenie i edycja grafiki. Tutaj pojawia się pierwsze rzeczywiście wymagające zastosowanie, i to zarówno pod względem szybkości procesora, jak i zapotrzebowania na pamięć operacyjną i przestrzeń dyskową. Zdjęcie o wysokiej rozdzielczości po rozpakowaniu w pamięci operacyjnej zajmuje kilkaset MB. Przetworzenie takiej ilości danych niestety trwa. Dlatego też w tym przypadku zalecamy wybieranie dobrych, szybkich procesorów wyposażonych w zestaw rozszerzeń SIMD możliwie najnowszej generacji. Jeśli dodatkowo zdecydujemy się na stosowanie wiodącego oprogramowania, takiego jak Adobe Photoshop, możemy mieć nadzieję, że w pełni będziemy w stanie wykorzystać nawet najnowocześniejszy procesor (tego typu oprogramowanie jako jedno z nielicznych jest w stanie w pełni wykorzystać procesory 64 bitowe czy też procesory wielordzeniowe). Nie popełnimy też błędu stwierdzając, że nawet największa ilość RAM nie będzie przesadą. Optymalna stacja do obróbki grafiki rastrowej powinna być wyposażona w co najmniej 8 GB RAM i procesor Intel lub AMD z profesjonalnej serii 64 bitowej oraz średniej wielkości (ok. 500 GB) dysk twardy. W przypadku, gdy taka konfiguracja przekracza założony budżet, sugerowalibyśmy raczej wybór wolniejszego (o niższym zegarze) procesora, niż mocne ograniczanie RAM. Paradoksalnie - stosowanie najnowocześniejszych akceleratorów graficznych nie jest do niczego potrzebne, dopóki nie zajmiemy się tworzeniem grafiki 3D. W przypadku obróbki zdjęć czy generowania grafiki dwuwymiarowej - średniej klasy układ graficzny renomowanej firmy będzie optymalnym wyborem.
- Tworzenie i edycja multimedialnych jest kolejnym wymagającym obliczeniowo zadaniem. Podstawowa różnica w stosunku do tworzenia grafiki polega na tym, że w tym wypadku wydajność procesora oraz przepustowość (szybkość) podsystemu dyskowego jest ważniejsza niż ilość dostępnej pamięci RAM. Powinniśmy także zainwestować w największy dostępny dysk twardy o dużej szybkości, lub dwa takie dyski pracujące równolegle, a przy wyborze karty grafiki zwrócić uwagę na te modele, które mają wbudowane sprzętowe wspomaganie kompresji video.
- Stacja do gier - tutaj nie ma ograniczeń.... niestety. Nawet najszybszy i najdroższy komputer szybko okaże się zbyt wolnym. Jest to zdecydowanie najbardziej wymagające zastosowanie dla komputerów osobistych, dlatego też powinniśmy przygotować się na wysoki wydatek. Potrzebujemy bardzo dobry procesor, ok. 8 GB RAM, duży dysk twardy oraz - co jest dość drogie - akcelerator graficzny najnowszej generacji. Szybko okaże się, że koszt takiego komputera zbliża się do ceny tanich samochodów ... Z praktycznego punktu widzenia wydaje się, że rozsądniej jest wybierać modele o oczko poniżej najlepszego dostępnego - czyli drugi od góry

procesor pod względem zegara, dobrą kartę grafiki (ale nie najszybszą istniejącą na rynku, lecz tą zaraz za nią), itp. W tym przypadku stosunek ceny do wydajności jest stosunkowo dobry. Należałoby również poważnie zastanowić się nad ewentualnością wyboru ... konsoli do gier. Po pierwsze - są one tańsze niż komputery. Po drugie - zdecydowanie wolniej się starzeją. Po trzecie - niekiedy są wydajniejsze od najlepszych PC. Przykładowo: Sony PlayStation 3 jako centralny procesor wykorzystuje układ Cell zaprojektowany wspólnie przez firmy IBM, Sony i Toshiba. Składa się on z głównego 64-bitowego procesora PowerPC (tzw. "Power Processing Element" lub PPE) oraz ośmiu rdzeni typu ""Synergistic Processing Elements" (SPEs) - czyli mamy procesor o 9 rdzeniach ... w porównaniu z nim procesory stosowane w stacjach roboczych wypadają raczej słabo ;). Cały układ taktowany jest zegarem 3,2 Ghz, co w sumie pozwala na przypuszczenie, iż taka konsola będzie całkiem sprawnie obsługiwać nowe gry jeszcze za 5 lat.

- System do nietypowych zastosowań (programowanie, CAD, sztuczna inteligencja, itd) - mowa o zastosowaniach nie wymienionych wyżej, a wymagających specjalizowanego oprogramowania. Tu sprawę mamy ułatwioną - po prostu musimy sprawdzić jakie wymagania sprzętowe ma oprogramowanie, które chcemy wykorzystywać i odpowiednio do nich dobrać sprzęt (jak to zrobiliśmy w przykładzie zawartym w pierwszym segmencie tej lekcji). Nie należy także zapominać o elementach nie wynikających wprost z wymagań oprogramowania, lecz z analizy przypadku (np 2-3 monitory o wysokich przekątnych do typowych stacji CAD).

Jak widzicie, każde z zastosowań komputerów osobistych ma swoje specyficzne wymagania i to w zależności od nich powinniśmy dobierać system komputerowy. Komputery PC są rozwiązaniami uniwersalnymi, co wcale nie znaczy że optymalnymi - do każdego zadania optymalna konfiguracja będzie nieco inna. Sporo też można zaoszczędzić analizując swoje potrzeby i dobierając system który je spełnia w miejsce ulegania magii gigaherzów i megabajtów z reklam.

PODSTAWOWE INFORMACJE O SYSTEMIE OPERACYJNYM

WSTĘP

Zakończyliśmy właśnie część podręcznika poświęconą sprzętowi komputerowemu, od tej pory głównym tematem naszych rozważań będzie oprogramowanie. Oprogramowanie podstawowe i kluczowe dla działania systemu komputerowego - czyli system operacyjny.

Wprowadzenie do tej niezwykle bogatej tematyki jaką są systemy operacyjne komputerów zaczniemy od podstawowych informacji dotyczących ich ogólnej charakterystyki, roli i zadań oraz budowy wewnętrznej (to ta lekcja). Potem postaramy się opisać nieco dokładniej kluczowe mechanizmy systemu operacyjnego, a cały blok zakończymy przedstawieniem współczesnych systemów operacyjnych oraz konfiguracją stacji roboczej.

Chcielibyśmy uświadomić Wam rolę systemu operacyjnego w funkcjonowaniu wszelkiego rodzaju systemów komputerowych. Chcielibyśmy także, byście rozumieli, jak są zbudowane najpopularniejsze systemy takie jak Windows czy Linux, byście potrafili sami ocenić, które ze specyficznych rozwiązań w nich zawartych są dla Was istotne, i na koniec - by w trakcie Waszej dalszej edukacji informatycznej podstawowe pojęcia z tej dziedziny - takie jak proces, wątek czy jądro - były dla Was jasne.

CO TO JEST SYSTEM OPERACYJNY?

Oto najkrótsza możliwa definicja: system operacyjny (ang. *operating system*) jest **najważniejszym** programem dla naszego komputera ;). A bardziej formalnie, rozpoczęliśmy od przytoczenia definicji systemu operacyjnego podanej w książce A.Silberschatza, P.Galvina "Podstawy systemów operacyjnych" (WNT 2000):

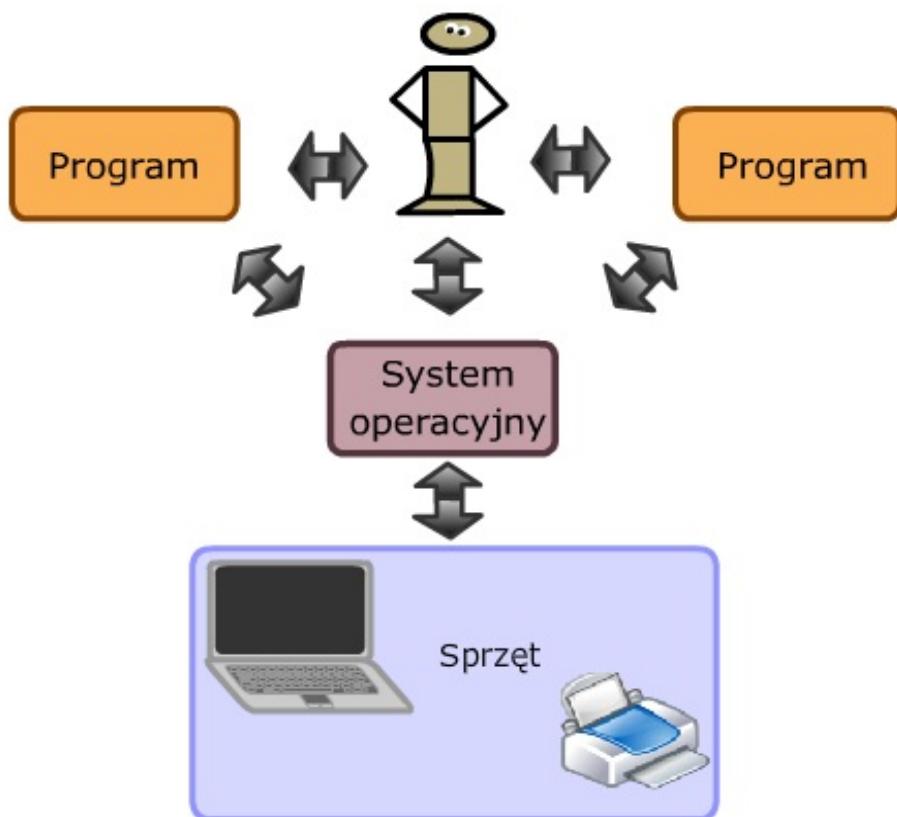
System operacyjny jest programem, który działa jako pośrednik między użytkownikiem komputera a sprzętem komputerowym. Zadaniem systemu operacyjnego jest tworzenie środowiska, w którym użytkownik może wykonywać programy. [...] Możemy uważać system operacyjny za dystrybutora zasobów. System komputerowy ma wiele zasobów (sprzęt i oprogramowanie), które mogą być potrzebne do rozwiązania zadania. [...] System operacyjny pełni funkcję zarządcy owych dóbr i przydziela je poszczególnym programom i użytkownikom wówczas, gdy są one nieodzowne do wykonywania zadań.

Jest to jedna z wielu definicji, które można znaleźć w literaturze. Pojęcie systemu operacyjnego jest trudne do zdefiniowania w zwartej, lakonicznej formie. Najczęściej krótki opis jest zbyt ogólny, żeby uzyskać wyobrażenie o roli i sposobie działania tego specyficznego oprogramowania. Podobnie, trudne jest określenie elementów składowych systemu operacyjnego, czyli jednoznaczne oddzielenie oprogramowania systemowego od aplikacyjnego.

Zazwyczaj najważniejszą cechą systemu operacyjnego (tym, co go odróżnia od innych programów), jest owo "pośredniczenie" między użytkownikiem komputera a sprzętem komputerowym. Jest on niezbędny do prawidłowej pracy komputera i w pewnym sensie określa możliwości wykorzystania sprzętu oraz komfort pracy użytkownika. System operacyjny musi być oczywiście dostosowany do sprzętu, na którym jest instalowany. Od niego zależy również w dużym stopniu jaką pracę możemy wykonać lub jakie programy uruchomić. Wprowadzenie systemu operacyjnego jako warstwy

pośredniczącej, oprócz zdjęcia z programistów końcowych niebanalnego problemu oprogramowywania (w ramach każdej aplikacji!) sprzętu, pozwoliło na wprowadzeniu pewnej standaryzacji dzięki czemu np. kolejne linie urządzeń są wzajemnie kompatybilne i aplikacje końcowe działające na jednym modelu można najczęściej uruchamiać na jego następcach, bez względu na różnice sprzętowo architektoniczne. Dobrym przykładem tego ułatwienia jest np. dowolny program do odtwarzania muzyki z plików mp3 działający pod kontrolą systemu Microsoft Windows – programista nie musi przewidywać jaki model karty dźwiękowej znajdzie się w komputerze na którym program ma działać, nie musi też zastanawiać się jakich przerwań i adresów IO dana karta używa, chcąc odtwarzać dźwięk odwołuje się do funkcji systemu operacyjnego, który sam przekierowuje dane do odpowiedniej karty w odpowiedni sposób.

Dzięki zastosowaniu systemu operacyjnego możliwe jest łatwe wykorzystanie komputerów o różnej architekturze wewnętrznej, zbudowanych z różnych podzespołów różnych producentów, gdyż z punktu widzenia użytkownika obsługa wszystkich podzespołów przeznaczonych do wykonywania jednego zadania jest taka sama. Szczegółami obsługi urządzeń wewnętrznych zajmuje się właśnie system operacyjny. Łatwiej mają również twórcy oprogramowania, którzy nie muszą dostosowywać swoich programów do wielkiej liczby różnorodnych rozwiązań sprzętowych dostępnych na rynku.



System operacyjny jako pośrednik

Podstawą wszystkich systemów operacyjnych jest wykonywanie podstawowych zadań takich jak: kontrolowanie i przypisywanie pamięci, ustalanie priorytetów w zadaniach, obsługa urządzeń, ustalanie połączeń sieciowych oraz zarządzanie plikami. Większość systemów operacyjnych posiada środowiska graficzne ułatwiające ich obsługę.

Nie ma precyzyjnego określenia, które składniki wchodzą w skład systemu operacyjnego jako jego części. Najczęściej akceptuje się definicję „marketingową”, zgodnie z którą to wszystko, co producent udostępnia w ramach zbioru oprogramowania nazywanego systemem operacyjnym, stanowi jego część. Lecz że tak pojedyncza definicja nie jest zawsze najlepszym wyborem, przekonała się np. firma Microsoft. Poprzez dołączanie typowych aplikacji użytkowych (takich jak Internet Explorer, Outlook czy też WordPad) do systemu, naraziła się na zarzut praktyk monopolistycznych.

W ogólnym przypadku w strukturze systemu operacyjnego wyróżnia się jądro oraz programy systemowe, które dostarczane są razem z systemem operacyjnym, ale nie stanowią integralnej części jądra. Jądro jest zbiorem modułów, które ukrywają szczególne szczegółowe realizacji systemu komputerowego, udostępniając pewien zestaw usług, wykorzystywanych między innymi do implementacji programów systemowych. W dalszej części system operacyjny będzie rozumiany głównie jako jądro, ewentualnie inne elementy oprogramowania integralnie związane z funkcjonowaniem jądra.

Z punktu widzenia kontaktu z użytkownikiem istotny jest interpreter poleceń, który może być częścią jądra lub programem systemowym (np. jest tak w systemie UNIX). Interpreter wykonuje pewne polecenia wewnętrznie, tzn. moduł lub program interpretera dostarcza implementacji tych poleceń. Jeśli interpreter nie może wykonać wewnętrznie jakiegoś polecenia, uruchamia odpowiedni program (tzw. polecenie zewnętrzne), jako odrębny proces.

Nie będziemy zagłębiać się zbytnio w strukturę systemu operacyjnego - będącie mieli tą tematykę rozwiniętą w przedmiocie poświęconym tylko systemom. Natomiast postaramy się przedstawić jego funkcjonalność oraz właściwości użytkowe.

FUNKCJE SYSTEMU KOMPUTEROWEGO

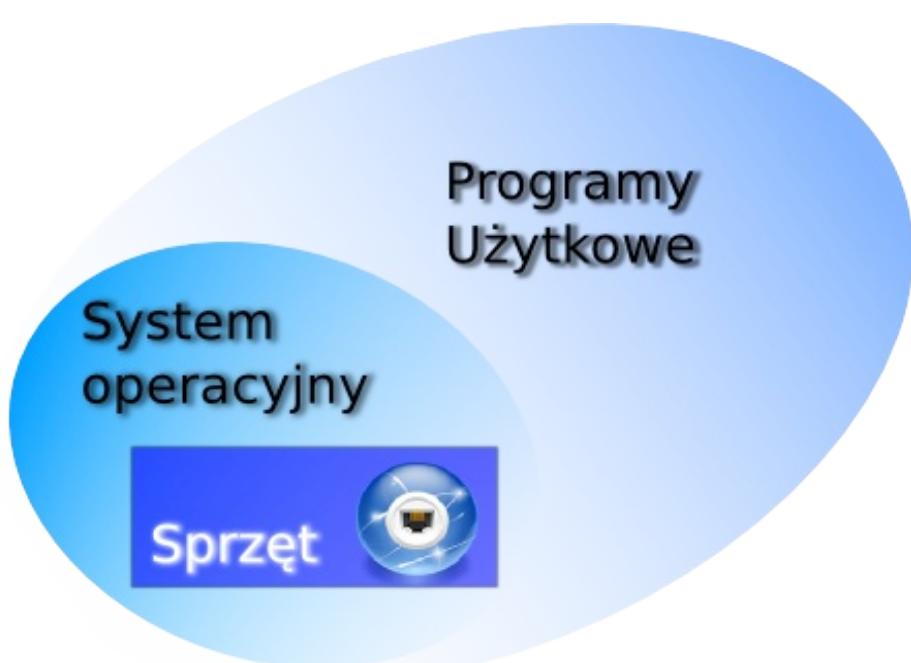
Komputer bez oprogramowania jest jedynie bezużyteczną skrzynką. Aby tchnąć w niego życie trzeba dodać do niego myśl człowieka zawartą w oprogramowaniu (co nie znaczy, że w konstrukcji sprzętu tej myśli nie ma zawartej). Dopiero połączenie tych dwóch elementów – sprzętu (ang. *hardware*) i oprogramowania (ang. *software*) – tworzy środowisko, w którym można wykonywać różnorodne zadania.

Sprzęt i oprogramowanie tworzą system komputerowy

$$\text{Sprzęt} + \text{Oprogramowanie} = \text{System komputerowy}$$

Funkcje systemu operacyjnego można rozpatrywać z różnych punktów widzenia. Lecz zanim przejdziemy do próby wyodrębnienia najważniejszych funkcji, zdefiniujmy główne zadania dla systemu operacyjnego:

- **zarządzanie zasobami systemu komputerowego.** System operacyjny pośredniczy pomiędzy użytkownikiem a sprzętem, dostarczając wygodne środowisko do wykonywania programów. Użytkownik końcowy korzysta z programów (aplikacji), na potrzeby których przydzielane są zasoby systemu komputerowego. Przydziałem tym zarządza system operacyjny, dzięki czemu można uzyskać stosunkowo duży stopień niezależności programów od konkretnego sprzętu oraz odpowiedni poziom bezpieczeństwa i sprawności działania;
- **stworzenie środowiska wygodnego dla użytkownika.** Z tego punktu widzenia zależy nam, aby jak najbardziej odseparować użytkownika od technicznych aspektów sprzętu komputerowego i ukryć przed nim zasadę jego działania (sami możecie ocenić na ile się to wspólnie udało). Przy czym jako użytkownika systemu operacyjnego rozumie się tutaj także programistów tworzących dla danego systemu aplikacje.



Efektywność zarządzania zasobami oraz wygodny interfejs dla użytkownika są dwoma ogólnymi, niezależnymi celami projektowymi systemów operacyjnych. Pierwszy z tych celów był kluczowy w rozwoju rodziny systemów uniksowych. Dopiero w późniejszych etapach ich rozwoju pojawił się intuicyjny okienkowy interfejs użytkownika. Systemy rodziny MS Windows zorientowane były natomiast przede wszystkim na interfejs użytkownika, na bazie którego w późniejszych etapach rozwoju powstawał pełnowartościowy system operacyjny, uwzględniający szerzej rozumiane zarządzanie zasobami.

ZARZĄDZANIE ZASOBAMI

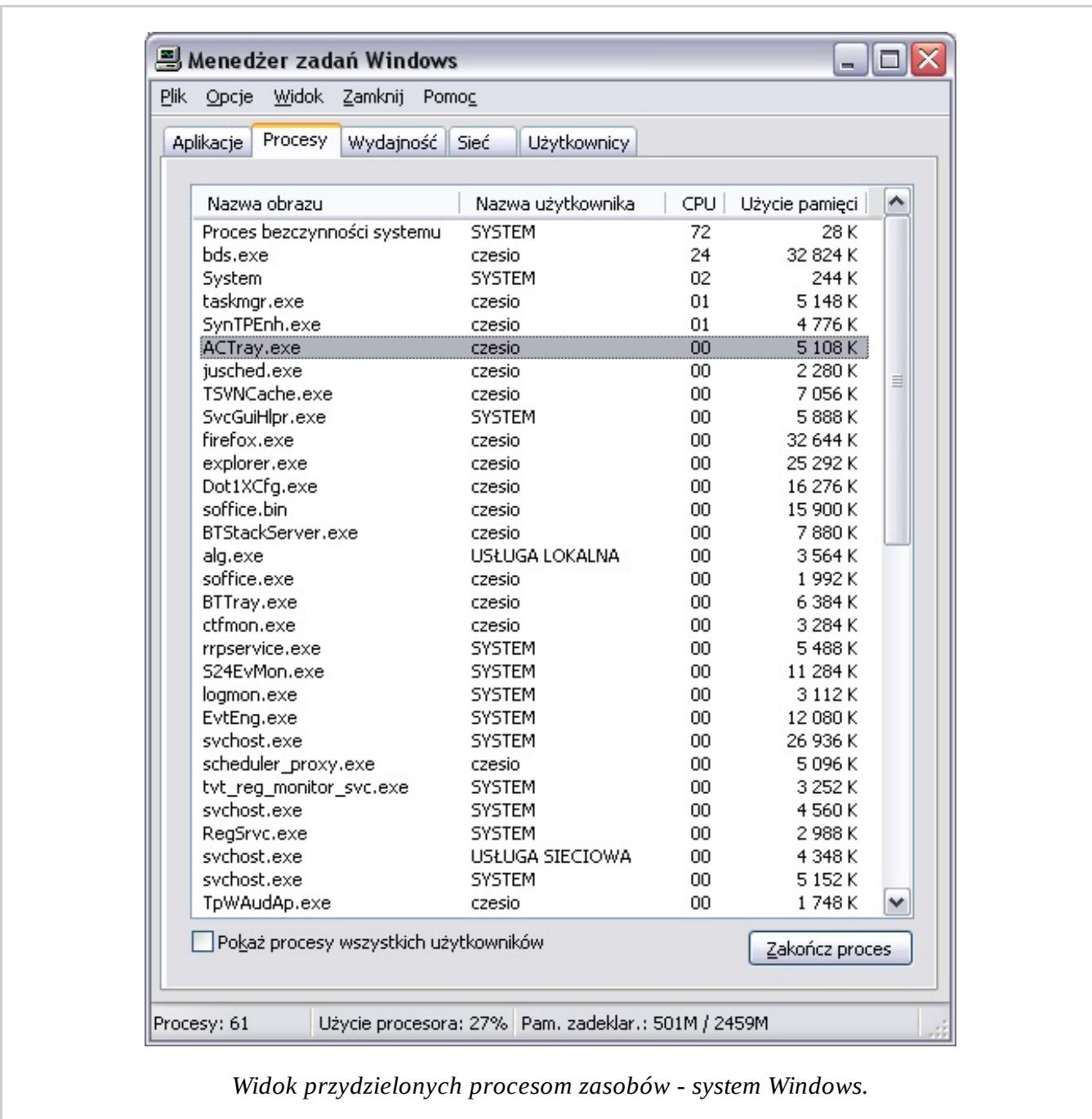
System operacyjny zarządza pracą wszystkich elementów wchodzących w skład systemu komputerowego. Wszystkie te elementy mają określone zasoby.

Formalne pojęcie zasobu wprowadzimy wraz z pojęciem procesu. Na razie zasób będzie rozumiany intuicyjnie jako element systemu komputerowego istotny, czy wręcz kluczowy dla realizacji przetwarzania danych. Tak więc zasobem dla systemu operacyjnego jest np.: czas procesora, pamięć operacyjna, dostęp do urządzeń wejściowych i wyjściowych, dostęp do pamięci masowej. Zazwyczaj funkcja zarządzania zasobami nie jest bezpośrednio wykorzystywana przez użytkownika (czasami nie jest przez niego w ogóle dostrzegana) - wykorzystują ją programy użytkowe (aplikacje). Celem tej funkcji jest optymalizacja wykorzystania zasobów przez użytkowników.

W ramach zarządzania ogólnie rozumianymi zasobami możemy wyróżnić następujące operacje:

- **przydział zasobów:** realizacja żądań dostępu do zasobów w taki sposób, że zasoby używane są zgodnie z intencją użytkowników (np. zagwarantowanie wyłącznego dostępu drukarki, żeby wydruk mojego wielostronnicowego dokumentu nie był przetykany stronami Excela kolegi);
- **planowanie dostępu do zasobów:** strategia przydziału zasobów gwarantująca bezpieczeństwo, żywotność, brak zakleszczenia, sprawiedliwość oraz optymalność ich wykorzystania. Zwróćcie uwagę na odróżnienie planowania od samego przydziału — przydział oznacza powiązanie zasobu z realizowanym zadaniem, podczas gdy planowanie wiąże się z podejmowaniem decyzji odnośnie wyboru zdania, któremu zasób będzie przydzielony. I może się zdarzyć taka sytuacja, że ze względu na brak wolnej pamięci jakiś program nie zostanie uruchomiony;
- **ochrona i autoryzacja dostępu do zasobów:** dopuszczanie możliwości użytkowania zasobu tylko przez osoby (procesy) uprawnione i w zakresie przydzielonych im uprawnień. Przez to należy rozumieć zarówno nadanie praw do druku tylko poszczególnym osobom, jak i np. zabronienie programowi A modyfikacji tego obszaru pamięci, który został przydzielony programowi B (obejście tego zabezpieczenia jest najczęściej celem twórców wirusów);
- **odzyskiwanie zasobów:** dołączanie zwolnionych zasobów do zbioru zasobów wolnych po zakończeniu ich użytkowania. Po zakończeniu pracy aplikacji użytkownika, cała przydzielona jej pamięć jest spowrotem zwacana do wspólnej puli i może być przydzielona innej aplikacji;
- **rozliczanie:** rejestrowanie i udostępnianie informacji o wykorzystaniu zasobów w celach kontrolnych i rozrachunkowych.

Brzmi to jak tekst ekonomisty ... trochę w tym prawdy. System operacyjny jest więc Centralnym Ekonomistą Komputera :).



Widok przydzielonych procesom zasobów - system Windows.

Typowymi zasobami zarządzanymi przez system operacyjny są: procesor, pamięć operacyjna, pamięć masowa (system plików) i urządzenia wejścia-wyjścia. Zarządzanie każdym z nich wymaga specyficznego podejścia.

Procesor jest zasobem współdzielonym przez wiele procesów (wiele programów pracujących razem), w związku z czym zadaniem systemu operacyjnego jest przydział kwantu czasu procesora i wywłaszczanie zadania, które:

- wykorzystało już swój czas;
- nie może kontynuować przetwarzania ze względu na brak innych zasobów (np. brak gotowości urządzeń wejścia-wyjścia);
- ma zbyt niski priorytet.

Wspomniane wywłaszczanie polega na zatrzymaniu działania programu: przeniesieniu go w swego rodzaju "stan uśpienia" oraz "obudzenie" innego, czekającego w kolejce na swój cykl. Z tego punktu widzenia zarządzanie czasem procesora jest zadaniem **cyklicznym**.

Pamięć jest zasobem, który przydzielany jest na wyłączność danego przetwarzania. Zadaniem systemu jest zatem utrzymywanie informacji o zajętości przestrzeni adresowej, znajdowanie i przydzielanie odpowiednich fragmentów wolnej pamięci na żądanie aplikacji użytkownika lub innych modułów systemu operacyjnego, jak również reagowanie na naruszenie ochrony pamięci. W przeciwieństwie do zarządzania czasem procesora - w przypadku pamięci system operacyjny nie ma możliwości "wywłaszczenia" procesu z przydzielonej mu pamięci - raz przyznanej pamięci nie może odebrać, dopóki proces sam jej nie zwolni, lub też nie zostanie przez system zatrzymany (w tym wypadku mówimy o "zabiciu" procesu) - tak więc przydział pamięci nie może być uznawany za zadanie cykliczne.

```

czesio@iair341d: ~ - Powłoka - Konsola
Sesja Edycja Widok Zakładki Ustawienia Pomoc
top - 18:09:25 up 4:59, 1 user, load average: 0.59, 0.34, 0.
Tasks: 116 total, 1 running, 115 sleeping, 0 stopped, 0 z
Cpu(s): 5.0%us, 1.0%sy, 0.0%ni, 94.0%id, 0.0%wa, 0.0%hi,
Mem: 1035740k total, 1010704k used, 25036k free, 57720
Swap: 578300k total, 0k used, 578300k free, 556732

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+
22394 czesio 15 0 33072 17m 13m S 4 1.7 0:13.29
5234 root 15 0 183m 35m 5200 S 2 3.5 9:22.91
22526 czesio 15 0 33620 19m 16m S 2 1.9 0:01.89
22396 czesio 15 0 2908 1312 908 S 1 0.1 0:02.35
16118 czesio 15 0 31124 12m 9468 S 1 1.2 0:16.57
16122 czesio 15 0 38964 21m 16m S 1 2.1 0:22.21
16110 czesio 15 0 38452 18m 14m S 0 1.9 0:17.48
16152 czesio 15 0 30004 16m 13m S 0 1.6 0:00.58
17695 czesio 15 0 67408 19m 14m S 0 1.9 0:10.31
22488 czesio 15 0 34356 15m 12m S 0 1.5 0:01.67
1 root 15 0 2912 1844 524 S 0 0.2 0:01.50
2 root RT 0 0 0 0 S 0 0.0 0:00.02
3 root 34 19 0 0 0 S 0 0.0 0:00.00
4 root RT 0 0 0 0 S 0 0.0 0:00.00
5 root RT 0 0 0 0 S 0 0.0 0:00.05
6 root 39 19 0 0 0 S 0 0.0 0:00.00
7 root RT 0 0 0 0 S 0 0.0 0:00.00
8 root 10 -5 0 0 0 S 0 0.0 0:00.27
9 root 10 -5 0 0 0 S 0 0.0 0:00.00
10 root 12 -5 0 0 0 S 0 0.0 0:00.10
11 root 11 -5 0 0 0 S 0 0.0 0:00.00

```

Przykład wyświetlenia aktualnego przydziału zasobów w systemie Linux

Oprócz prostego przydzielania czasu procesora czy też pamięci, system operacyjny wykonuje także inne zadania powiązane z zarządzaniem zasobami:

- udostępnia środowisko do wykonywania programów : system operacyjny dostarcza struktur danych do organizacji wykonywania programu oraz zachowywania i odtwarzania jego stanu. System operacyjny udostępnia też programistom mechanizmy komunikacji pomiędzy programami (kolejki komunikatów, strumienie, pamięć współdzielona) oraz mechanizmy synchronizacji procesów (np. semafory);
- steruje urządzeniami wejścia-wyjścia: odpowiednie moduły sterujące, integrowane z systemem operacyjnym, inicjalizują pracę urządzeń zewnętrznych oraz pośredniczą w efektywnym przekazywaniu danych pomiędzy jednostką centralną a tymi urządzeniami;
- gwarantuje obsługę podstawowych błędów: system operacyjny reaguje na błędy użytkowników

- (np. niedostępność zasobów, brak prawa dostępu), programistów (np. błąd dzielenia przez 0, naruszenie ochrony pamięci, nieprawidłowy kod rozkazu) oraz systemu (np. błąd braku strony, błąd magistrali);
- ukrywa przed aplikacjami skomplikowaną budowę sprzętu komputerowego oraz różnice pomiędzy poszczególnymi urządzeniami przez tworzenie abstrakcji, np.:
 - zbiory zapisanych klastrów na dysku widziane są jako pliki o symbolicznych nazwach;
 - każda karta graficzna może być przez programistę traktowana jako dwuwymiarowa tablica pikseli, którym można nadać jedną z 16 milionów barw (niezależnie od trybu graficznego który mamy ustawiony). System operacyjny sam przetworzy obraz i dostosuje go do wyświetlenia na zainstalowanym sprzęcie.

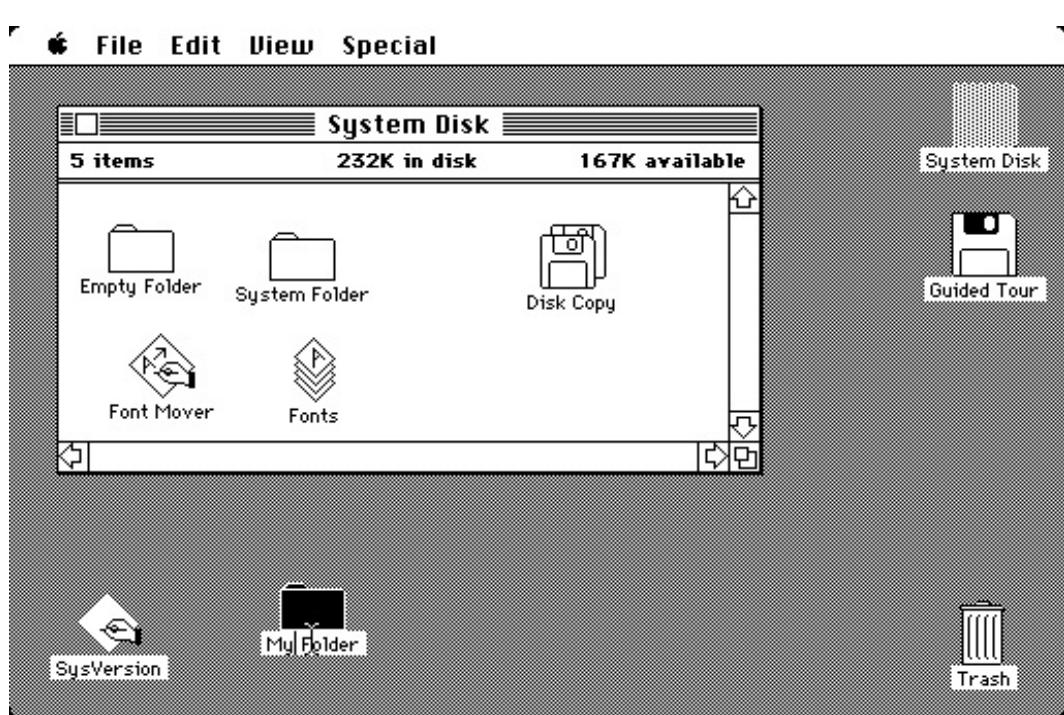
INTERFEJS UŻYTKOWNIKA

INTERFEJS UŻYTKOWNIKA

W początkowych rozwiązańach systemów operacyjnych interfejs użytkownika był traktowany po macoszemu. Królował terminal tekstowy (inaczej zwany konsolą), w którym można było wpisywać kolejne polecenia dla systemu. I zawsze trzeba było wiedzieć, co należy napisać ... to był problem ;)

Dlatego też przez długi okres czasu systemy komputerowe były rzeczywiście trudne w obsłudze i przeznaczone raczej dla dobrze przygotowanych użytkowników. Konsola pozwalała tylko na wprowadzanie i wyświetlanie tekstu, w najbardziej prymitywnych rozwiązańach wyświetlanie możliwe było tylko linia po linii.

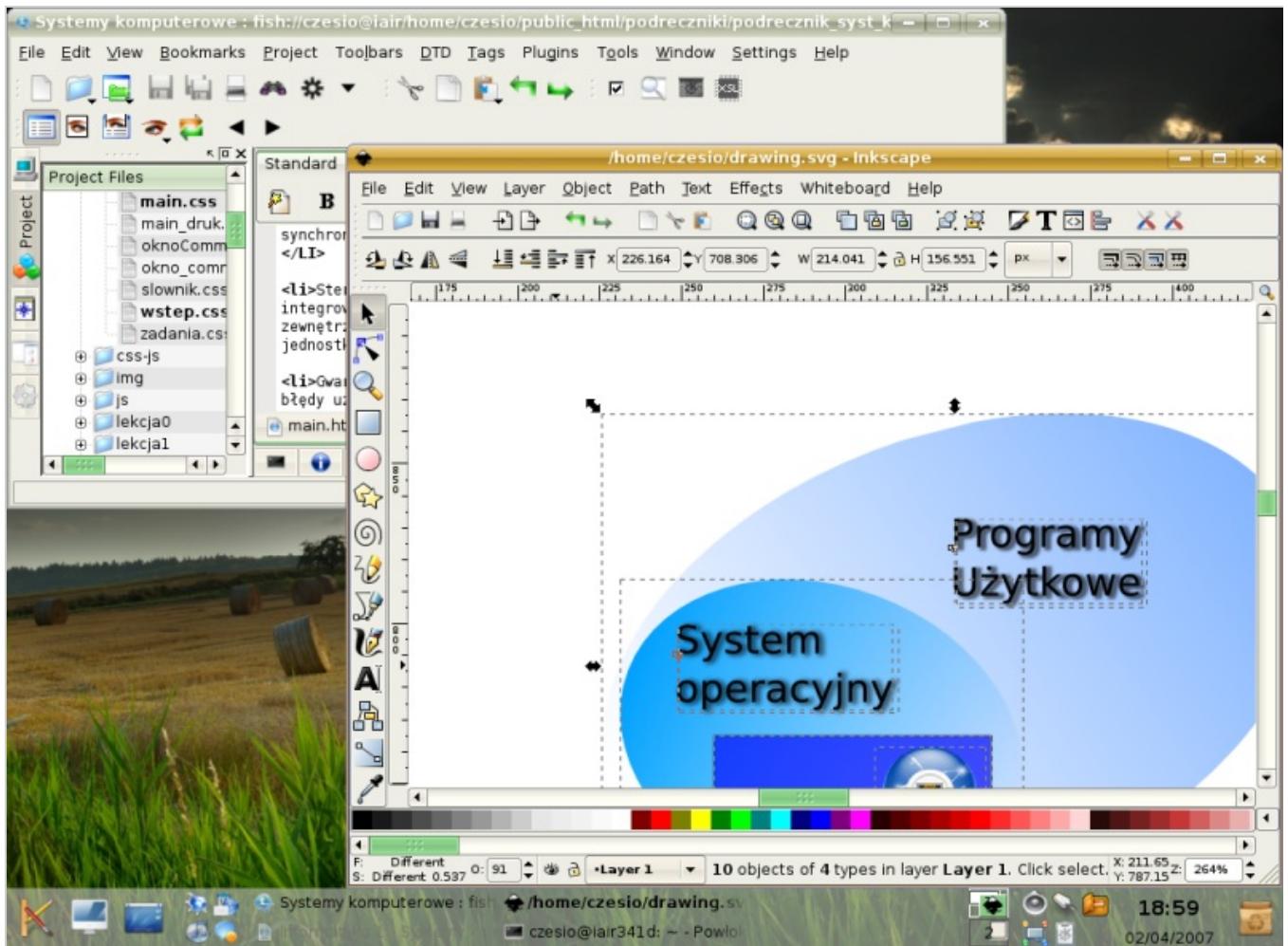
Przełomem, który pozwolił na rzeczywiste zadomowienie się systemów komputerowych "pod strzechami", było pojawienie się graficznych interfejsów użytkownika (ang. *Graphical User Interface*, GUI). W roku 1979 Steve Jobs zwiedzając laboratoria Xerox PARC dostrzegł potencjał opracowanego tam systemu komputerowego z graficznym interfejsem użytkownika (GUI). Xerox nie uznał tego projektu za ważny natomiast Jobs wkrótce rozpoczął pracę nad nowym systemem z GUI w swojej macierzystej firmie Apple Computers Inc. System ten został nazwany Apple Lisa, nie odniósł jednak sukcesu komercyjnego z racji wysokiej ceny i braku wystarczająco zróżnicowanego oprogramowania. Jobs, straciwszy na skutek wewnętrznych tarć dostęp do Apple Lisa rozpoczął pracę nad niskobudżetowym komputerem z GUI. Tak powstał Apple Macintosh. Te dwa komputery i ich systemy operacyjne z graficznym interfejsem stały się prekursorami współczesnych systemów operacyjnych. Upowszechnione przez nie takie standardy graficznego interfejsu jak okna, rozwijalne menu, kurSOR myszy, kosz na niepotrzebne pliki stały się od tej pory podstawą współczesnych systemów operacyjnych.



Pierwszy upowszechniony na szeroką skalę GUI - MacOS

Środowisko graficzne jest grupą wzajemnie współpracujących programów (części systemu operacyjnego), zapewniającą możliwość wykonywania podstawowych operacji na komputerze (takich jak uruchamianie programów, poruszanie się po katalogach, zmiana konfiguracji systemu itp.) w trybie graficznym, najczęściej okienkowym. Zapewnia alternatywny dla konsoli sposób pracy na komputerze.

Najważniejszym elementem graficznego interfejsu jest okno programu (lub kilka takich okien, choć takie konfiguracje są niezbyt lubiane przez użytkowników). Wewnątrz takiego okna są rozmieszczone elementy interakcyjne, zwane widgetami (lub "kontrolkami", nawiązując do pulpitu sterowniczych). Użytkownik komunikuje się z aplikacją pośrednio przez te widgety najczęściej za pomocą myszy i klawiatury. Mysz jest odpowiedzialna za przesuwanie tzw. kurSORA myszy, wskazującego odpowiednią pozycję na ekranie, a naciskanie przycisków jest związane z obszarem, w którym zawiera się aktualna pozycja kurSORa. Klawiatura jest związana z kolei z pojęciem tzw. "skupienia". Skupienie jest stanem, który może posiadać w jednym czasie dokładnie jeden widget w całym systemie okienkowym. Jeśli użytkownik nacisnął klawisz na klawiaturze, to wtedy informacja o naciśnięciu klawiszy przekazywana jest do tego widgetu, który aktualnie "posiada skupienie" lub "jest w stanie skupienia".



KDE - jeden z najlepszych GUI dla Unix-ów

BUDOWA SYSTEMU OPERACYJNEGO

Obarczyliśmy więc system operacyjny zadaniami. Z lektury poprzedniego rozdziału wynika, że jest ich niemal ... podobnie jak niemała jest liczba możliwych podzespołów sprzętowych, które należy obsłużyć. Z tego względu system operacyjny jest dziś najczęściej systemem dużym i złożonym. To skomplikowany twór, który musi zapewniać z jednej strony dużą niezawodność i stabilność działania, a z drugiej łatwość rozbudowy i modyfikacji (na przykład dostosowania do szybko zmieniających się rozwiązań sprzętowych).

Dlatego też w praktyce nie spotyka się już systemów w pełni monolitycznych (rozwiążanie kiedyś popularne, szczególnie dla 8-bitowych komputerów). Natomiast często spotyka się opis systemu za pomocą modelu warstwowego.

MODEL WARSTWOWY

W takim przypadku możemy rozrysować sobie komputer w następujący sposób:



Model warstwowy systemu komputerowego

Na samym dole mamy sprzęt, potem kolejne warstwy abstrakcji, z których każda wykorzystuje możliwości tej, która jest poniżej i udostępnia swoją funkcjonalność wyżej, stanowiąc jednocześnie pewnego rodzaju granicę - ukrywając niepotrzebne szczegóły przed warstwami wyższymi.

Mały przykład: rozważmy w jaki sposób obsługiwany jest dysk twardy. Jak wiecie z poprzednich lekcji, dysk twardy jest to zespół okrągłych talerzy pokrytych substancją magnetyczną, nad którymi poruszają się głowice. Tylko jak je zmusić do ruchu? W warstwie fizycznej realizują to mikrosilniki krokowe. Lecz z punktu widzenia systemu operacyjnego jest to zbyt skomplikowane. Dlatego też producenci dysków twardych wyposażają je w tzw. firmware - oprogramowanie bardzo niskopoziomowe, zaszyte w pamięci ROM urządzenia, które zawiera procedury sterowania silnikami krokowymi. Aby z nich korzystać, nie trzeba znać szczegółów budowy silnika, napięć sterujących, itp, itd. Wystarczy wywołać taką procedurę z numerem cylindra, nad którym ma się znaleźć głowica. Tak więc, dla komputera (warstwa assemblera) pewne, nieistotne z jego punktu widzenia, szczegóły

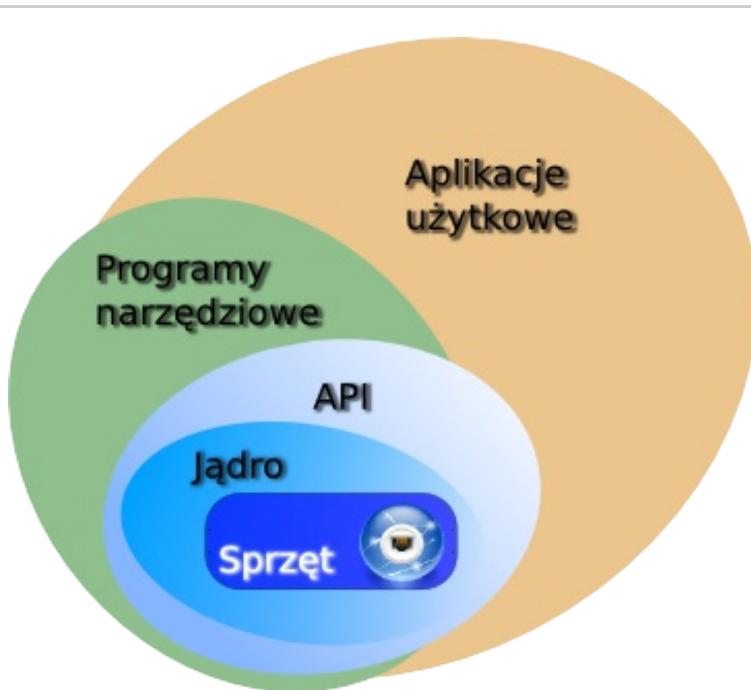
zostały ukryte poprzez warstwę firmware. Idąc dalej tym krokiem - jądro wykorzystuje procedury firmware wewnętrz swoich procedur, które zapisują logiczne struktury danych (czyli pliki) w fizycznych blokach i sektorach HDD. Z punktu widzenia aplikacji i finalnego użytkownika znów pewne nieistotne szczegóły zostały ukryte. Aby skorzystać z tych procedur, wydajemy proste polecenia - utworzenia, zapisania, skasowania pliku na dysku. Jądro systemu przetworzy te polecenia na ciąg poleceń pozycjonujących i sterujących głowicami HDD, a firmware przetworzy te polecenia na natężenia prądów w silnikach i głowicy.

Dostaliśmy więc coś w rodzaju radia (na zewnątrz mamy tylko pokrętła, które pozwalają nam na słuchanie muzyki i nie musimy wiedzieć co się dzieje wewnętrz) - mamy dostęp do plików, które można zapisywać i odczytywać bez konieczności posiadania wiedzy, co się dzieje fizycznie w maszynie nazywanej komputerem. Co więcej - dostaliśmy też abstrakcję - system plików. I z punktu widzenia aplikacji użytkownika nie jest istotne gdzie jest on umieszczony fizycznie. Korzystając z MS Word tak samo zapisujemy plik na pen-drive jak na HDD, mimo że fizyczna zasada działania tych dwóch urządzeń jest kompletnie różna - lecz aplikacja w ogóle o tym nie wie.

Na tym zakończymy omawianie modelu warstwowego. Jest on przydatny w analizie i budowie systemów komputerowych, jednakże jak na potrzeby tego podręcznika - zbyt abstrakcyjny i teoretyczny. W jego miejsce proponujemy przyjąć uproszczony schemat systemu operacyjnego.

SCHEMAT BUDOWY SYSTEMU OPERACYJNEGO

W podejściu pokazanym poniżej ograniczamy liczbę warstw i jednocześnie pokazujemy która warstwa może korzystać z której:



W podobny sposób można traktować wszystkie współczesne systemy operacyjne.

- Jądro systemu (występuje w kilku odmianach, opisanych poniżej) obsługuje sprzęt. I jako jedyny program w systemie ma prawo na bezpośrednią interakcję ze sprzętem. Oczywiście ten postulat dotyczy systemów idealnych, lecz o ile nie będącie wykorzystywali nietypowych urządzeń

peryferyjnych - będzie spełniony dla większości popularnych systemów operacyjnych.

- API udostępnia funkcje jądra aplikacjom, w tym powłoce systemowej.
- Programy narzędziowe (w tym powłoka) są pierwszym zewnętrznym "oknem na świat" systemu operacyjnego, z którym może zetknąć się użytkownik.
- Aplikacje użytkownika - w ten sposób wreszcie dotarliśmy do programów, z których korzystamy na codzień.

JĄDRO

Większość systemów operacyjnych opiera się o koncepcję jądra, która w naturalny sposób wynika z modelu warstwowego. W takim wypadku jądro jest po prostu nazwą nadaną najniższej warstwie w pełni implementowanej jako program wykonywany przez CPU.

Do typowych funkcji realizowanych przez jądro można zaliczyć:

- zarządzanie sprzętem: przydziół czasu procesora, obsługę przerywań, obsługę układów wejścia/wyjścia;
- zarządzanie procesami: kolejkowanie, zapewnianie im wzajemnej komunikacji, synchronizację procesów, tworzenie i destrukcję procesów oraz ich usypanie i wznowianie (więcej o tym będzie w dalszej części podręcznika);
- zarządzanie pamięcią: Jądro jako jedyny program w systemie komputerowym ma nieograniczony dostęp do całej zainstalowanej pamięci. To ono przydziela fragmenty tej pamięci pracującym programom (procesom) oraz tworzy tzw. wirtualną przestrzeń adresową - czyli konwertuje fizyczne adresy pamięci RAM na ich odpowiednik unikalny dla każdego z procesów. W ten sposób możemy zapobiegać próbom odwołań się do tej samej komórki pamięci przez kilka procesów. Co więcej - dzięki wirtualnej przestrzeni adresowej można część danych z RAM (chwilowo nie wykorzystywanych) przenieść np. na HDD, dzięki czemu zwiększa się wirtualnie ilość pamięci operacyjnej dostępnej w systemie operacyjnym;
- zarządzanie urządzeniami: we wzorcowo zbudowanym systemie operacyjnym jedną metodą dostępu do urządzeń peryferyjnych jest wywołanie odpowiedniej funkcji jądra. Dzięki temu możliwe jest równoczesne korzystanie z tego samego urządzenia przez wiele aplikacji użytkownika - bez wchodzenia sobie nawzajem w drogę. Napisaliśmy o wzorcowym systemie - niestety, spełnienie tego postulatu w pełni nie jest możliwe. Popatrzmy przykładowo na karty graficzne: skoro dostęp do nich miałby być możliwy tylko i wyłącznie poprzez jądro, to jądro powinno znać wszystkie funkcje karty (żeby mogło je wywołać). A skoro jądro ma być uniwersalne, to powinno znać wszystkie funkcje wszystkich kart dostępnych teraz i w przyszłości (sic!) na rynku. Dlatego też, w uzasadnionych przypadkach, zezwala się na interakcję programu z urządzeniem peryferyjnym albo bezpośrednią, lub za pośrednictwem specjalnej, wydzielonej warstwy nie będącej częścią jądra (czyli np. DirectX dla systemu Windows);
- realizacja wywołań systemowych - czyli z jednej strony realizacja żądań aplikacji użytkownika, a z drugiej generowanie żądań do programów pochodzących od systemu (np. wykonywanie jakichś funkcji co określony czas czy przerysowanie zawartości okienka).

Po uruchomieniu komputera specjalny mały program, nazywany bootloaderem, ładuje jądro systemu operacyjnego do pamięci i przekazuje mu sterowanie maszyną. Jądro uruchamia w tym momencie pierwszy, główny proces, inicjalizuje wszystkie urządzenia peryferyjne i przechodzi w stan oczekiwania na zdarzenia (np. ruchy myszą, żądania wydruku od aplikacji, przerwania, itp.) W czasie, gdy nie pojawiają się żadne zdarzenia, jądro wykonuje specjalny proces, nazywany *idle* (w polskiej wersji Windows przetłumaczony jako *Proces bezczynności systemu*). W trybie jądra (kernel mode) każda operacja dozwolona przez architekturę procesora może być wykonana (dowolna instrukcja, dowolna operacja wejścia/wyjścia, zapis lub odczyt dowolnego miejsca w pamięci). W trybie użytkownika niektóre z tych operacji są zabronione, przy czym wykonania niedozwolonej operacji w tym trybie zapobiega sprzętowe ograniczenie wbudowane w procesor.

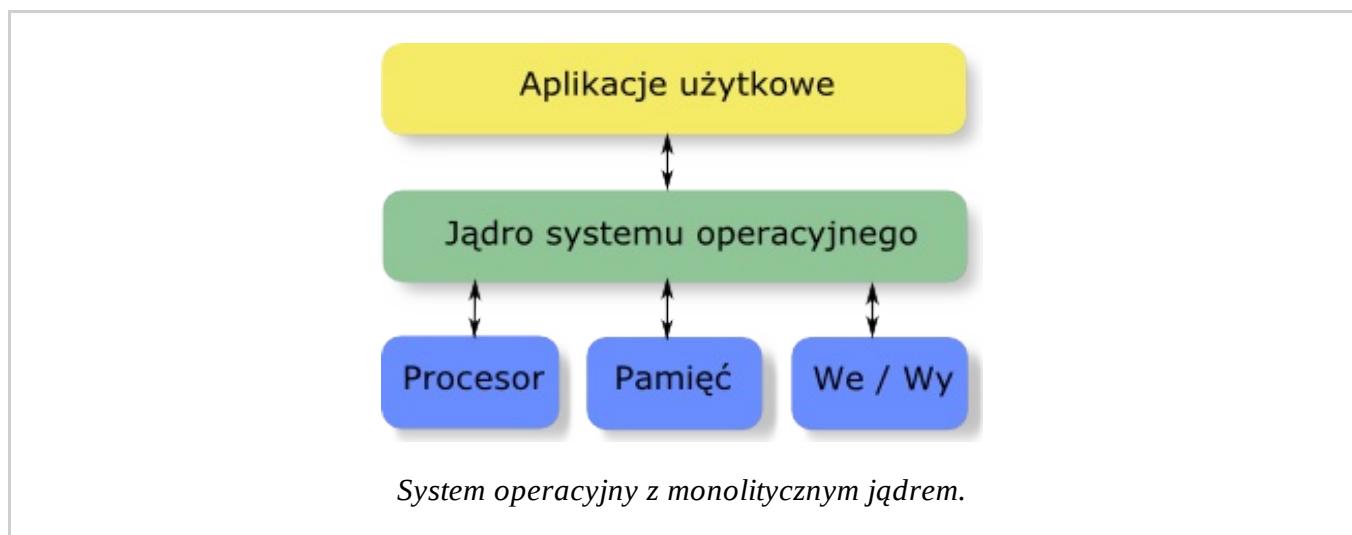
Każdy proces będący częścią jądra może pracować w jednym z dwóch trybów: jako proces użytkownika - kiedy dostęp do pamięci i urządzeń wejścia / wyjścia jest nadzorowany, oraz w tzw.

trybie jądra - kiedy wszystkie mechanizmy ochrony są wyłączone. Jak łatwo się domyśleć - błędnie napisany program czy funkcja uruchomiona jako proces jądra może w łatwy sposób "powiesić" cały system komputerowy.

Opracowanie jądra jest jednym z najtrudniejszych i najbardziej skomplikowanych zadań programistycznych. Jako główny, najważniejszy program w systemie komputerowym, jądro powinno spełniać różne (często przeciwnostne) wymagania. Musi cechować się wysoką wydajnością, stabilnością i odpornością na błędy aplikacji oraz awarie podzespołów. Dodatkowo, często nie może wykorzystywać mechanizmów abstrakcji, które samo udostępnia innym programom, czyli np. przy jego tworzeniu nie można zapomnieć o fizycznej realizacji zapisu na dysk i opierać się jedynie na koncepcji plików. Tak więc jądro samo w sobie stanowi całkiem skomplikowany system informatyczny, który może być zbudowany w różny sposób. Wyróżniamy kilka podstawowych metod konstrukcji jąder

JĄDRO MONOLITYCZNE

Często stosowane w systemach Unix-owych. Wszystkie zadania są wykonywane przez jądro, będące jednym, dużym programem pracującym w pełni (wszystkie funkcje) w trybie jądra. Zaletami tej techniki są: prostota, stabilność oraz łatwość komunikacji pomiędzy różnymi członami jądra (to przecież w tym wypadku jeden program!). Dodatkowo, systemy z jądrem monolitycznym zazwyczaj działają szybciej (szczególnie w komputerach o małej liczbie CPU) niż ich główna alternatywa - mikrojądro. Główna wada natomiast wynika z problemów w rozwijaniu takiego jądra (ze względu na jego wielkość) oraz dodawaniu do niego procedur obsługi nowych urządzeń peryferyjnych. Dodanie obsługi nowego urządzenia wymaga komplikacji całego jądra, co zajmuje czas i jest trudne do wykonania przez nawet zaawansowanego użytkownika.

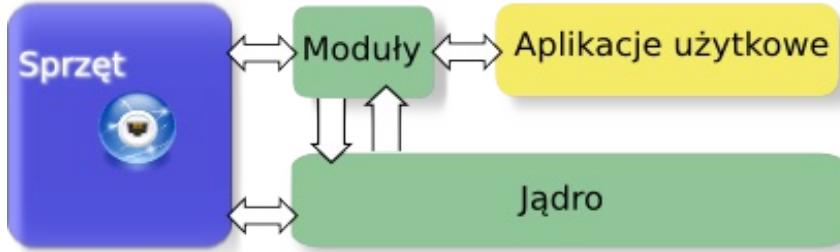


W czystej postaci jądro monolityczne występuje tylko w zastosowaniach niszowych, natomiast w dużej mierze takie rozwiązanie stosują systemy Linux, Solaris, OpenBSD, FreeBSD czy Mac OS-X - jądra tych systemów umożliwiają ładowanie modułów, co jest sprzeczne z koncepcją jednego dużego programu, ale daje możliwość stosowania driver'ów, czyli dodawania obsługi nowego sprzętu bez konieczności ponownej komplikacji całego jądra.

MIKROJĄDRO

W tej technice z monolitycznego jądra zostaje tylko jego podstawowa część (odpowiedzialna za zarządzanie procesami oraz pamięcią). Części odpowiedzialne za bardziej wyrafinowane funkcje (m.inn. obsługę urządzeń peryferyjnych) są wydzielone do funkcjonalnych bloków albo realizowane jako zwykłe procesy w trybie użytkownika. W większości przypadków takie bloki funkcjonalne mogą być ładowane i resetowane nie przerywając pracy systemu komputerowego, ponadto zwiększenie

funkcjonalności jądra (i w konsekwencji - całego systemu) jest możliwe bez wprowadzania jakichkolwiek zmian w jego podstawowej części. Główną wadą takiego podejścia jest wolniejsza praca systemu.



System operacyjny z mikrojądrem.

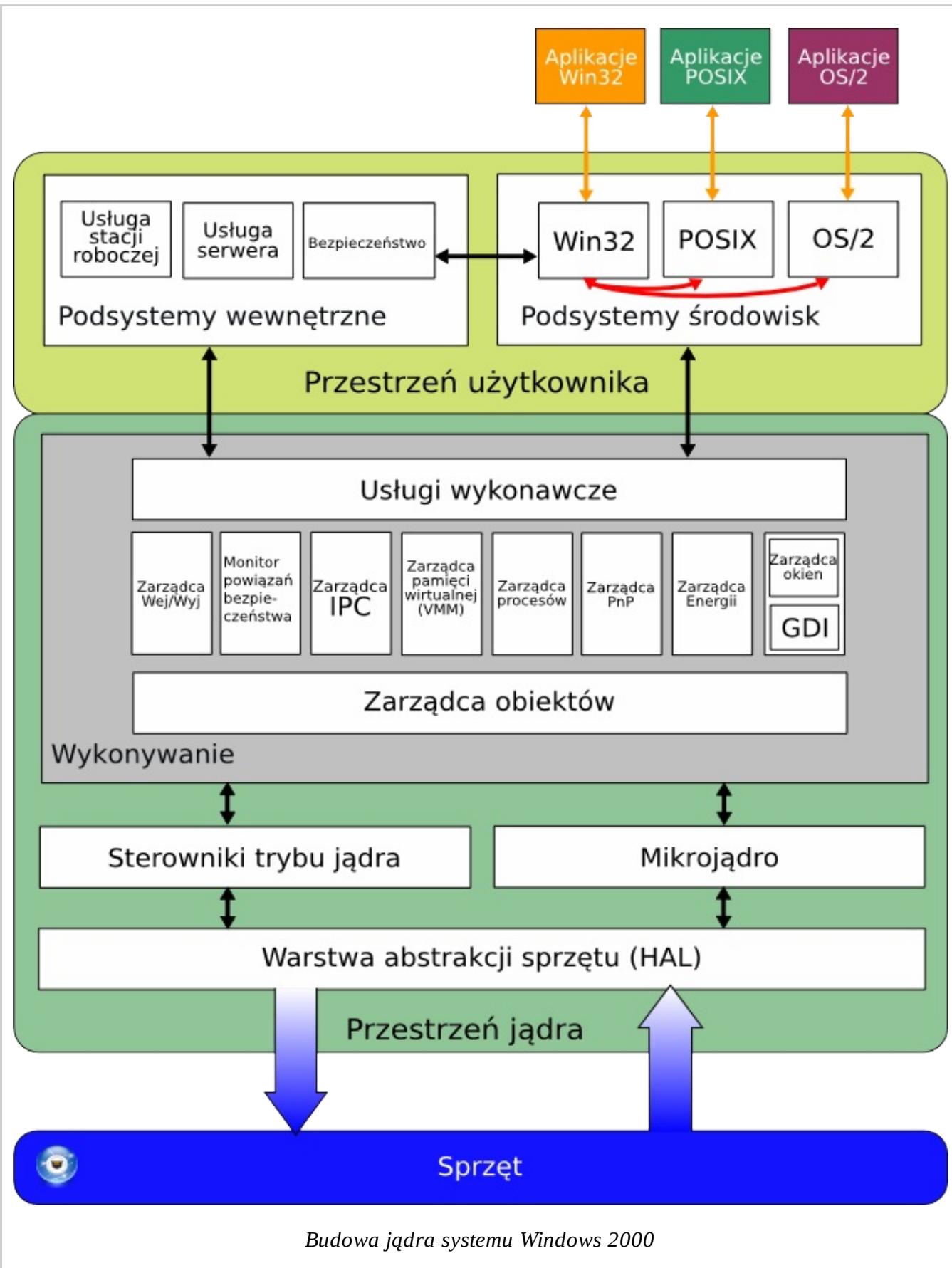
Dobrymi przykładami systemów operacyjnych opartych na mikrojądrze są AmigaOS, Amoeba, QNX, Minix, czy GNU Hurd, mikrojądrami są także jądra znane jako Mach i L4. Firma Microsoft pracuje nad własnym rozwiązaniem tego typu w projekcie Singularity.

JĄDRO HYBRYDOWE

Jest to kompromis między architekturą jądra monolitycznego i mikrojądra. W krytycznych usługach - np. stos sieci - usługi są na stałe wkompilowane w główny kod jądra, inne usługi pozostają oddzielone od głównego jądra i działają jako serwery (w przestrzeni jądra). Dzięki temu rozwiązaniu możliwe jest zachowanie wydajności jądra monolitycznego dla kluczowych usług.

Wydzielenie takiego rodzaju jądra budzi kontrowersje wśród specjalistów. Wielu z nich uważa, iż w porównaniu do podobnego jądra monolitycznego, pojęcie to jest niczym innym jak dobrym marketingiem.

Pomysłem stojącym za tym nowym pseudo-rodzajem jądra jest struktura podobna do mikrojądra, ale zaimplementowana jak jądro monolityczne. W przeciwieństwie do mikrojądra, wszystkie (lub prawie wszystkie) usługi wykonywane są w przestrzeni jądra. Podobnie jak w jądrze monolitycznym, nie ma strat w wydajności wywołanych przepływem komunikatów mikrojądra. Jednakże, podobnie jak w jądrach monolitycznych, nie ma korzyści wynikających z umieszczenia usług w przestrzeni użytkownika.



Tego typu jądro jest podstawą systemów Windows NT, 2000, XP, 2003, Vista. Jest ono wykorzystywane również przez mniej popularne systemy, jak np. BeOS.

INNE TYPY JĄDER

Oprócz omówionych, istnieją także rozwiązania które nie zdobyły większej popularności, jednakże dla pełnego przekazu wymienimy je tutaj:

- nanojądro - tylko bardzo mała część operacji wykonywana jest w trybie kernela. Czasem tylko warstwa dostępu do urządzeń fizycznych wykonywana jest w tym trybie w celu dostarczenia funkcjonalności systemów czasu rzeczywistego do normalnych systemów operacyjnych.
Przykładowa implementacja - Adeos;
- exojądro - minimalistyczny kernel umożliwiający uruchomienie tzw. maszyn wirtualnych, z których każda jest kopią fizycznego systemu z ograniczonymi zasobami. Rolą jądra jest odpowiedni przydział tych zasobów do poszczególnej wirtualnej maszyny i pośredniczenie wymianie informacji między nimi a fizycznymi urządzeniami. Przykładowe implementacje - Nemesis, ExOS.

API (

Usługi jądra nigdy nie są bezpośrednio widoczne dla użytkownika czy aplikacji. Użytkownik może je wykorzystywać i wywoływać korzystając z powłoki systemu, natomiast aplikacje najczęściej wykorzystują tzw. API - zbiór poleceń, które aplikacje użytkownika mogą wydawać systemowi operacyjnemu (np.: „zamknij plik”, „odczytaj znak z klawiatury”). Warstwa API (ang. *Application Program Interface*) pełni rolę pośrednika między systemem operacyjnym a programami użytkownika. API wykorzystywane jest jedynie przez programistów i zazwyczaj dostarczane w formie zestawu plików nagłówkowych języka programowania C. Kompletne API zaawansowanego systemu operacyjnego to olbrzymia biblioteka (kilkaset tysięcy funkcji), najczęściej w dodatku kompletnie różna dla różnych systemów operacyjnych.

PROGRAMY NARZĘDZIOWE

Najważniejszym z nich jest **powłoka systemu**, czyli jego interfejs widoczny dla użytkownika. To ona decyduje o wyglądzie i funkcjonalności systemu operacyjnego dla każdego z Was.

Dwa najczęściej spotykane typy powłoki to:

- tekstowy interpreter poleceń, np. BASH (*Bourne-Again Shell*) w Unixach czy CLI (*Command Line Interpreter*) w Windows. Tego typu powłokę najczęściej wykorzystują bardziej zaawansowani użytkownicy;
- graficzny interfejs użytkownika (GUI), czyli np. Explorer w Windows, czy też KDE czy Gnome dla Unixów. Graficzne interfejsy użytkownika to jest ten typ powłoki, z którym na codzień styka się zdecydowana większość użytkowników komputerów.

Powłoka często sama zawiera podstawowe polecenia, gdy jednak wydane przez użytkownika polecenie nie jest wbudowane, uruchamiany jest zewnętrzny program narzędziowy, np. do wyświetlenia zawartości aktualnego katalogu w powłoce tekstowej wywoływany jest program nazwany **dir** dla Windows oraz **ls** dla Unix-ów.

Z technicznego punktu widzenia powłoka nie różni się od innych programów narzędziowych, czy też nawet aplikacji zewnętrznych. Jest normalnym programem, pracującym w trybie użytkownika i może być zmieniona przez inny program definiowany przez użytkownika, co jest wykorzystywane do tworzenia rozmaitych nakładek, podmieniających np. interfejs Windows na podobny do MacOS.

KLASYFIKACJA SYSTEMÓW OPERACYJNYCH

Systemy operacyjne w czasie swego istnienia przeszły znaczące zmiany. Pierwsze z nich były bardzo proste i mało wydajne. Dzisiejsze systemy operacyjne są systemami o dużym stopniu skomplikowania, zajmują nierzaz wiele megabajtów pamięci, a ich wydajność jest niewyobrażalna w porównaniu z możliwościami ich odpowiedników sprzed lat.

SYSTEMY WSADOWE

Systemy wsadowe były jednymi z pierwszych systemów operacyjnych. Ich konstrukcja była bardzo prosta, a co za tym idzie również i ich możliwości były niewielkie.

Były one instalowane w pierwszych komputerach, w których urządzeniami wejściowymi były czytniki kart perforowanych. Użytkownik takiego systemu przygotowywał najpierw program i zapisywał go na kartach. Tak przygotowane dane trafiały do operatora komputera, który grupował podobne zadania w tak zwane wsady (ang. *batch*) i wprowadzał do komputera. Po przetworzeniu przez jednostkę centralną wyniki były wyprowadzane na drukarkę wierszową.

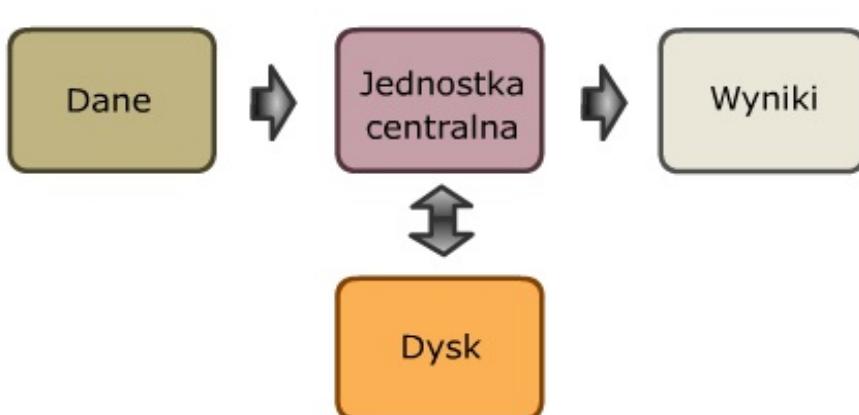
Zasadniczą cechą wsadowego systemu operacyjnego było:

- brak kontroli użytkownika nad wykonywanymi zadaniami;
- wykonywanie przez jednostkę centralną tylko jednego zadania w danym czasie;
- bardzo długie okresy bezczynności jednostki centralnej, a co za tym idzie niska wydajność.

Niskie wykorzystanie jednostki centralnej wynikało przede wszystkim z przestojów spowodowanych niską prędkością urządzeń wejścia wyjścia.

Aby podnieść wydajność takich systemów wprowadzono mechanizm, któremu nadano nazwę *spooling*.

Spooling wykorzystuje technologie dyskowe i polega na buforowaniu danych na dyskach przed i po ich wprowadzeniu do pamięci operacyjnej. Umożliwiło to znaczne zwiększenie wydajności systemu, rzadziej procesor musiał oczekивать na dane od użytkownika podawane poprzez czytnik kart dziurkowanych



Mechanizm spoolingu

Spooling umożliwiał jednocześnie przetwarzanie danych przez jednostkę centralną i pracę urządzeń wejścia-wyjścia. Mechanizm ten jest wykorzystywany do dziś między innymi przy przetwarzaniu danych w instalacjach zdalnych (np. przy komunikacji ze zdalną drukarką).

SYSTEMY WIELOPROGRAMOWE

Zastosowanie buforowania danych na dysku (lub aktualnie w pamięci operacyjnej) dało systemowi nowe możliwości. Nie musiał on już teraz wykonywać zadań w kolejności, w jakiej zostały one wprowadzone do systemu. Otrzymał prawo wyboru :) System operacyjny dysponując listą zadań, do której miał łatwy dostęp, rozpoczął pracę koncepcyjną polegającą na *planowaniu zadań*. Zadania były wybierane do realizacji w sposób maksymalizujący wydajność systemu. Odbywa się to mniej więcej tak...

Wybrane zadania są umieszczane w pamięci operacyjnej.

Jednostka centralna rozpoczyna pracę nad jednym zadaniem. Po pewnym czasie może okazać się, że wykonywane zadanie zażądało dostępu do dysku lub czytnika taśm. W systemie jednozadaniowym jednostka centralna musiała czekać na dostęp do dysku lub zamontowanie czytnika. W systemie wielozadaniowym system operacyjny może odłożyć takie zadanie i pobrać do wykonania następne, znajdujące się w pamięci operacyjnej. Tego typu zachowanie zwiększa oczywiście wydajność systemu, ale wymusza konieczność uporania się z wieloma trudnymi problemami:

- w jaki sposób wybierać zadania do wykonania?
- jak zapewnić ochronę danych umieszczonych w pamięci?
- w jaki sposób i kiedy współpracować z pamięcią dyskową?

i tym podobne.

Systemy wieloprogramowe nie rozwiązały wszystkich problemów. W dalszym ciągu użytkownik nie miał możliwości ingerencji w wykonywany program w trakcie jego trwania. Musiał więc przewidzieć możliwe zachowania programu przed jego uruchomieniem. Trudną sprawą było też testowanie programów, gdyż nie można było zmienić jego przebiegu obserwując jego zachowanie.

SYSTEMY Z PODZIAŁEM CZASU

Systemy te, zwane też systemami wielozadaniowymi (ang. *multitasking*), dzielą czas procesora między zadania umieszczone w pamięci. Jednostka centralna wykonuje więc jedno zadanie przez pewien czas, a następnie odkłada je i zabiera się za następne. Procesor przełącza się więc nieustannie między zadaniami. Te przełączenia występują na tyle szybko, że użytkownik może pracować z każdym z wykonywanych programów nie zauważając tych przełączeń.

Dzięki temu użytkownik systemu komputerowego ma możliwość interakcji z systemem. Może też wydawać polecenia programowi i oglądać w trakcie jego wykonywania wyniki działania programu.

Aby w sposób wydajny wykorzystywać mechanizm wielozadaniowości system operacyjny powinien charakteryzować się następującymi cechami:

- bezpośrednim dostępem do systemu plików;
- krótkim czasem odpowiedzi;
- mechanizmami decyzyjnymi;
- mechanizmami zarządzania pamięcią;
- mechanizmami ochrony zadań;
- mechanizmami współpracy z dyskiem;

- mechanizmami synchronizacji zadań;
- mechanizmami komunikacji między zadaniami.

SYSTEMY RÓWNOLEGŁE

Systemy równoległe są przystosowane do obsługi systemów komputerowych wyposażonych w więcej niż jeden procesor. Procesory te współpracują ze sobą wykorzystując wspólną szynę danych, zegar, często pamięć i urządzenia wejścia-wyjścia. Dzięki zastosowaniu systemów wyposażonych w kilka procesorów można zwiększyć wydajność całego systemu, zmniejszyć jego koszty poprzez wykorzystanie wspólnych urządzeń zewnętrznych, obudów, zasilaczy itp. oraz podnieść niezawodność dzięki dublowaniu zadań.

SYSTEMY JEDNO- I WIELO- DOSTĘPOWE

Systemy operacyjne można podzielić również biorąc pod uwagę liczbę użytkowników, których system potrafi obsłużyć w jednym czasie. Starsze systemy operacyjne oraz niektóre nowoczesne systemy typowe dla komputerów osobistych przeznaczone były i są dla jednego użytkownika. W systemach tych wszystkie zasoby komputera dostępne są jedynie dla jednej osoby (operatora), nie ma lub są bardzo ograniczone mechanizmy autoryzacji dostępu do danych oraz ochrony informacji. Do tego typu systemów zalicza się m.inn. Windows (wszystkie wersje poza serwerowymi).

Na drugim biegunie mamy do czynienia z systemami wielodostępowymi. W tym przypadku na jednym komputerze może jednocześnie pracować wiele osób, albo bezpośrednio korzystając z lokalnych urządzeń wejścia/wyjścia, lub zdalnie - z poziomu terminala czy też innego systemu komputerowego. Najlepszym przykładem takiego systemu operacyjnego jest współcześnie Linux, lub serwerowe wersje Windows z uruchomionymi usługami Terminal Services.

SYSTEMY ROZPROSZONE

Systemy rozproszone przeznaczone są do obsługi komputerów, w których wykonywane zadania są rozdzielone między kilka procesorów. W odróżnieniu jednak od systemów równoległych, procesory te nie dzielą wspólnej szyny danych ani zegara. Procesory komunikują się ze sobą za pomocą szybkiej specjalizowanej sieci lokalnej, lub szybkich łączy dalekiego zasięgu.

Dzięki zastosowaniu systemów rozproszonych można łączyć ze sobą komputery słabsze i lepsze, wykorzystywać drukarki zdalne, stosować rozproszone bazy danych itp. Kolejną zaletą systemów rozproszonych jest możliwość przyspieszenia obliczeń. Dzieje się tak, jeśli dużą liczbę obliczeń można podzielić na kilka współbieżnych procesów. Użytkownik może wykonywać nawet skomplikowane obliczenia siedząc przy mniej wydajnej maszynie, jeśli część zadań przejmie procesor z bardzo silnej i wydajnej jednostki zdalnej.

SYSTEMY KLASTROWE

System klastrowy jest pojęciem zawężającym pojęcie systemu rozprozonego. Kontroluje on pracę komputerów, które stanowią klastrowy, czyli zbiór powiązanych ze sobą komputerów, którego celem jest poprawienie wydajności, dostępności i odporności na awarię w stosunku do pojedynczego komputera. Najbardziej znaną klasą komputerów klastrowych jest klastrowy Beowulf będący zbiorem identycznych

komputerów PC połączonych siecią Ethernet. Systemem operacyjnym kontrolującym pracę Beowulfa jest najczęściej Linux (np. ClusterKnoppix), BSD (Dragonfly BSD) lub Solaris wyposażone w specjalne oprogramowanie wspomagające przekazywanie komunikatów między poszczególnymi komputerami klastra, takie jak Open Source Cluster Application Resources (OSCAR).

SYSTEMY CZASU RZECZYWISTEGO

Systemy operacyjne czasu rzeczywistego stanowią odrębną grupę systemów operacyjnych. Są to systemy wysoce specjalizowane, które muszą zapewnić określony czas reakcji systemu komputerowego na bodźce zewnętrzne. Tak więc widać, że podstawową cechą odróżniającą takie systemy od pozostałych rozwiązań jest czas. Jednakże nie chodzi tutaj głównie o jak najszybsze działanie systemu (jak się często mylnie sądzi), lecz o pewność, że w możliwym do określenia czasie system wykona zadanie mu powierzone (czyli determinizm czasowy). Istnieją dwa rodzaje systemów RT – soft i hard real time OS. Różnica między tymi systemami polega na tym, że w systemach hard zachowanie systemu (łącznie z czasami i kolejnością wywołań) jest w pełni zdeterminowane, to znaczy zawsze możliwe do określenia. W systemach soft RT wymagania co do determinizmu zdarzeń są nieco obniżone, choć zwykle można określić czas i kolejność wywołań.

Ze względu na rygorystyczne wymagania czasowe muszą one być lepiej dopasowane do sprzętu, na którym działają. W nich również stosowane są najbardziej wyrafinowane rozwiązania ochrony procesów.

Nietrywialnym problemem jest w tym przypadku algorytm szeregowania oraz podziału czasu. W systemie operacyjnym czasu rzeczywistego trzeba określić, któremu z procesów należy przydzielić procesor oraz na jak długi czas, aby wszystkie wykonywane procesy spełniały zdefiniowane dla nich ograniczenia czasowe, dlatego też stosuje się w nich specjalizowane, nie omawiane w tym podręczniku, algorytmy szeregowania procesów.

Systemy operacyjne czasu rzeczywistego znajdują zastosowanie głównie w komputerach wykorzystywanych do sterowania - od sterowania ABS w samochodach osobowych, przez przemysłowe układy automatyki, a kończąc na sterowaniu łazikiem marsjańskim czy promem kosmicznym. Inną grupą zastosowań jest sprzęt medyczny, wykorzystywany do nadzorowania życia ludzkiego.

Do przykładowych reprezentantów tego typu systemów należy [QNX](#), [VxWorks](#), [LynxOS](#) czy rozszerzenie systemu Linux - RT-Linux.

SYSTEMY OPERACYJNE TYPU WBUDOWANEGO (EMBEDDED)

Systemy typu embedded są przeznaczone do kontrolowania prostych urządzeń takich jak palmtopy, telefony, konsole nawigacji GPS itp. Częstość systemy te zaprojektowane są podobnie jak systemy czasu rzeczywistego z uwzględnieniem zwykle małej wydajności sprzętowej urządzeń dla których są przeznaczone. Funkcjonalność tych systemów zazwyczaj jest minimalna co pozwala na minimalizację zajmowanej przez nie pamięci operacyjnej i innych zasobów.

Kiedyś ten typ systemów był wyraźnie wydzielony ze względu na wspomnianą niską wydajność urządzeń na których pracował. W dniu dzisiejszym osiągi procesorów ARM montowanych we współczesnych urządzeniach kieszonkowych jest niewiele mniejsza niż klasycznych PC, w związku z tym najpopularniejsze systemy wbudowane (Android, iOS, Windows Phone) pod względem funkcjonalności zbliżają się do typowych systemów operacyjnych.

Przykładowymi systemami operacyjnymi typu wbudowanego są Embedded Linux, Minix3, VxWorks, Windows CE, PalmOS, Windows Mobile, Android, iPhone OS, DD-WRT (system operacyjny sterujący

pracą access pointów Linksys), Cisco IOS, BrickOS (LEGO Mindstorms), Robotic Operating System.

PODSUMOWANIE

System operacyjny kontroluje pracę poszczególnych komponentów systemu komputerowego. Umożliwia poprawną pracę podzespołów komputera oraz komunikację jednostki centralnej z innymi podzespołami. Troszczy się również o prawidłową pracę programów użytkowych i odpowiednie przydzielanie zasobów pamięci operacyjnej. Wreszcie kontroluje procesy zapisu danych na nośnikach pamięci masowej. System operacyjny pośredniczy między użytkownikiem i programami użytkowymi a sprzętem komputerowym. Dzięki temu twórcy programów użytkowych mogą łatwo wykorzystywać możliwości sprzętu nie zagłębiając się w techniczne aspekty ich budowy i funkcjonowania. Z drugiej zaś strony producenci sprzętu komputerowego mogą w prosty sposób dostosowywać swoje produkty do pracy z różnymi systemami poprzez dostarczanie wraz z nimi sterowników urządzeń.

Jak więc widziecie - jego rola w systemie komputerowym jest równie istotna jak sprzętu. Od jego wydajności i możliwości zależy w dużej mierze komfort pracy z systemem komputerowym oraz zakres prac, które można na nim wykonać.

Mamy nadzieję że teraz łatwiej Wam zrozumieć, dlaczego największą firmą informatyczną na świecie nie jest Intel, IBM, Dell czy im podobne, lecz ... Microsoft czy Google, które sprzętu nie produkują.

Systemy operacyjne przeszły wiele zmian w czasie swego istnienia, ich ewolucja jednak się nie zakończyła. Współcześnie można zauważać wydzielanie się dwóch wyraźnych nurtów w ich rozwoju:

- istnieje grupa systemów uniwersalnych - takich jak Windows, Mac OS czy Linux - w przypadku których dalszy rozwój jest skoncentrowany na zapewnieniu ... jeszcze większej uniwersalności. Wymienione systemy można stosować jako system operacyjny dla komputerów osobistych równie dobrze jak system stanowiący podstawę dużego serwera internetowego czy nawet (głównie Linux) jako podstawowy system operacyjny superkomputera. I to wszystko przy wykorzystaniu tego samego jądra, tej samej powłoki i dużej części wspólnych programów narzędziowych. Właściwości najnowszych wersji wspomnianych systemów przedstawimy w jednej z następnych lekcji;
- mimo ekspansji systemów uniwersalnych istnieje także duża grupa zastosowań, gdzie do tej pory się nie sprawdziły. Są to rozwiązania o specyficznych wymaganiach i / lub ograniczeniach sprzętowych. Tak więc powstały i dynamicznie rozwijają się specjalizowane systemy operacyjne, dostosowane do jednego i tylko jednego zadania. Najpopularniejszą grupą takich systemów (z którą większość z Was już się zetknęła) są systemy dla smartfonów i tabletów, takie jak Android, iOS, Windows Phone, czy wymarłe już Symbian OS czy Palm OS. Inna szeroka gama zastosowań to systemy czasu rzeczywistego, takie jak QNX czy VxWorks, szeroko wykorzystywane we wszelkiego rodzaju sterownikach. W tym podręczniku potraktowaliśmy je nieco po macoszemu, większość uwagi poświęcając systemom uniwersalnym - lecz nie można o nich zapominać;

Czy te dwa nurty utrzymają się - czas pokaże. Jedyne co możemy na pewno powiedzieć to to, że możemy spodziewać się dalszego rozwoju i zmian w systemach operacyjnych nowych generacji :).

PAMIĘĆ, PROCESY, WĄTKI

W tej części zajmiemy się nieco dokładniej tym, jak działa jądro systemu operacyjnego, jakie mechanizmy musi implementować oraz jakie algorytmy do tego wykorzystuje.

Zaczniemy od zarządzania pamięcią operacyjną. System jako taki powinien umożliwić jej ochronę (to przede wszystkim), dlatego pokażemy Wam jak to się robi. Sama ochrona to często za mało. Powiemy więc o mechanizmach umożliwiających tworzenie tzw. pamięci wirtualnej, o pamięci swap i mechanizmie nakładek.

W dalszej części zajmiemy się procesami. Przedstawimy definicję pojęć (zarówno procesu, jak i wątku) a następnie postaramy się pokazać, w jaki sposób możliwe jest działanie wielu programów na jednym systemie komputerowym w tym samym czasie. I to działania bezpiecznego - kiedy to programy mogą w tym samym czasie korzystać ze wspólnie dzielonych zasobów.

Nie będzie to prosta lekcja, będzie jedną z trudniejszych ... na pocieszenie powiemy, że potem to już będzie tylko łatwiej :)

PAMIĘĆ - ROLA, ADRESOWANIE

ROLA PAMIĘCI OPERACYJNEJ

Pamięć operacyjna jest jedną z podstawowych części systemu komputerowego. Do niej trafiają niemal wszystkie dane programów (a także i same programy - zostanie to wyjaśnione później), które będą przetwarzane przez procesor. To z tej właśnie pamięci, procesor doczytuje potrzebne informacje i to w tej pamięci znajduje się system operacyjny, który nadzoruje działanie całego komputera. Ale nie tylko. Do pamięci operacyjnej załadowanych jest wiele sterowników urządzeń I/O (a w systemie MS-DOS także część BIOS'u komputera).

Pamięć operacyjną należy traktować jako zbiór komórek przechowujących informacje. Z dostępem do komórek pamięci ściśle wiąże się pojęcie szyny danych, której szerokość (ilość bitów, które można przesyłać jednocześnie - popularnie 32-bity) determinuje w jak duże słowa grupowane są dane (np. w słowa 32-bitowe). Z tym zaś wiąże się pojęcie organizacji pamięci, czyli przydzielania poszczególnym słowom konkretnych i unikatowych adresów fizycznych.

Podczas wykonywania programu, jednostka centralna zbiera rozkazy z pamięci, zależnie od wartości licznika rozkazów. Typowy cykl wykonania rozkazu zaczyna się pobraniem rozkazu z pamięci, później następuje dekodowanie rozkazu i jeśli to konieczne z pamięci pobierane są argumenty. Po wykonaniu rozkazu wyniki jego działania mogą zostać zachowane również w pamięci. Rozkazy mogą być wykonywane tylko i wyłącznie w pamięci operacyjnej i z jej użyciem (!). Stąd nazwa "pamięć operacyjna" - pamięć, w której wykonywane są operacje.

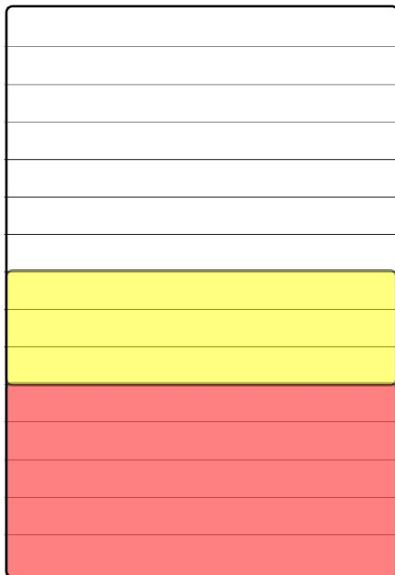
Pamięć operacyjna zwykle podzielona jest na dwie części. Jedną wykorzystuje system operacyjny, drugą programy użytkownika. W pierwszych pseudo-systemach operacyjnych, zdolnych do uruchomienia tylko jednego procesu cała pamięć operacyjna zamontowana w systemie była dostępna dla tego procesu. Z momentem pojawienia się systemów operacyjnych pojawiło się zagadnienie dzielenia pamięci na obszar systemu i programu. Istnieją różne rozwiązania (OS w pamięci ROM1, pamięć RAM przeznaczona dla programu, OS w pamięci RAM, na początku przestrzeni adresowej, część OS w RAM, sterowniki urządzeń w ROM). W obecnych systemach typu embedded lub niektórych palmtopach i telefonach używany jest model pierwszy, czyli przechowujący system

operacyjny w pamięci ROM, a dokładniej EPROM.

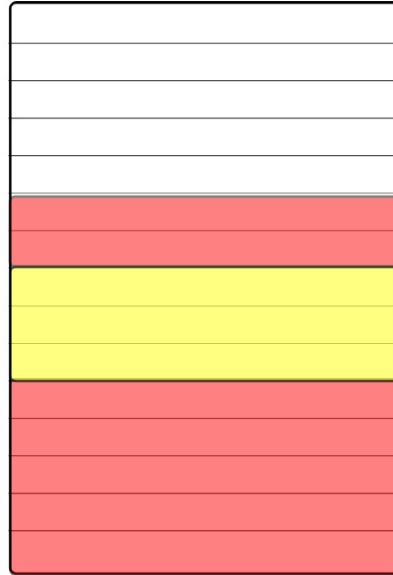
Jak wspomniano, na początku, w systemach jednoprocesowych nie istniał problem przydziału i chronienia fragmentów pamięci przed różnymi procesami. W momencie powstania pierwszych systemów wieloprogramowych, np. OS/360 zagadnienie to zaczęło być analizowane a pierwsze, dość proste rozwiązania optymalizowane. W systemach wieloprogramowych pamięć przeznaczona na programy użytkownika musi być dalej podzielona tak, aby mogła współpracować z wieloma procesami. Zadaniem tym zajmuje się właśnie system operacyjny i nosi ono nazwę **zarządzania pamięcią**.

ADRESOWANIE PAMIĘCI

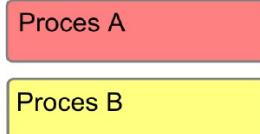
System operacyjny musi przydzielać pamięć aplikacji w postaci ciągłego segmentu. Niedopuszczalna jest sytuacja jak na poniższym rysunku, kiedy pamięć procesu A jest rozdzielona przez blok pamięci procesu B.



Alokacja pamięci przez procesy A i B



Próba zwiększenia przydziału pamięci procesowi A



Niedopuszczalny sposób alokacji pamięci

Niedopuszczalność takiej sytuacji wynika stąd, że dowolny program komputerowy w postaci skompilowanej i zlinkowanej (czyli języka maszynowego) jest umieszczany w jednej przestrzeni adresowej, która nie może być podzielona. Dodatkowo, linker adresuje pamięć w ten sposób jakby program miał dostęp do niej od adresu 0. W rzeczywistości, system operacyjny umieszcza uruchamiane programy w pewnym miejscu fizycznej pamięci, a adres startowy różni się od zakładanego przez linker. Z tego powodu SO musi modyfikować adresy pamięci z których korzysta uruchomiony proces. Może się to odbywać na dwa sposoby – poprzez modyfikację kodu maszynowego procesu (rozwiązanie obecnie niewykorzystywane) lub poprzez modyfikację adresów w momencie odwoływania się do nich przez proces (najczęściej poprzez dodawanie do oryginalnego adresu wartości specjalnego rejestru sprzętowego przechowującego informację o początku przestrzeni adresowej aktualnego procesu). Dodanie kolejnego rejestru, określającego koniec przestrzeni adresowej procesu pozwala na sprzętowe zabezpieczenie nadpisywania pamięci należącej do innych procesów przez niewłaściwie napisany program.

Dlatego też mamy do czynienia z dwoma typami adresów:

- **adres logiczny** definiowany jest jako położenie względem początku programu;
- **adres fizyczny** czyli położenie w pamięci głównej o unikatowym numerze.

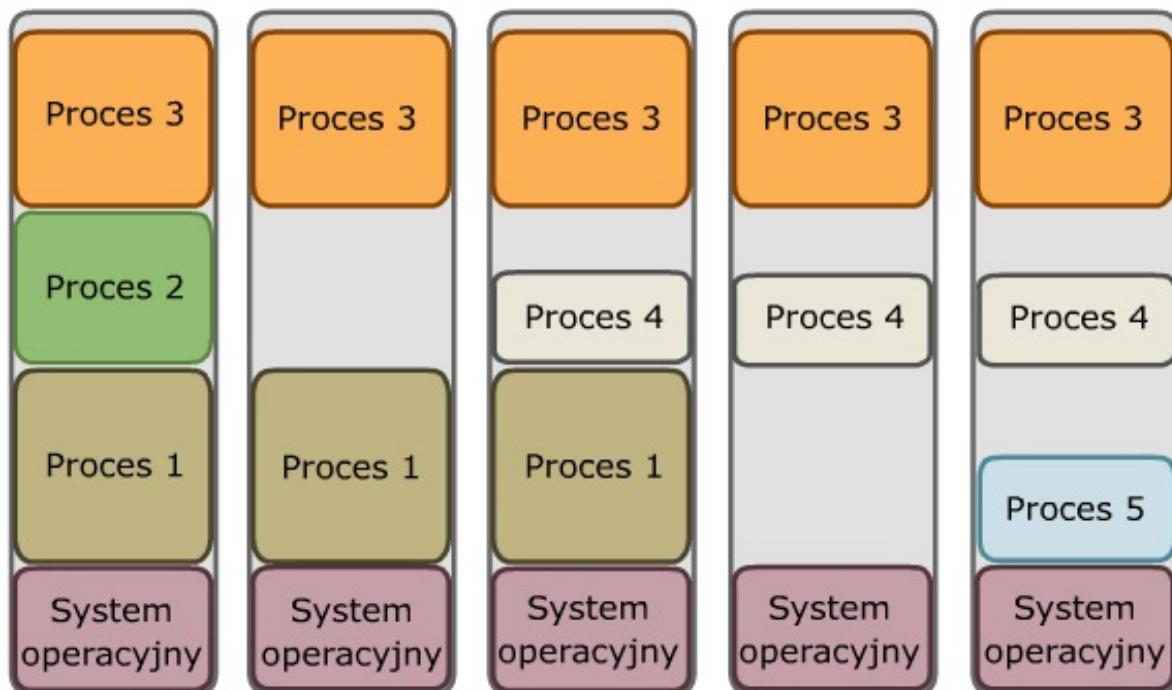
Realizacja przez procesor procesu wiąże się z konwersją adresów logicznych na fizyczne. Do każdego adresu logicznego w procesie dodawane jest aktualne położenie początku procesu - **adres bazowy**.

PARTYCJONOWANIE

Terminem **partycjonowanie** określa się podział pamięci w celu udostępnienia jej wielu procesom. Partycjonowaniu podlega ta część pamięci, której nie zajmuje system operacyjny. Wyróżniamy dwa rodzaje partycjonowania, którym odpowiadają dwa rodzaje partycji:

1. Partycje o **ustalonym rozmiarze** - wprowadzany do pamięci proces umieszczany jest w najmniejszej partycji, w której jest w stanie się pomieścić. Wyjaśnijmy, iż termin "ustalone" nie oznacza, że rozmiary partycji są równe, wręcz przeciwnie - nie są. Takie rozwiązanie sprawi jednak, że pojemność pamięci będzie niecałkowicie wykorzystana, ponieważ mało prawdopodobne (ale możliwe) jest zdarzenie, w którym rozmiar procesu będzie równy rozmiarowi przydzielonej mu partycji.
2. Partycje o **zmiennych rozmiarach** - proces wprowadzany do pamięci jest umieszczany w partycji o dokładnie takiej pojemności, jakiej ten proces wymaga. Metoda ta jest bardziej efektywna od opisanej powyżej.

Z partycjonowaniem wiążą się pewne problemy. Rozważmy na przykład sytuację przedstawioną na rysunku:



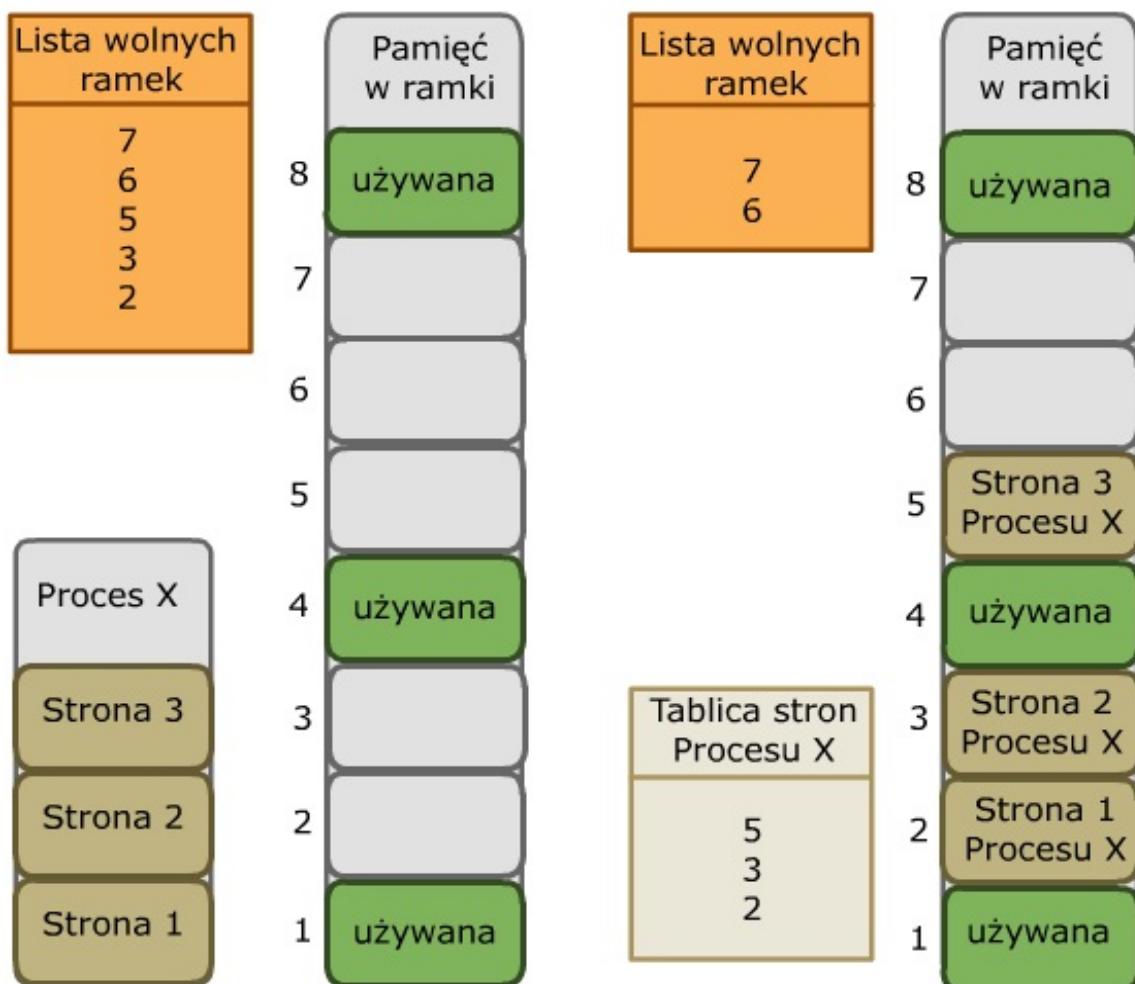
Hipotetyczny wynik partycjonowania

Na początku proces 2 zostaje usunięty z pamięci, na jego miejsce wchodzi proces 4, następnie wychodzi proces 1 i wchodzi proces 5. Na przykładzie tym doskonale widać, jak schemat partycjonowania doprowadził do powstania luk w pamięci. Zauważmy, że im dłużej trwa partycjonowanie, tym więcej powstaje takich luk i tym mniejsze procesy mogą zostać do pamięci wprowadzane. Takie postępowanie z pewnością doprowadziłoby do zapełnienia pamięci, bez jednoczesnego wykorzystania całej jej pojemności. Dlatego też stosuje się **upakowywanie** (ang. *compaction*), czyli okresowe przesuwanie przez system operacyjny procesów w pamięci, w celu skumulowania wolnej pamięci w jednym bloku.

STRONICOWANIE

Wydajniejszym sposobem przydziału pamięci jest **stronicowanie**. Stronicowanie opiera się na koncepcji podziału pamięci na małe fragmenty, zwane **ramkami**, przy równoczesnym podziale procesów na takie same fragmenty, zwane **stronami**. Przy takim schemacie tracona pojemność stanowi jedynie część ostatniej strony. Jeżeli przypomnimy sobie, że strony mają małe rozmiary, zorientujemy się, że tracona pojemność również jest niewielkich rozmiarów.

Zasadę działania stron i ramek ilustruje następny rysunek, który przedstawia pewien jeszcze mało znany proces X składający się z trzech stron. Gdy proces ów wczytywany jest do pamięci, system operacyjny dysponując listą wolnych ramek (która sam utrzymuje), potrafi w odpowiedni sposób załadować strony do wolnych, niekoniecznie sąsiadujących ramek.



Stronicowanie

Ale skąd proces ma wiedzieć, w których ramkach znajdują się jego strony? Odpowiedź jest bardzo prosta. System tworzy dla każdego procesu **tablicę stron**. Pokazuje on gdzie są ramki pasujące do stron procesu. W przypadku stronicowania, tłumaczenie przez procesor adresów logicznych (numer strony, adres względny) umożliwia właśnie tablica stron. Procesor uzyskując do niej dostęp może, znając adres logiczny, utworzyć adres fizyczny (numer ramki, adres względny).

Stronicowanie zapewnia bardzo efektywne wykorzystanie pamięci operacyjnej przy minimalnych stratach. Ułatwiony jest również dostęp do pamięci.

SEGMENTACJA

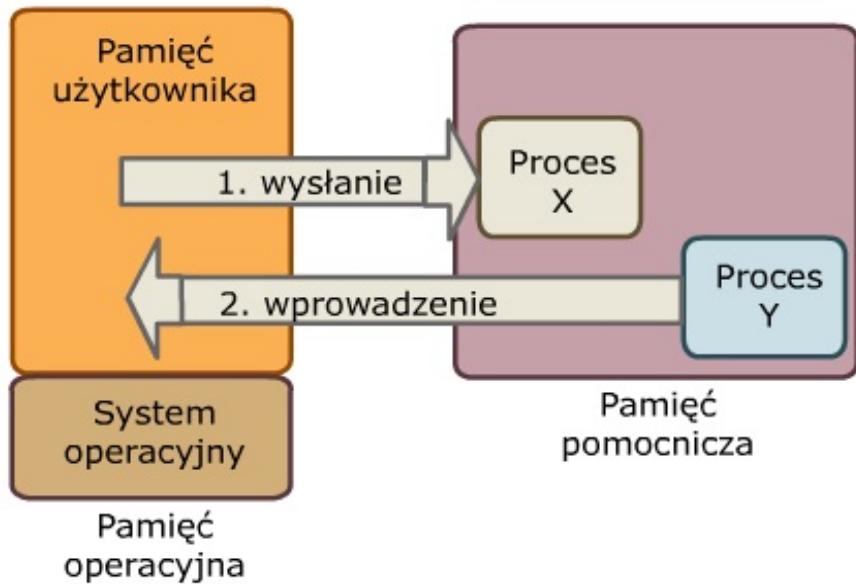
Segmentacja dzieli przestrzeń adresową na segmenty o zmiennym, dynamicznym rozmiarze, do których przypisane są programy i dane, zwykle przez system operacyjny. Możliwe jest istnienie wielu segmentów programów przeznaczonych dla różnych rodzajów programów, a także wiele segmentów danych. Ponadto każdy segment może mieć ustawione prawa dostępu i użytkownika, zaś odniesienia do tej pamięci stanowi adres, na który składa się numer segmentu i adres względny.

Segmentacja jest widzialna dla programisty i w przeciwnieństwie do stronicowania umożliwia wygodniejsze organizowanie programów i danych oraz może być wykorzystywana jako środek kojarzenia atrybutów przywileju i ochrony z rozkazami i danymi. Ponadto zapewnia min. ochronę zasobów i wykorzystanie zbioru danych przez kilka procesów. Mimo wielu zalet segmentacja nie zapewnia tak efektywnego zarządzania pamięcią. Dlatego też łączy się zalety obu schematów, stosując segmentację i stronicowanie.

SWAPPING

Współczesne systemy operacyjne (a także sprzęt komputerowy) pozwalają na implementację zaawansowanych metod zarządzania pamięcią. Do tej grupy zalicza się **wymiana** (ang. *swapping*). Wymieniany jest wykonywany proces znajdujący się w pamięci operacyjnej na proces znajdujący się w pamięci pomocniczej, zaś proces znajdujący się w pamięci pomocniczej wymieniany jest na proces z pamięci operacyjnej. Przy czym proces może być wysłany do pamięci pomocniczej, a następnie przywołyany z powrotem w celu kontynuowania działania.

Dla środowiska sterowanego rotacyjnym algorytmem planowania przydziału procesora cykl wymiany wygląda następująco:



Wymiana dwóch procesów pomiędzy pamięcią operacyjną a pomocniczą.

Po wyczerpaniu przez proces X przeznaczonego mu kwantu czasu procesora, a więc tymczasowego zakończenia działania, system operacyjny wymienia go na proces Y, któremu również przydzielany jest kwant czasu procesora. Po upływie tego czasu również ten proces zostanie wymieniony.

Przy procedurach wymiany istotną rolę odgrywa pamięć pomocnicza (ang. *backing store*). Musi być ona dostatecznie pojemna, aby pomieścić kopie obrazów pamięci wszystkich użytkowników oraz umożliwić bezpośredni dostęp do tych obrazów pamięci. Wszystkie procesy gotowe do działania, których obrazy pamięci znajdują się w pamięci pomocniczej lub operacyjnej, przechowywane są przez system operacyjny w **kolejce procesów gotowych** (ang. *ready queue*). Za każdym razem, gdy system operacyjny chce wykonać proces, wywołuje ekspedytora. **Ekspedytor** (ang. *dispatcher*) sprawdza dostępność procesu (nazwijmy go X) w pamięci operacyjnej. Jeżeli proces (X) nie znajduje się w pamięci operacyjnej oraz brak jest wolnego miejsca w tej samej pamięci, to *dispatcher* wysyła na dysk jeden z procesów znajdujących się w pamięci operacyjnej (np. Y). Na zwolnione w ten sposób miejsce wprowadzany jest potrzebny proces (X). Na koniec ekspedytor aktualnia rejestrze i przekazuje sterowanie do wprowadzonego procesu (X).

Wadą wymiany jest dość długi czas potrzebny na przełączenie zadania, w stosunku do krótkiego czasu na ich wykonanie. Z tego względu klasyczna technika wymiany jest stosowana coraz rzadziej przez systemy operacyjne - a przynajmniej na tyle rzadko na ile to możliwe.

PAMIĘĆ WIRTUALNA

Pamięć wirtualna jest to pamięć znajdująca się na części obszaru dysku. Jest ona antonimem dla pamięci głównej, czyli rzeczywistej (rzeczywista, gdyż proces tylko w tej pamięci może być wykonywany). Pamięć wirtualna wykorzystuje schemat **stronicowania na żądanie**, które tym różni się od omówionego wcześniej *stronicowania*, że strony procesu są wprowadzane do pamięci operacyjnej dopiero wówczas, gdy jest to konieczne tj. na żądanie.

Schemat stronicowania na żądanie umożliwia wczytywanie tylko tych stron procesu, których realizacja (a więc tym samym obecność w pamięci operacyjnej) jest konieczna. Pozostałe strony procesu mogą przebywać w pamięci pomocniczej. W przypadku, gdy program rozgałęzi się do rozkazu znajdującego się poza pamięcią lub nastąpi odwołanie do danych spoza obszaru pamięci, sygnalizowany jest **błąd strony**, co zmusza system do dostarczenia wymaganej strony.

Ograniczenie ilości stron procesu znajdujących się w pamięci powoduje, że możliwa staje się współpraca z większą ich liczbą. Ponadto nieużywane strony nie zajmują niepotrzebnie czasu procesora. System operacyjny musi jednak w odpowiedni sposób pilnować, aby wprowadzając do pamięci jedną stronę, wyrzucić jednocześnie inną. W przeciwnym wypadku może dojść do **szamotania** (ang. *trashing*), polegającego na wykorzystaniu większości czasu procesora na wymianę stron, a nie na przetwarzaniu polecen.

Technika stronicowania na żądanie pozwala na stworzenie procesu o rozmiarach większych niż pojemność pamięci głównej oraz dała programistom ogromne zaplecze - pamięć wirtualną.

PROCESY

DEFINICJA PROCESU

Do tej pory już często przewijało się pojęcie procesu. Często używaliśmy go zamiennie z bardziej intuicyjnym określeniem program ... nie zawsze jest to podejście prawidłowe. Oba pojęcia występują w informatyce i co więcej - nie są sobie równoważne. W systemach nie umożliwiających uruchomienia więcej niż jednego programu naraz nie istniał problem rozdzielenia dostępu do poszczególnych urządzeń składowych komputera różnym programom, nie było też potrzeby zatrzymywania i uruchamiania kolejnych procesów. Wprowadzenie możliwości uruchomienia i wykonywania więcej niż jednego programu jednocześnie spowodowało konieczność rozwiązymania przedstawionych wyżej zagadnień.

Przez proces rozumiemy program będący w trakcie wykonywania. Wykonanie procesu musi przebiegać w sposób sekwencyjny - tzn. w danej chwili na żądanie procesu może być wykonany tylko jeden rozkaz kodu programu.

Na pojęcie procesu, oprócz kodu programu, składają się również: bieżąca czynność reprezentowana przez **licznik rozkazów** (ang. *program counter*) oraz zawartość rejestrów procesora, zwykle także **stos procesu** (ang. *process stack*), **sekcja danych** (ang. *data section*) zawierająca zmienne globalne, oraz odwołania do zasobów przydzielonych przez system operacyjny np. uchwytów (handle) w Microsoft Windows. Często spotkacie się także z określeniem **kontekst wywołania** (*execution context*). Kontekst ten to zestaw danych, dzięki którym SO jest w stanie nadzorować wykonywanie programu. Do kontekstu należy zawartość rejestrów procesora (licznika programu i rejestrów danych), priorytet danego procesu oraz informacja o tym czy proces oczekuje na wystąpienie określonego zdarzenia (event). Każdy proces charakteryzuje się jednocześnie stanem procesu. Nie mylcie jednak pojęcia procesu z wątkiem opisany dalej - podstawowa różnica polega na tym, iż proces ma przydzieloną własną pamięć na dane, a wątek takiej nie ma.

Ważne jest rozgraniczenie pojęcia programu i procesu. Pamiętajmy, że program jest obiektem pasywnym, a wszelka aktywność należy do procesów. Po uruchomieniu każdemu procesowi przydzielone zostają zasoby, czyli:

- czas procesora (i sam procesor w systemach wieloprocesorowych);
- pamięć (czyli stos procesu i sekcja danych);
- dostęp do urządzeń wejścia-wyjścia;
- pliki.

STAN PROCESU

Proces, który się wykonuje, zmienia również swój stan (ang. *state*). Stan ten jest częściowo określany przez to, co robi proces w chwili bieżącej. Możliwe są następujące stany procesów:

- **nowy** - pierwszy stan procesu;
- **aktywny** - proces wykonuje instrukcje;
- **oczekiwanie** - proces czeka na wystąpienie jakiegoś zdarzenia (np. zakończenie operacji wejścia-wyjścia);
- **gotowy** - proces czeka na przydział procesora;
- **zakończony** - proces zakończył działanie.

W różnych systemach operacyjnych stany noszą różne nazwy. W niektórych omówione stany są

bardziej rozdrobnione, jednak przedstawione powyżej stany występują zawsze. Pamiętajmy, że w każdej chwili, w określonym procesorze, tylko jeden proces może być aktywny, zaś wiele procesów może być gotowych lub oczekujących.

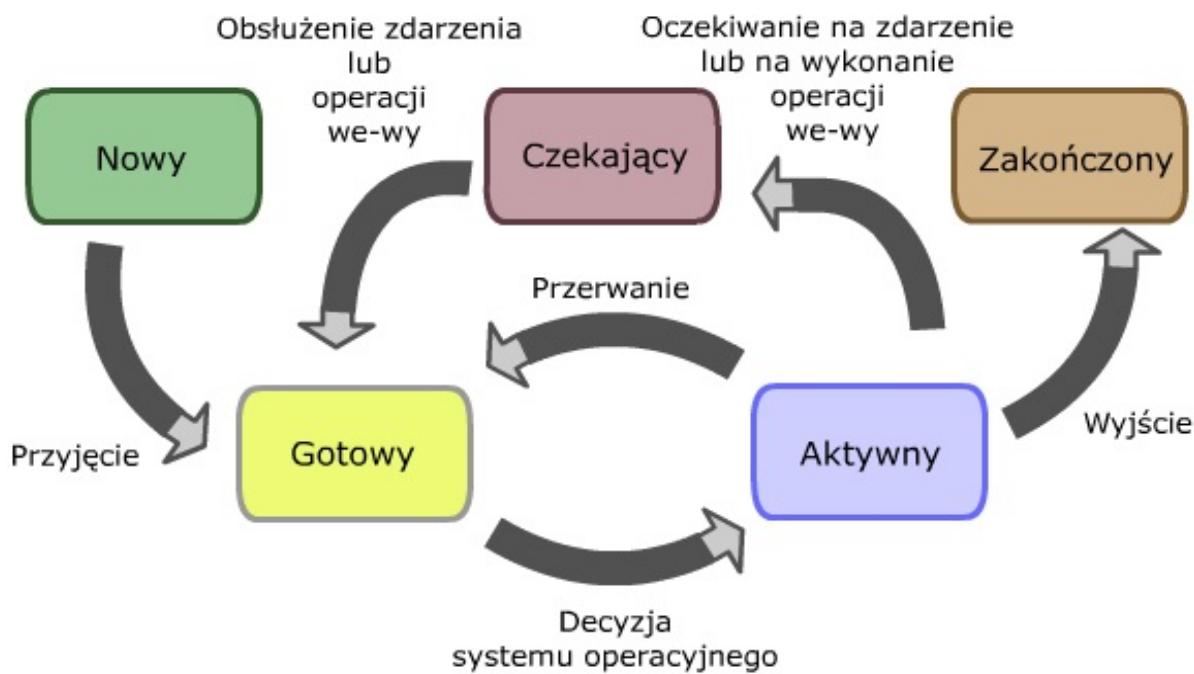


Diagram stanów procesu.

Powyższy rysunek przedstawia wszelkie możliwe przejścia pomiędzy poszczególnymi stanami procesu.

BLOK KONTROLNY PROCESU

System operacyjny przechowuje informacje o uruchomionych procesach w specjalnym miejscu pamięci operacyjnej zwany **tablicą procesów** (process table). Każdy proces reprezentowany jest tam przez **blok kontrolny procesu** (ang. *process control block* - PCB). Blok kontrolny procesu zawiera następujące pola:

- **stan procesu** - jeden ze stanów procesu;
- **licznik rozkazów** - adres następnego rozkazu do wykonania w procesie;
- **rejestry procesora** - lista i typ rejestrów zależą od procesora;



Blok kontrolny procesu

- **informacje o planowaniu przydziału procesora** - należy do nich priorytet procesu, wskaźnik do kolejek porządkujących zamówienia i inne parametry planowania;
- **informacje o zarządzaniu pamięcią** - zawartości rejestrów granicznych, tablice stron i segmentów;
- **informacje do rozliczeń** - należą do nich: ilość zużytego czasu procesora i czasu rzeczywistego, ograniczenia czasowe, numery kont, numery zadań lub procesów.
- **Informacje o stanie wejścia-wyjścia** - występują tu informacje o urządzeniach I/O przydzielonych do procesu, wykaz otwartych plików, itp.

Dzięki tym informacjom SO może w dowolnym momencie zatrzymywać i uruchamiać procesy, chronić dane obszary pamięci co pozwala na uruchamianie wielu procesów równolegle. Oczywiście procesy te nie są wykonywane jednocześnie (uogólniając – w systemie może jednocześnie działać tyle procesów ile procesorów jest w nim fizycznie obecnych). W systemie, w którym uruchomione jest więcej procesów niż procesorów jądro systemu operacyjnego wykonuje je na przemian, uruchamiając każdy z procesów na krótki okres czasu.

TWORZENIE PROCESÓW

Proces może utworzyć kolejne procesy. Mówimy, że proces "stwórca" - to **proces macierzysty** (ang. *parent process*), zaś proces przez niego utworzony - to **proces potomny** lub **potomek** (ang. *child*).

Dopuszczalne są dwie możliwości przydziału przestrzeni adresowej dla procesu potomnego:

- proces potomny jest kopią swego rodzica;
- proces potomny wiąże się z wykonaniem nowego programu.

Ponieważ proces do wykonania powierzonych mu zadań potrzebuje zasobów systemowych (np. czasu procesora), to muszą one zostać przydzielone. Proces potomny utworzony przez proces macierzysty może dostać zasoby systemowe od systemu operacyjnego lub też od procesu macierzystego - przy czym ta druga droga jest rzadko wykorzystywana. Po utworzeniu procesu potomnego, proces macierzysty może:

- kontynuować działanie współbieżnie ze swoimi potomkami;
- zaczekać na zakończenie działań swoich procesów potomnych (części z nich albo wszystkich).

Drugi z wyżej wymienionych przypadków pozwala na zastosowanie w środowiskach wieloprocesorowych mechanizmów synchronizacji (np. barier).

KOŃCZENIE PROCESU

Koniec działania procesu następuje, gdy proces wykona swoją ostatnią instrukcję. Wówczas proces wywołuje funkcję systemową *exit* i zostaje usunięty przez system operacyjny. Przed swym wyjściem może jednak przekazać dane do procesu macierzystego. Przed wyjściem odbierane są mu również wszystkie zasoby procesu przez system operacyjny.

Zakończenie procesu może przebiegać także inaczej. Proces macierzysty może zakończyć działanie procesu potomnego używając funkcji *abort*. Do tego proces macierzysty musi znać identyfikatory swych potomków - więc tworzenie nowego procesu zawsze wiąże się z przekazaniem jego identyfikatora do procesu macierzystego. Funkcja *abort* może zostać użyta w przypadku, gdy:

- wykonanie zadania przez potomka jest już zbędne;
- potomek nadużył przydzielonych mu zasobów;
- po zakończeniu procesu macierzystego system operacyjny nie pozwala jego potomkowi na dalszą pracę.

W przypadku gdy proces macierzysty nie obsługuje odpowiednio zakończenia procesu potomnego (w uproszczeniu - nie odbierze informacji wygenerowanej przez system o zakończeniu procesu potomnego) - w tablicy procesów pozostaje wpis opisujący program, którego wykonanie w systemie operacyjnym zostało zakończone. Taki proces nazywa się **procesem zombie**. Termin ten odnosi się zazwyczaj do systemów z rodziny Unix, gdzie pozostawienie wpisu zombie tymczasowo zajmującego pozycję w tablicy procesów zapobiega ponownemu wykorzystaniu identyfikatora procesu i możliwym na skutek tego pomyłkom programistycznym.

Zakończenie działania procesu nie oznacza wcale, że proces ten nie może zostać uruchomiony ponownie z takimi samymi zasobami, jakie miał wcześniej. W przypadku procesów potomnych, dla których proces macierzysty

WSPÓŁPRACA POMIĘDZY PROCESAMI

We współczesnym systemie operacyjnym współbieżnym procesy, które są w trakcie wykonania, mogą współpracować ze sobą lub działać niezależnie.

Proces niezależny (ang. independent) to taki, który nie może oddziaływać na inne procesy wykonywane w systemie, a i inne procesy nie mogą oddziaływać na niego. Procesy są więc niezależne tylko i wyłącznie wtedy, gdy nie dzielą żadnych danych z żadnymi procesami.

Proces współpracujący (ang. *cooperating*) to taki, który może wpływać na inne procesy lub inne procesy mogą oddziaływać na niego. Analogicznie, proces współpracujący to proces dzielący dane z innymi procesami.

Dzięki wprowadzeniu środowiska umożliwiającego współpracę pomiędzy procesami możliwe staje się:

- dzielenie informacji - możliwość dostarczania kilku użytkownikom tych samych informacji (np. kopii arkusza kalkulacyjnego);
- przyspieszenie obliczeń - duże zadania mogą być dekomponowane na podzadania, które z kolei mogą być wykonywane równolegle (o ile dysponujemy kilkoma procesorami);
- modularność - możliwość stworzenia systemu składającego się z osobnych procesów;
- wygoda - możliwość korzystania z kilku funkcji systemu równocześnie (np. redagowanie tekstu i drukowanie).

Zagadnienia współpracy pomiędzy procesami, oraz pomiędzy omówionymi dalej wątkami są jedną z największych zmør programistów - często doprowadzając do sytuacji, w której program (programy) nie działają i nikt nie wie dlaczego ... ;) Trochę dalej w tym podręczniku postaramy się powiedzieć Wam dlaczego ...

PODZIAŁ CZASU PROCESORA

W systemie, w którym uruchomione jest więcej procesów niż procesorów jądro systemu operacyjnego wykonuje je na przemian, uruchamiając każdy z procesów na krótki okres czasu. Czasy przełączania oraz wykonywania poszczególnych fragmentów procesów są na tyle krótkie, że dla użytkownika systemu wykonywanie procesów wydaje się być jednokrotnego. Opisany mechanizm przemiennej wykonywania programów wykonywany jest przez systemy operacyjne z tzw. **wywłaszczającą wielozadaniowością** (ang. *preemptive multitasking*). W tym trybie działają wszystkie współczesne systemy obsługujące komputery (Microsoft Windows, Linux, Mac OS X), oraz współczesne systemy klasy embedded (Android, iOS).

Pierwsze wersje Windows i Mac OS (do wersji 9), a także specjalizowane systemy operacyjne przeznaczone głównie dla prostych komputerów pracują w trybie **wielozadaniowości współpracującej** (ang. *cooperative multitasking*). Różnica w działaniu polega na tym, że w przypadku uruchomienia wielu procesów, pierwszy z nich jest wykonywany ciągle do momentu przejścia przez niego w tryb oczekiwania na zdarzenie lub komunikat od innego procesu. W tym momencie uruchamiany jest kolejny proces, który jest znowu wykonywany do momentu przejścia w stan oczekiwania. W takim przypadku nie występuje konieczność zarządzania czasem procesora - mamy pełną demokrację, i jak proces A dostanie czas procesora (dorwie się do mikrofonu) to tylko od niego zależy kiedy skończy swoje działanie (przestanie mówić). Wyobraźcie sobie co działyby się, gdyby takie zasady zastosować w polskim Sejmie .. ;) W praktyce to rozwiązanie nie sprawdziło się najlepiej, i współcześnie nie jest stosowane.

Wracając do systemów z wywłaszczającą wielozadaniowością - by to rozwiązanie działało optymalnie, należy zapewnić algorytmy **planowania przydziału czasu** procesora, czyli określenia kolejności przydziału jego mocy obliczeniowej procesom z *kolejki procesów gotowych do działania*. Jeszcze prościej - określenie który z procesów (i ich wątków) ma być właśnie uruchomiony i na jak długo.

Część systemu operacyjnego odpowiedzialna za tą funkcjonalność nazywana jest **schedulerem** (polski odpowiednik - planista - raczej się nie przyjął). W zależności od przeznaczenia systemu operacyjnego (np. dla komputera osobistego, smartfona, serwera) algorytmy sterujące wykonaniem procesu mogą być różne, bardziej lub mniej skomplikowane. Generalnie scheduler ma kilka zadań:

- wybranie odpowiedniego procesu do uruchomienia (w większości systemów operacyjnych procesy mogą mieć różne priorytety, oznaczające ich istotność w systemie i warunkujące w pewnym stopniu kolejność ich uruchamiania);
- zapewnienie jak najlepszego wykorzystania czasu procesora;
- minimalizację czasu jaki procesy oczekują na uruchomienie;
- minimalizację liczby procesów oczekujących (czyli możliwe szybkie zakończenie procesów uruchomionych);
- przydział takiego samego czasu procesora każdemu procesowi (z uwzględnieniem jego priorytetu);
- w szczególnych przypadkach (systemy czasu rzeczywistego) zapewnienie wykonania procesu przed upływnięciem wyznaczonego czasu;
- maksymalizację wykorzystania urządzeń peryferyjnych

Dodatkowo, ze względu na mnogość operacji, które są wykonywane przy przełączaniu procesów (zmiana trybu pracy do trybu jądra, zapisanie bieżącego stanu procesu, zachowanie mapy pamięci przypisanej procesowi, wczytanie tych danych dla nowo uruchamianego procesu, itp... , jest to operacja kosztowna czasowo i z tego względu powinna być wykonywana najrzadziej jak to możliwe co z kolei stoi często w konflikcie z wymienionymi powyżej zadaniami schedulera.

Poniżej zostaną omówione najciekawsze z algorytmów planowania przydziału procesora.

ALGORYTM FCFS

FCFS (ang. *first-come, first-served*) oznacza "pierwszy zgłoszony - pierwszy obsłużony". Algorytm ten opiera się na prostej zasadzie, którą właściwie wyjaśnia jego nazwa. Najlepiej do implementacji algorytmu nadaje się kolejka FIFO, w której blok kontrolny procesu wchodzącego do kolejki dołączany jest na jej koniec, zaś wolny procesor przydzielany jest procesowi z początku kolejki.

Wadą wspomnianej metody przydziału procesora jest to, że średni czas oczekiwania może być bardzo długi. Jeżeli rozważymy przyjście w danym momencie czasu trzech procesów, których długości faz procesora wynoszą:

Proces	Czas trwania fazy
P ₁	21 ms
P ₂	6 ms
P ₃	3 ms

przekonamy się, żeadejscie procesów w kolejności P₁,P₂,P₃ wyglądać będzie na *diagramie Gantta* następująco dla algorytmu FCFS:



Dla procesu P₁ czas oczekiwania wynosi więc 0 ms, dla procesu P₂ 21 ms, dla procesu P₃ zaś 27 ms. Średni czas oczekiwania wynosi więc: $(0+21+27)/3 = 16$ ms. Dla sytuacji, w której procesy nadeszłyby w kolejności P₃, P₂, P₁ diagram Gantta wyglądałby następująco:



Dla takiego przypadku średni czas oczekiwania wyniósłby: $(0+3+9)/3 = 4$ ms. Widzimy więc, że zastosowanie algorytmu niesie ze sobą pewne ryzyko.

W warunkach dynamicznych działanie algorytmu również nie jest zadowalające. W przypadku, gdy mamy do czynienia z dużym procesem ograniczonym przez procesor oraz małymi (szyciejsią wykonyującymi) procesami ograniczonymi przez wejście-wyjście, wtedy proces ograniczony przez procesor jako pierwszy uzyska dostęp do procesora, procesy wejścia-wyjścia oczekującą będą w kolejce procesów gotowych stosunkowo długo, a urządzenia wejścia wyjścia w tym czasie nie będą mogły być używane. Po zakończeniu wykonania procesu ograniczonego przez procesor, proces ten przejdzie do kolejki oczekującej na wejście-wyjście, natomiast małe procesy ograniczone przez wejście-wyjście wykonają się w krótkim czasie i zwolnią procesor, który może wtedy być bezczynny. Następnie powróci nasz duży proces, który ponownie będzie obsługiwany przez procesor, a procesy mniejszych rozmiarów będą musiały czekać. Sytuację wyżej opisaną, w której procesy oczekują na zwolnienie procesora przez jeden duży proces, nazywamy **efektem konwoju** (ang. *convoy effect*).

Algorytm ten, z uwagi na to, że pozwala procesowi na zajmowanie procesora przez dłuższy czas, nie nadaje się do stosowania w środowiskach z podziałem czasu, w których każdy użytkownik powinien dostawać przydział procesora w równych odstępach czasu.

Taki właśnie algorytm był stosowany w Windows od 1.0 do 3.11, stanowił także istotną część systemu Windows serii 9x (95-ME). Z powodu wad tego algorytmu, wspomniane systemy nigdy nie były przez informatyków traktowane jako poważna alternatywa dla serwerów czy wymagających stacji roboczych.

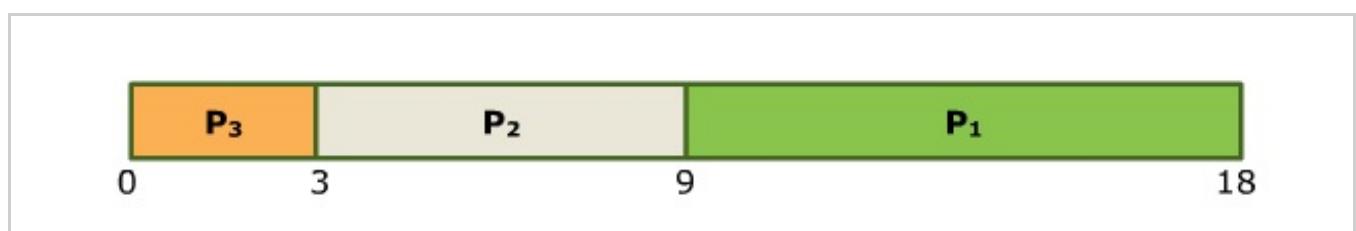
ALGORYTM SJF

Algorytm SJF (ang. *shortest-job-first*), czyli "najkrótsze zadanie najpierw", opiera swe działanie na powiązaniu procesu z długością jego najbliższej fazy. Procesor zostaje przydzielony procesowi, który ma najkrótszą następną fazę procesora. W przypadku, gdy dwa procesy mają następujące fazy procesora równe, stosuje się algorytm FCFS.

Porównajmy działanie tego algorytmu do algorytmu FCFS. Założymy więc, że mamy 3 następujące procesy:

Proces	Czas trwania fazy
P ₁	6 ms
P ₂	9 ms
P ₃	3 ms

Diagram Gantta dla tych procesów wygląda następująco:



Jeżeli procesy przychodzą w kolejności P₁, P₂, P₃ to dla algorytmu SJF średni czas oczekiwania wynosi $(3+9+0)/3 = 4\text{ms}$, zaś dla algorytmu FCFS wyniósłby $(0+6+15)/3 = 7 \text{ ms}$.

Algorytm SJF daje minimalny średni czas dla danego zbioru procesów. Mówimy, że algorytm SJF jest **czasowo-optimalny**.

Dużym problemem w algorytmie SJF jest określenie długości kolejnego zamówienia na przydział procesora. Przy planowaniu długoterminowym zadań, za czynnik ten możemy przyjąć limit czasu procesu określony przez przedkładającego zadanie użytkownika. Algorytm SJF jest więc chętnie używany w planowaniu długoterminowym. Natomiast w przypadku planowania krótkoterminowego, nie ma sposobu na określenie długości kolejnej fazy procesora. Dlatego też oszacowuje się tę wartość, obliczając ją na ogół jako średnią wykładniczą pomiarów długości poprzednich faz procesora.

Możliwe są dwa sposoby realizacji algorytmu SJF - wywłaszczająca (ang. *shortest-remaining-time-first*) oraz niewywłaszczająca. Różnica pomiędzy tymi realizacjami widoczna jest w sytuacji, gdy w kolejce procesów gotowych pojawia się nowy proces o krótszej kolejnej fazie procesora niż pozostała faza "resztki" procesu aktualnie wykonywanego. W takim układzie algorytm wywłaszczający usuwa

aktualny proces z procesora, zaś algorytm niewyłączający pozwala procesowi bieżącemu na zakończenie fazy procesora. Forma wyłączająca algorytmu SJF jest więc szybsza.

ALGORYTM PLANOWANIA ROTACYJNEGO - RR

Algorytm planowania rotacyjnego (ang. *round-robin* - RR) jest oparty na algorytmie FCFS, jednakże różni się od FCFS faktem, że możliwe jest wyłączanie. Algorytm ten jest chętnie stosowany w systemach z podziałem czasu.

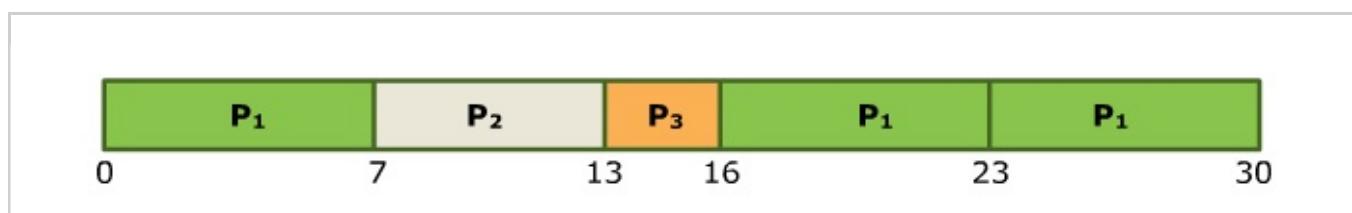
Zasada działania algorytmu jest następująca. Z każdym procesem powiązany jest kwant czasu, najczęściej 10 do 100 ms. Procesom znajdującym się w kolejce (traktowanej jako cykliczna) procesów gotowych do wykonania, przydzielane są odpowiednie odcinki czasu, nie dłuższe niż jeden kwant czasu.

Do implementacji algorytmu, podobnie jak w przypadku FCFS, używa się kolejki FIFO - brany jest pierwszy proces z wyjścia kolejki, ustawiany jest timer na przerwanie po upływie 1 kwantu czasu i proces jest wysyłany do procesora. Jeżeli czas trwania fazy procesu jest krótszy od 1 kwantu czasu, proces sam zwolni procesor i rozpocznie się obsługiwanie kolejnego procesu z kolejki. Jeżeli proces ma dłuższą fazę od 1 kwantu czasu, to nastąpi przerwanie zegarowe systemu operacyjnego, nastąpi przełączenie kontekstu, a proces do niedawna wykonywany, trafi na koniec kolejki procesów gotowych do wykonania.

Zobaczmy, ile wynosi średni czas oczekiwania w porównaniu z metodą FCFS dla następujących danych:

Proces	Czas trwania fazy
P ₁	21 ms
P ₂	6 ms
P ₃	3 ms

Dla kroku 7 ms diagram Gantta dla powyższych procesów wyglądał będzie następująco:



Średni czas oczekiwania wynosi więc: $(0+13+16)/3 = 9,67$ ms.

Wydajność algorytmu rotacyjnego zależy od kwantu czasu. W przypadku, gdy kwant czasu jest bardzo długi (dąży do nieskończoności) metoda RR działa jak FCFS. Natomiast w przypadku, gdy kwant czasu jest bardzo krótki (dąży do zera) planowanie RR nazywamy **dzieleniem procesora** (ang. *processor sharing*), gdyż zachodzi złudzenie, że każdy z n procesów ma własny procesor działający z prędkością 1/n prędkości prawdziwego procesora.

W przypadku algorytmu RR, podczas mówienia o wydajności, należy zastanowić się nad zagadnieniem przełączania kontekstu. Im mniejszy jest kwant czasu, tym system bardziej jest narażony na częste przełączania kontekstu. Jeżeli czas przełączania kontekstu wynosi 10 procent kwantu czasu, to niemalże 10 procent czasu procesora jest tracone w wyniku przełączania kontekstu.

ALGORYTM PLANOWANIA PRIORYTETOWEGO

Większość użytkowników komputerów osobistych zauważa z pewnością że wskazane jest aby niektóre zadania wykonywane były szybciej niż inne, które mogą być przesunięte w czasie. Prostym przykładem jest reakcja komputera na ruch myszy czy naciśnięcie klawisza – jeżeli system reaguje na te zdarzenia z widocznym opóźnieniem to komfort pracy jest znacznie obniżony, a użytkownik narzeka na „wolny komputer”. Jednocześnie to czy np. defragmentacja dysku lub indeksowanie jego zawartości działające w tle wykona się wolniej czy szybciej jest z punktu widzenia użytkownika nieistotne. Wynika z tego wniosek, że każde zadanie wykonywane przez komputer powinno mieć swój priorytet aby uszeregować je w zbiorze innych, równolegle wykonywanych. Aby wykorzystać informację o priorytecie procesu podstawowy algorytm planowania rotacyjnego musi zostać nieco zmodyfikowany. W tym celu stosuje się organizację uruchomionych procesów w szereg kolejek, po jednej dla każdego priorytetu.

W ramach każdej kolejki szeregowanie zadań odbywa się według zasad zwykłego algorytmu round robin, natomiast numer kolejki do uruchomienia dobierany jest tak aby faworyzowane były te z wyższym priorytetem (podobnie jak to ma miejsce w przypadku algorytmu SJF, tyle że tam kryterium był czas wykonania - tu jest priorytet kolejki).

Z takim podejściem wiąże się zagrożenie wstrzymywania zadań niskopriorytetowych w nieskończoność, nazywane **głodzeniem** (starving). W sytuacji, w której system jest bardzo obciążony, może zdarzyć się, że proces o niskim priorytecie będzie czekał w kolejce procesów gotowych w nieskończoność. Z tego powodu oryginalny algorytm planowania jest modyfikowany o dynamiczną regulację priorytetów procesów, często bazującą na ilości odwołań do systemów wejścia / wyjścia w danym procesie lub czasie oczekiwania na uruchomienie (priorytet długo oczekujących procesów jest zwiększany - fachowo nazywa się to **postarzaniem procesów**).

ALGORYTM MULTI-LEVEL FEEDBACK QUEUES (MFQ)

Rozwinięciem idei priority schedulingu jest algorytm multi-level feedback queues. Podobnie jak w przypadku planowania priorytetowego tutaj także procesy umieszczane są w kilku listach tworzonych dla każdego poziomu priorytetów.

Każda z kolejek może mieć jednak inny model zarządzania (niekoniecznie RR, a na przykład SJF) oraz wielkość kwantu czasu procesora. Generalną zasadą jest otrzymywanie krótkich kwantów przez procesy o wysokim priorytecie. Nowy proces otrzymuje wysoki priorytet i jest uruchamiany na bardzo krótki czas. Jeżeli w tym fragmencie wykonania proces nie wykonuje odwołań blokujących (czyli CPU jest zajęty przez cały kwant), to priorytet procesu jest obniżany, zwiększany za to jest jego kwant. Jeżeli proces odwołuje się do urządzeń wejścia / wyjścia to jego priorytet jest podnoszony. W efekcie najszybciej wykonywane są krótkotrwałe zadania, a także takie, które często odnoszą się do urządzeń wejścia / wyjścia (np. wprowadzanie danych przez użytkownika).

ALGORYTM O(1)

System operacyjny Linux w wersjach jądra od 2.6 do 2.6.23 wykorzystywał scheduler nazywany O(1). Cechą charakterystyczną (od której zresztą algorytm ten wziął nazwę) było to, że czas planowania uruchomienia dowolnego procesu był niezmienny, bez względu na liczbę procesów uruchomionych w systemie, co było osiągnięciem bardzo istotnym, zwłaszcza w przypadku zastosowań w systemie czasu rzeczywistego (niezmienność czasu wyznaczania schedulingu w dużym stopniu poprawia determinizm wykonywanych przez system czynności).

Zasada działania tego algorytmu opiera się na dwóch tablicach procesów – aktywnych i wygasłych (active, expired). Uruchomiony proces (znajdujący się w tablicy aktywnej) wykonywany jest przez określony czas (zdefiniowany w systemie) po czym jest on usypani, a informacja o nim przeniesiona z tablicy procesów aktywnych do uśpionych. W ten sposób tablica aktywna jest opróżniana i w momencie kiedy staje się pusta zostaje podmieniona tablicą procesów uśpionych, która z kolei zostaje zamieniona tablicą aktywną.

Algorytm ten automatycznie reguluje również priorytet danego procesu, analizując jego czasy aktywności (oczekiwania na dane), przyjmując że ten proces, który czeka dugo jest procesem związany z interfejsem użytkownika, który dla zachowania komfortu obsługi powinien mieć wysoki priorytet. Zadania obciążające procesor w dużym stopniu i nieoczekujące na dane mają ustawiany niski priorytet.

Niestety użytka metoda identyfikacji zadań oczekujących na dane (opierająca się na heurystyce) powoduje potrzebę wykonywania dużej liczby obliczeń (niepotrzebnie obciążających procesor) dodatkowo często dając niewłaściwe wyniki.

ALGORYTM CFS

Z powodu wyżej wymienionych wad opracowano nowy algorytm planowania procesów, zwany CFS czyli completely fair scheduler (całkowicie sprawiedliwy scheduler). „Sprawiedliwość” oznacza tutaj przydzielanie każdemu procesowi jednakowej, wynikającej z liczby procesów, mocy obliczeniowej procesora symulując działanie „idealnie wielozadaniowego procesora” (cyt. Ingo Molnar, autor CFS).

Założenia CFS realizowane są poprzez śledzenie całkowitego czasu oczekiwania na procesor, który minął od rozpoczęcia procesu. Czas ten, dzielony przez współczynnik wynikający z liczby równolegle uruchomionych procesów oraz priorytetu analizowanego procesu. Proces, którego czas oczekiwania jest największy, otrzymuje dostęp do procesora, a jego wait runtime jest odpowiednio zmniejszany. W momencie kiedy jakikolwiek proces ma współczynnik oczekiwania większy niż bieżący, następuje zmiana przydziału procesora.

Tego typu szeregowanie zadań pozwala na rezygnację z zaawansowanych i czasochłonnych algorytmów heurystycznych jak w schedulerze O(1). Algorytm CFS zamiast standardowych kolejek zadań do organizacji listy zadań używa tzw. struktury binarnych, czerwono-czarnych drzew (dowiecie się o takich strukturach więcej w trakcie przedmiotu Algorytm i Struktury Danych).

Interesującą właściwością algorytmu CFS jest to, że czas, na który procesor przypisywany jest poszczególnemu procesowi wynika z liczby aktualnie uruchomionych procesów i jest określany dynamicznie.

Nowoczesne systemy operacyjne używają różnych technik zarządzania procesami. Linux wykorzystuje CFS, Windows zmodyfikowany MFQ, podobnie jak Solaris czy FreeBSD. Rozwiążanie zastosowane w Mac OS X jest jeszcze bardziej skomplikowane - nazywa się Mach, i łączy cechy systemu ogólnego stosowania z systemem soft real time (czasu rzeczywistego w sensie miękkim).

WĄTKI

Wątek (ang. *thread*) określany również jako **proces lekki** (ang. *lightweight process* - LWP) to podstawowa jednostka wykorzystania procesora. W tym przypadku już nie możemy mówić o tym, że wątek to jest po prostu działający program - jeden działający program (proces) może składać się z wielu wątków.

W skład wątku wchodzą:

- licznik rozkazów;
- zbiór rejestrów;
- obszar stosu.

Wątek współużytkuje z innymi równorzędnymi wątkami sekcję kodu i danych oraz takie zasoby systemu operacyjnego jak otwarte pliki i sygnały, co łącznie nazywane jest **zadaniem** (ang. *task*).

Aby zrozumieć pojęcie wątku najlepiej porównać go z procesem tradycyjnym, tzw. *ciężkim* (ang. *heavyweight*). Proces = zadanie z jednym wątkiem. Takie właśnie równanie opisuje zależność pomiędzy procesem a wąkiem.

Wątki, w przeciwieństwie do procesów, można przełączać tanim kosztem, tj. z małym użyciem czasu procesora. Oczywiście przełączenie wątku również wymaga od systemu operacyjnego przełączenia zbioru rejestrów, jednakże nie jest konieczne zarządzanie pamięcią.

Porównując wątki i procesy, rozważmy sposób sterowania wieloma wątkami i wieloma procesami. Każdy z procesów działa niezależnie, posiada własny licznik rozkazów, wskaźnik stosu oraz przestrzeń adresową. W przypadku, gdy zadania wykonywane przez procesy nie są ze sobą powiązane, wszystko jest w porządku. Jednakże, gdy mamy do czynienia z jednym zadaniem wykonywanym przez kilka procesów - marnowane są zasoby systemu (każdy proces przechowuje osobne kopie tych samych informacji). Proces wielowątkowy natomiast, wykonujący jedno zadanie, zużywa mniej zasobów (wątki korzystają z tego samego zbioru informacji).

Wątki są podobne do procesów pod wieloma względami. Tak jak procesy, wątki mogą znajdować się w jednym z określonych stanów: gotowości, zablokowania, aktywności i zakończenia. Aktywny może być tylko jeden wątek (tak jak proces). Każdy wątek ma własny licznik rozkazów i stos, a jego wykonanie w procesie przebiega sekwencyjnie. Wątki, podobnie jak procesy, również mogą tworzyć wątki potomne i mogą blokować się do czasu zakończenia wywołań systemowych - możliwa jest więc sytuacja działania jednego wątku, gdy zablokowany jest inny. Różnicą pomiędzy wątkiem a procesem jest fakt, że wątki są zależne od siebie. Każdy z wątków ma dostęp do danych innego z wątków (dostęp do każdego adresu w zadaniu). Niemożliwa jest więc ochrona na poziomie wątków. Ponieważ jednak w obrębie jednego zadania wątki muszą należeć do jednego użytkownika, ochrona taka nie jest konieczna.

Różne systemy operacyjne w różny sposób traktują wątki. Popularne są dwie koncepcje:

- wątki są obsługiwane przez jądro - system zawiera zbiór funkcji do obsługi wątków;
- wątki są tworzone powyżej jądra systemu za pomocą funkcji bibliotecznych wykonywanych z poziomu użytkownika (ang. *user-level threads*).

Ponieważ wątki tworzone na poziomie użytkownika nie wymagają dostępu do jądra (korzysta się z nich za pomocą wywołań bibliotecznych zamiast odwołań do systemu), mogą być szybciej przełączane. Może jednak zdarzyć się sytuacja, w której po dowolnym wywołaniu systemowym cały proces musi czekać (nie jest więc realizowany), gdyż jądro planuje tylko procesy (nie wie o istnieniu wątków). Dlatego też, coraz częściej w systemach operacyjnych, stosuje się kombinację tych dwóch rozwiązań, realizując wątki w jądrze i z poziomu użytkownika.

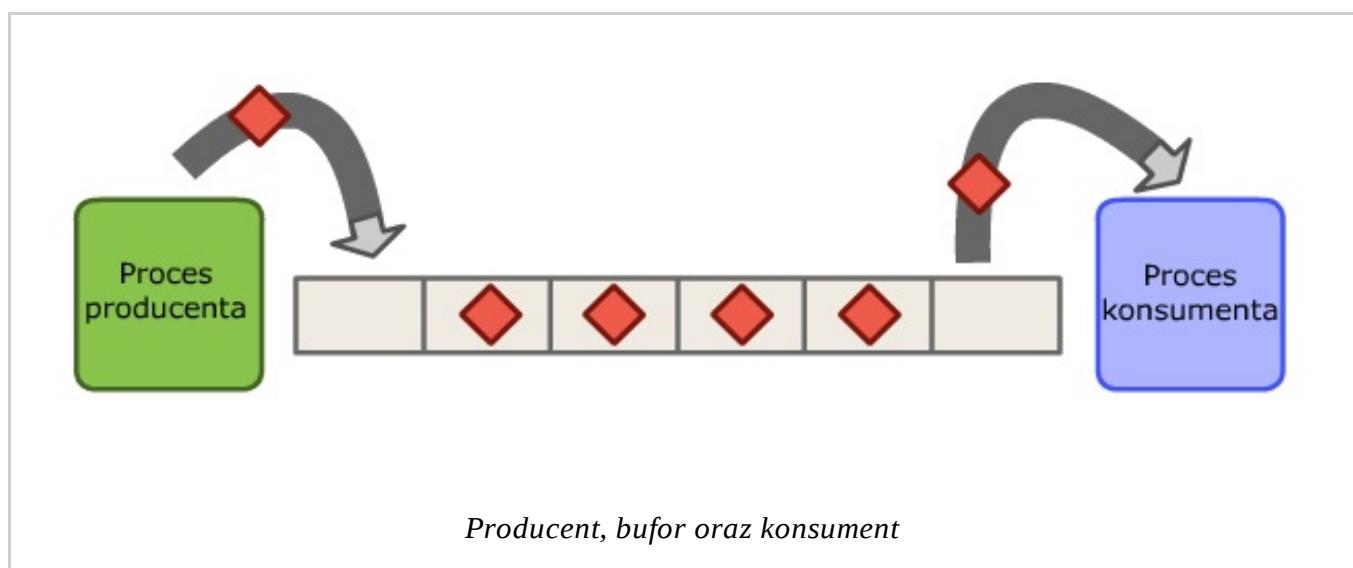
SYNCHRONIZACJA

Przetwarzanie danych w systemie współbieżnym polega na wykonywaniu instrukcji różnych procesów - to przetwarzanie jest naprzemienne lub w pełni współbieżne (w systemach z więcej niż jednym rdzeniem) - lecz w obu przypadkach instrukcje powodują modyfikację danych, i w rezultacie - modyfikację systemu jako całości. Często jest to problem pomijalny - gdy przetwarzane dane należą tylko i wyłącznie do jednego procesu. Lecz co w przypadku gdy procesy lub wątki chcą się nawzajem ze sobą komunikować?

W przypadku konieczności współpracy musimy zawsze zapewniać, że współdzielone zasoby (najczęściej dane) pozostaną w spójnym stanie. Dane pozostają w spójnym stanie wtedy, gdy ich pojedynczą paczkę może w danej chwili modyfikować tylko jeden z procesów, i żaden z innych procesów nie powinien jej w tym czasie modyfikować. Skoro tak - to potrzebne są jakieś mechanizmy zapewniające spójność danych. Postaramy się omówić tutaj w skrócie kilka najbardziej popularnych. Ale zaczniemy od przykładowego algorytmu współpracy:

ALGORYTM PRODUCENT / KONSUMENT

Zagadnienie producenta-konsumenta stanowi jeden z możliwych modeli współpracy procesów. Zakładamy, że proces producenta tworzy dane, które konsumuje proces konsumenta. Aby było to możliwe musi istnieć bufor (wyobraźmy sobie go jako tablicę) jednostek, który będzie zapełniany przez producenta i opróżniany przez konsumenta. W takim układzie, gdy producent produkuje jednostkę, konsument może wyciągnąć z bufora inną. Ważne jest jednak, aby procesy producenta i konsumenta podlegały synchronizacji tak, by konsument nie "wyciągał" jednostek jeszcze niewyprodukowanych, a w przypadku pustego bufora poczekał na wyprodukowanie czegoś do "wyciągania".



Taki typ bufora, jak widzicie w tym przykładzie, w programowaniu nazywa się kolejką FIFO (*First In - First Out*, pierwszy wchodzi - pierwszy wychodzi).

Problem producenta-konsumenta może być dwojakiego rodzaju:

- z nieograniczonym buforem (ang. *unbounded-buffer*) - możliwa jest sytuacja oczekiwania przez konsumenta na nowe jednostki, jednakże producent produkuje je nieustannie;
- z ograniczonym buforem (ang. *bounded-buffer*) - możliwa jest sytuacja oczekiwania przez konsumenta na nowe jednostki, ale możliwe jest również, że czekał będzie producent w wyniku

zapełnienia bufora.

Poniżej znajdziecie zapis algorytmu producenta-konsumenta w tzw. pseudokodzie, czyli uproszczonym języku programowania o składni zbliżonej do Pascal-a. Rzeczywiste implementacje są bardziej skomplikowane, lecz ten zapis powinien pozwolić Wam zrozumieć ideę działania takiego algorytmu. Przedstawiliśmy rozwiązanie problemu producenta-konsumenta z ograniczonym buforem, który używa pamięci dzielonej. W algorytmie tym procesy konsumenta i producenta korzystają z następujących wspólnych zmiennych:

var n; type jednostka=...; var bufor: array [0..n-1] of jednostka; we, wy: 0..n-1;	// rozmiar bufora // jednostka do konsumpcji // bufor jako tablica cykliczna // wskaźniki do wejścia i wyjścia bufora
--	---

Zmienna *we* wskazuje na następne wolne miejsce w buforze. Gdy bufor jest pusty wskaźniki *we* i *wy* pokrywają się. Gdy bufor jest pełny to prawdziwy jest warunek *we+1 mod n = wy* (*mod* oznacza resztę z dzielenia).

Poniżej przedstawione są kody procesów producenta i konsumenta. Warto jeszcze wspomnieć iż instrukcja *nic* (ang. *NOP*) jest instrukcją pustą tzn. jej wykonanie nie daje żadnego efektu. Jednostki nowo produkowane znajdują się w zmiennej lokalnej *nowyprodukt*, zaś jednostki do skonsumowania są przechowywane w zmiennej *nowakonsumpcja*.

Proces producenta:

```
repeat
...
// produkuj jednostkę w nowyprodukt
...
while we+1 mod n=wy do nic;
bufor[we]:=nowyprodukt;
we:=we+1 mod n;
until false
```

Proces konsumenta:

```
repeat
while we=wy do nic;
nowakonsumpcja:=bufor[wy];
wy:=wy+1 mod n;
...
// konsumuj jednostkę z nowakonsumpcja
...
until false
```

Podany algorytm zakłada, że wszelkie operacje wykonywane na pamięci dzielonej są tak zwymi **operacjami atomowymi** - to oznacza, iż w momencie wykonywania takiej operacji proces nie może zostać przerwany. Jeśli mógłby zostać przerwany - to oznaczałoby, że np. paczka wyprodukowana przez producenta jest tylko w połowie umieszczona w buforze (reszta to śmieci). Wtedy proces producenta został wywolany, proces konsumenta wznowiony - i zacząłby pobierać połowiczne dane jako kompletną całość.

I tak doszliśmy do momentu, w którym czasem coś w programie komputerowym nie działa, ale zdarza

się to tylko czasami, i nikt nie wie dlaczego tak się zdarza ... ;)

SYNCHRONIZACJA

Wracając do głównego zagadnienia niniejszej części - skoro w systemie komputerowym możemy uruchomić więcej niż jeden program, i każdy z nich ma prawo do korzystania ze wspólnych zasobów (przyjmijmy jako przykład zapis danych na dysk), to w jaki sposób zapewnić, że zapis wykonywany przez proces A nie zostanie przerwany przez algorytm przydzielu czasu procesora zanim zostanie skończony? I że następny w kolejce proces B nie zniszczy naszych częściowo zapisanych danych?

Unikanie takich problemów jest możliwe dzięki mechanizmom synchronizacji procesów i wątków.

Celem synchronizacji jest kontrola przepływu sterowania między procesami tak, by dopuszczalne stały się tylko przeploty instrukcji dopuszczone przez programistę (by nie zachodziła interferencja w dostępie do współdzielonych danych i zasobów).

Mechanizm powstrzymywania procesów przed dostępem do danego zasobu jeżeli jest on używany przez jakikolwiek inny proces, nazywa się **wzajemnym wykluczaniem** (mutual exclusion) a zasoby chronione – **zasobami krytycznymi**. Blokowanie to nie może odbywać się w pełni automatycznie (tzn. przez system operacyjny) - realizacja automatycznego mechanizmu wykluczeń jest zbyt trudna, choć istnieją języki programowania które takie automatyczne blokady umozliwiają (np. Java) kosztem niższej wydajności. System operacyjny udostępnia jednakże mechanizmy blokowania i kontroluje je.

Odwołanie do zasobu krytycznego w procesie odbywa się w tzw. **sekcią krytyczną** (critical section). Mechanizm sekcji krytycznych musi być zaimplementowany w kodzie procesu, a realizowany jest najczęściej poprzez użycie udostępnianych przez system operacyjny metod synchronizacyjnych.

SEMAFORY

Przy omawianiu złożonych zagadnień synchronizacji wygodnie jest posługiwać się narzędziem synchronizacji nazywanym **semaforem** (ang. *semaphore*). Semafor jest strukturą zawierającą zmienną typu całkowitego, której można nadać wartość początkową i do której można się odwoływać tylko za pośrednictwem dwu niepodzielnych (atomowych) operacji: P (hol. *proberen*, ang. *wait*, pol. *czekaj*) oraz V (hol. *verhogen*, ang. *signal*, pol. *sygnalizuj*). Niepodzielność tych instrukcji jest zapewniana przez system operacyjny, często przy wsparciu procesora.

Definicja semafora przedstawia się następująco:

```
poczekaj(S):
while S<=0 do nic;
S:=S-1;
... wykonuj zadania ...

zasygnalizuj(S):
S:=S+1;
```

Ważne jest, aby zmiany wartości semafora wykonywane za pomocą operacji *poczekaj* i *zasygnalizuj* uniemożliwiały sytuację, w której dwa procesy próbowałyby modyfikować wartość semafora. Dodatkowo, podczas operacji *poczekaj(S)*, nie może wystąpić przerwanie w trakcie sprawdzania wartości zmiennej S (*dopóki S<=0*) oraz jej aktualizacji.

MUTEXY

Semafony stosuje się często przy rozwiązywaniu problemów dostępu do sekcji krytycznej, której przydzielone jest kilka procesów. Procesy te korzystają ze wspólnego semafora nazywanego w tym przypadku **mutex** (ang. *mutual exclusion*), zapewniającego wzajemne wykluczanie procesów tak, aby zapewnić dostęp tylko jednemu spośród nich. Organizacja każdego z procesów korzystających z semafora przedstawia się następująco:

```
repeat
    poczekaj(mutex);
    ...
    sekcja krytyczna
    ...
    zasygnalizuj(mutex);
    ...
    reszta działań procesu
    ...
until false;
```

Semafony stosujemy także przy rozwiązywaniu problemów związanych z synchronizacją. Jeżeli na przykład chcielibyśmy, aby dwa procesy P-X i P-Y, zawierające odpowiednio rozkazy S-X i S-Y, wykonały się sekwencyjnie, tzn. w kolejności X, potem Y. Możliwa jest prosta realizacja takiego problemu, gdy wprowadzimy wspólną dla obu procesów zmienną *synchronizacja*, nadawszy jej wartość początkową "0". Zakładając, że dołączymy do procesów P-X i P-Y instrukcje:

P-X:	P-Y:
S-X; sygnalizuj(synchronizacja);	czekaj(synchronizacja); S-Y;

możliwe stanie się wykonanie najpierw kodu S-X, później S-Y, ponieważ dopóki zmienna *synchronizacja* wynosi "0", semafor *czekaj* blokuje wykonanie instrukcji S-Y. Wykonanie S-Y możliwe staje się dopiero po wykonaniu rozkazu *sygnalizuj(synchronizacja)*.

IMPLEMENTACJA

Obsługa semaforów wymaga **aktywnego czekania** (ang. *busy waiting*). Polega to na tym, że procesy chcąc dostać się do sekcji krytycznej, zajmowanej przez jakiś proces, zmuszone są do wykonywania pętli w sekcji wejściowej. Aktywne czekanie marnuje czas procesora, który mógłby wykonywać nieco ważniejsze zadania. Semafony tego typu nazywa się też **wirującą blokadą** (ang. *spinlock*).

Wirujące blokady w przypadku założenia, że czas oczekiwania procesu pod blokadą jest bardzo krótki, stają się bardzo użyteczne w środowiskach wieloprocesorowych. Ich zaletą jest fakt, że postój procesu pod semaforem nie wymaga przełączania kontekstu, który może trwać długo.

Nie trzeba jednak traktować aktywnego czekania jako niezniszczalnego. Aktywne czekanie można bowiem wyeliminować. Aby było to możliwe, proces, który musiałby czekać aktywnie, powinien mieć możliwość zablokowania się (ang. *block*). Zablokowanie się procesu powoduje umieszczenie go w kolejce do danego semafora oraz zmianę stanu procesu na "czekanie", po czym wybierany jest do wykonania kolejny z procesów. Wznowienie działania procesu zablokowanego powoduje sygnał *zasygnalizuj* pochodzący od innego procesu. Wykonywana jest wtedy operacja budzenia (ang. *wakeup*), która zmienia stan procesu na "gotowość". Powoduje to przejście procesu do kolejki procesów gotowych do wykonania przez procesor.

Semafor, o którym tak już dużo napisaliśmy można zrealizować jako prosty rekord:

```
type semaphore = record
wartosc: integer;
L: list of proces;
end;
```

Jak widzimy semafor składa się z dwóch pól: wartości całkowitej i listy procesów, które muszą czekać pod semaforem. Operacje na tak zdefiniowanym semaforze można przedstawić następująco:

poczekaj(S):	S.wartosc:=S.wartosc-1; if S.wartosc<0 then begin dodaj proces do S.L; blokuj; end;
zasygnalizuj(S):	S.wartosc:=S.wartosc+1; if S.wartosc>=0 then begin uzuń proces P z S.L; obudz(P); end;

Tak zaimplementowane operacje *poczekaj* i *zasygnalizuj* nie eliminują całkowicie efektu aktywnego czekania, jednakże zapewniają usunięcie tego zjawiska z wejść do sekcji krytycznych programów użytkowych. Ma to ogromne znaczenie, gdyż sekcje krytyczne tych programów mogą zajmować dużo czasu lub być często zajęte. Aktywne czekanie w takich warunkach mogłoby być fatalnym posunięciem i zakończyłoby się dużym spadkiem wydajności systemu.

ZDARZENIA

Zdarzenia (events) są kolejnymi obiektami mogącymi służyć do synchronizacji. Ich ogólnym przeznaczeniem jest komunikacja międzyprocesowa, a dokładniej przekazywanie sygnałów do procesów lub wątków poprzez ustawienie danego obiektu zdarzenia (event object) w stan sygnalizujący (signalled state). Tworzenie obiektu zdarzenia inicjowane jest przez pierwszy z synchronizowanych procesów (CreateEvent), w ramach inicjalizacji ustawiane są również jego parametry takie jak nazwa, tryb kasowania stanu (automatyczny lub ręczny). Wszystkie inne procesy i wątki które chcą korzystać z tego obiektu mogą uzyskać do niego wskaźnik poprzez wywołanie funkcji OpenEvent, której jedynym wymagany parametrem jest nazwa oczekiwanej zdarzenia. Z racji zdolności sygnalizacji stanów procesom obiekty te mogą być również wykorzystywane do synchronizacji dostępu do danych.

ZAKLESZCZENIE I GŁODZENIE

Nieodpowiednie wykorzystywanie semaforów, mutex-ów i im podobnych elementów może prowadzić do nielubianych przez programistów sytuacji, które pospolici użytkownicy komputerów nazywają "powieszeniem się" aplikacji. Mowa tutaj o zakleszczeniu (ang. *deadlock*), czyli stanie, gdy każdy proces w jakimś zbiorze procesów oczekuje na zdarzenie, które może być spowodowane tylko przez inny proces z tego zbioru. Dla semaforów zdarzeniem tym jest operacja *zasygnalizuj*.

Przykładem takiego zdarzenia są procesy Px i Py, które korzystają z semaforów A i B, o wartościach "1", których postać przedstawią się następująco:

Px:

```
poczekaj(A);    poczekaj(B);  
poczekaj(B);    poczekaj(A);  
.               .  
.               .  
zasygnalizuj(A); zasygnalizuj(B);  
zasygnalizuj(B); zasygnalizuj(A);
```

Py:

W takiej sytuacji powstaje zakleszczenie, gdyż proces Px musi czekać na wykonanie operacji *zasygnalizuj(A)* przez proces Py, zaś Proces Py czeka na wykonanie przez Px rozkazu *zasygnalizuj(B)*. I tak będą czekać w nieskończoność ... podobnie jak użytkownik tak napisanej aplikacji.

Kolejnym problemem związanym z semaforami jest głodzenie, czyli konieczność nieskończonego oczekiwania przez proces pod semaforem. Problem ten jest wynikiem niskiego priorytetu danego procesu. Proces jest ignorowany np. w wyniku dużego natłoku innych procesów, które mają wyższe przerwania. Najczęściej głodzenie pojawia się w przypadku użycia kolejki LIFO (ang. *Last In First Out*) jako listy procesów oczekujących do semafora.

PRZYDZIAŁ ZASOBÓW I ZAKLESZCZANIE

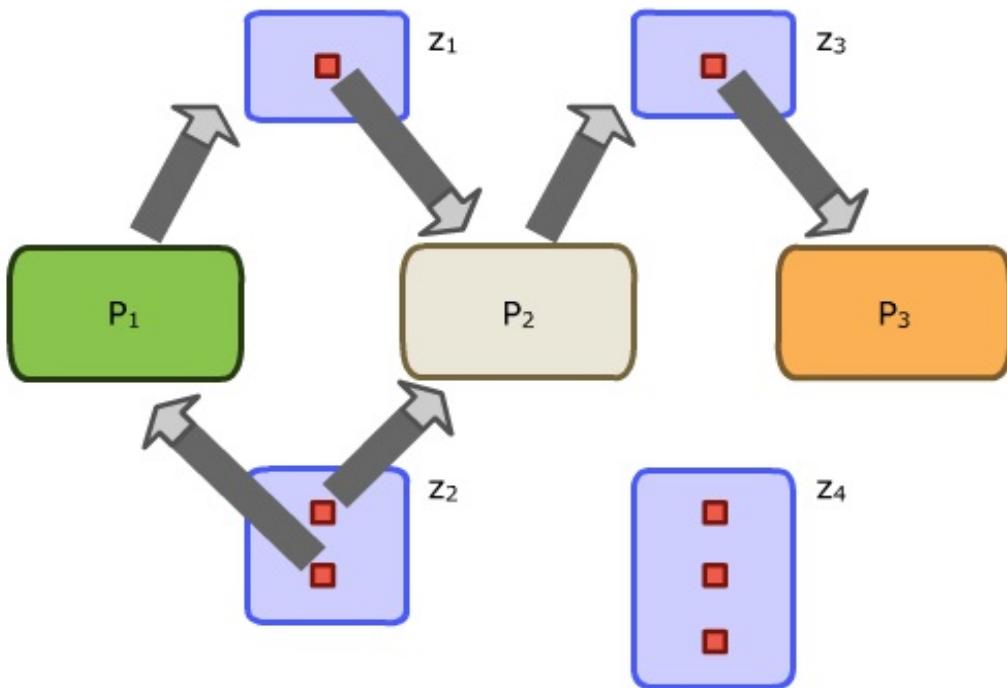
W przypadku próby uzyskania dostępu do współdzielonych zasobów dość często występuje problem zakleszczania. Ba - nie będzie przesadą stwierdzenie, że - obok zarządzania pamięcią - zakleszczenie jest największą zmorą programistów. Poniżej znajdziecie opis kilku metod, które pozwalają radzić sobie z tym nieprzyjemnym zjawiskiem.

GRAF PRZYDZIAŁU ZASOBÓW

Graf przydziału zasobów (ang. *system resource-allocation graph*) bardzo dobrze nadaje się do opisywania zakleszczeń. Składnikami tego skierowanego grafu są:

1. wierzchołki W, składające się z węzłów:
 - o $P=\{P_1, P_2, \dots, P_n\}$ - wszystkie procesy systemu;
 - o $Z=\{Z_1, Z_2, \dots, Z_n\}$ - wszystkie typy zasobów systemowych;
2. krawędzie K, które dzielimy na:
 - o krawędzie skierowane od procesu P_x do zasobu Z_y (zapisujemy $P_x \rightarrow Z_y$) - oznacza to zamówienie zasobu Z_y przez proces P_x ;
krawędzie zwane są *krawędziami zamówienia* (ang. *request edge*);
 - o krawędzie skierowane od zasobu Z_y do procesu P_x (zapisujemy $Z_y \rightarrow P_x$) - oznacza to przydzielenie zasobu Z_y do procesu P_x ;
krawędzie te nazywa się *krawędziami przydziału* (ang. *assignment edge*).

Ponadto zbiór procesów P na grafie przedstawia się w postaci kólek, zaś zbiór typów zasobów Z w postaci prostokątów. Każdy z typów zasobów Z_y może mieć więcej niż jeden egzemplarz, zaznaczany jako kropka w prostokącie zasobu. Podczas gdy krawędź zamówienia dochodzi tylko do boku prostokąta zasobu, krawędź przydziału rozpoczyna się od konkretnego egzemplarza (czytaj kropki) typu zasobu wewnątrz prostokąta.



Graf przydziału zasobów

Sposób działań systemu odpowiadających zmianom krawędzi jest następujący:

1. Zamówienie przez proces P_x zasobu Z_y powoduje pojawienie się krawędzi zamówienia.
2. Realizacja zamówienia powoduje zamianę krawędzi zamówienia w krawędź przydziału.
3. Brak jakiekolwiek krawędzi oznacza, że proces zwolnił już niepotrzebny mu zasób.

Czytając o grafie przydziału zasobów, na pewno zastanawiacie się: co to ma wspólnego z zakleszczeniami? Jeśli graf zawiera cykle (pętle) może dojść do zakleszczeń, jeżeli nie zawiera cykli sytuacja taka jest niemożliwa. A czym jest cykl? Na rysunku wyżej nie występuje żaden cykl. Jeżeli jednak wyobraźlibyśmy sobie sytuację, w której proces P₃ zamawia zasób Z₂ (pojawia się wtedy krawędź zamówienia P₃ - Z₂), to w tak zmodyfikowanym układzie występują dwa cykle:

- P₁ → Z₁ → P₂ → Z₃ → P₃ → Z₂ → P₁
- P₂ → Z₃ → P₃ → Z₂ → P₂.

Zakleszczeniem możemy określić sytuację, w której każdy z zasobów w cyklu ma tylko jeden egzemplarz. W przypadku natomiast, gdy któryś z zasobów w cyklu ma większą liczbę egzemplarzy niż 1 nie przesąduje to o występowaniu zakleszczenia.

W przedstawionym powyżej grafie systemu, do zakleszczenia dojdzie na pewno, gdy proces P₃ zechce skorzystać z zasobu Z₂.

PRZETRZYMANIE I OCZEKIWANIE

Warunek przetrzymywania i oczekiwania jest spełniony, gdy istnieje proces, któremu został przydzielony co najmniej jeden zasób. Ponadto proces ten oczekuje na przydział kolejnego zasobu, który jest używany przez inny proces.

Zapobiegniemy zakleszczeniu wtedy i tylko wtedy, gdy uda nam się dokonać negacji powyższego

warunku. Musimy więc doprowadzić do sytuacji, w której proces zamawiający zasób nie posiada sam żadnych zasobów. Opiszemy pokrótce dwa protokoły realizujące takie założenie:

1. Należy umieścić wywołania funkcji systemowych, opisujących zamówienia zasobów dla procesu, za wywołaniami wszystkich innych funkcji systemu.
2. Umożliwić procesowi zamawianie zasobów wtedy i tylko wtedy, gdy proces nie posiada żadnych zasobów (przed zamówieniem nowych zasobów proces musi oddać wszystkie zasoby, z których w danej chwili korzysta).

Podstawową wadą obu powyższych rozwiązań jest to, że *wykorzystanie zasobów* może być zaskakującym małe, gdyż niemożliwe będzie korzystanie z dużej części przydzielonych zasobów przez długi czas. Drugą z wad opisanych protokołów jest możliwość wystąpienia głodzenia. Proces, który chce skorzystać z "rozchwytywanego" zasobu, może czekać do nieskończenie długo tylko dlatego, że zasób ten będzie ciągle w posiadaniu innego procesu.

CZEKANIE CYKLICZNE

Warunek czekania cyklicznego występuje wtedy i tylko wtedy, gdy istnieje zbiór procesów czekających $P=\{P_0, P_1, \dots, P_n\}$, takich że P_0 oczekuje na zasób będący w posiadaniu procesu P_1 , P_1 zaś czeka na zasób, który jest w posiadaniu procesu P_2, \dots , a P_n oczekuje na zwolnienie zasobu przez proces P_0 .

Najbardziej popularnym sposobem uniknięcia czekania cyklicznego jest uporządkowanie wszystkich typów zasobów i dopilnowanie, by każdy z procesów otrzymywał te zasoby rosnąco. Przyporządkujemy więc jednoznacznie każdemu zasobowi należącemu do zbioru $Z=\{Z_0, Z_1, \dots, Z_i\}$ liczbę całkowitą (w celu określenia rosnącego porządku procesów). Na początku proces może zamówić dowolną liczbę egzemplarzy wybranego typu zasobu. Potem proces może zamawiać jedynie egzemplarze tych typów, które mają większe numery. Jeśli na przykład na początku proces zamówił zasób Z_2 , to później musi zamawiać zasoby $Z_x > Z_2$. Należy zaznaczyć przy tym, że możliwe jest zamówienie kilku egzemplarzy tego samego typu z użyciem jednego tylko zamówienia.

TYPOWE PROBLEMY SYNCHRONIZACJI

ALGORYTM PRODUCENT/KONSUMENT W WERSJI OGÓLNEJ

Wróćmy do przykładu [producenta - konsumenta](#) z początku tej lekcji. Jeżeli przyjmiemy, że mamy n buforów, a każdy z nich mieści 1 jednostkę, oraz że semafor *mutex* (umożliwiający wzajemne wykluczanie dostępu do buforów) jest na starcie równy 1, zaś semafory *pusty* i *pełny* zawierają informację o liczbie pustych (wartość początkowa n) i pełnych (wartość początkowa 0) buforów, to ogólna wersja algorytmu wygląda tak:

```
repeat
  ...
  produkuj jednostkę w nowyprodukt
  ...
  poczekaj(pusty);
  poczekaj(mutex);
  ...
  dodaj jednostkę nowyprodukt do bufora bufor
  ...
  zasygnalizuj(mutex);
  zasygnalizuj(pełny);
```

```
until false
```

Proces producenta ...

```
repeat
    poczekaj(pełny);
    poczekaj(mutex);
    ...
    wyjmij jednostkę z bufora bufor do nowakonsumpcja
    ...
    zasygnalizuj(mutex);
    zasygnalizuj(pusty);
    ...
    konsumuj jednostkę z nowakonsumpcja
    ...
until false
```

... i proces konsumenta.

Instrukcje *poczekaj* oznaczają oczekiwanie z powodu faktu, że bufor jest pusty lub pełny. Instrukcja *zasygnalizuj* opisuje sytuację, w której bufor staje się pusty lub pełny, a także uruchomienie semafora *mutex*.

PROBLEM CZYTELNIKÓW I PISARZY

Problem ten odnosi się do środowisk, w których zasoby mogą być dzielone. W takich środowiskach może zachodzić sytuacja dostępu do danego zasobu przez kilka procesów jednocześnie. Niektóre z tych procesów odczytują tylko zawartość obiektu dzielonego, inne zapisują do niego informacje (aktualizacja). Procesy odczytujące dane z obiektu dzielonego to *czytelnicy*, zaś zapisujące dane to *pisarze*.

Problem czytelników i pisarzy (ang. *readers-writers problem*) polega na zapewnieniu "prywatności" pisarzowi. Nie może zdarzyć się sytuacja, w której dany obiekt współdzielony jest przez pisarza i inny proces, czy to czytelnika, czy to innego pisarza - nie może istnieć sytuacja w której książka ma kilku autorów, oraz w której ktoś czyta nieskończone dzieło.

Problem ten ma kilka odmian. Najłatwiejsza z nich nazywana jest pierwszym problemem czytelników i pisarzy. Metoda ta zakłada, że żaden czytelnik nie musi czekać (pomimo tego, że pisarz jest gotowy) na zakończenie pracy pozostałych czytelników. W odniesieniu do świata informatyki - istnieje różnica w dostępie do zapisu i do odczytu. Dostęp do zapisu jest dostępem na wyłączność - nikt w tym czasie nie może pisać ani czytać. Dostęp do odczytu jest dostępem współdzielonym - ciągle nikt inny nie może pisać, ale czytać może wielu.

Kolejna odmiana problemu czytelników i pisarzy zakłada, że gotowość pisarza oznacza, że żaden nowy czytelnik nie będzie mógł współpracować z obiektem. Ponieważ oba przedstawione powyżej schematy mogą powodować głodzenie, stosuje się również inne wersje rozwiązania tego problemu.

Poniżej zostanie opisane rozwiązanie pierwszego problemu czytelników i pisarzy. Założymy, że wszystkie z procesów mają wspólne następujące zmienne:

var mutex, pisarz: semaphore; liczba_czytelnikow: integer;	// wartość początkowa "1" dla obu semaforów // wartość początkowa "0"
--	---

Przed napisaniem struktury procesu czytelnika warto jeszcze nadmienić, że semafor *mutex* zajmuje się wzajemnym wykluczaniem procesów podczas nadpisywania zmiennej *liczba_czytelnikow*. Semafor *pisarz* daje gwarancję wzajemnego wykluczania pisarzy, a ponadto jest używany przez pierwszy proces wchodzący lub ostatni proces wychodzący z sekcji krytycznej czytelnika. Semafor ten nie jest natomiast używany przez czytelników wchodzących/wychodzących podczas, gdy inni czytelnicy znajdują się w sekcjach krytycznych. Rozwiążanie naszego problemu.

Tak wygląda struktura procesu pisarza:

```
poczekaj(pisarz)
...
teraz proces sobie pisze i pisze, i pisze ...
...
zasignalizuj(pisarz);
```

Struktura procesu czytelnika wygląda następująco:

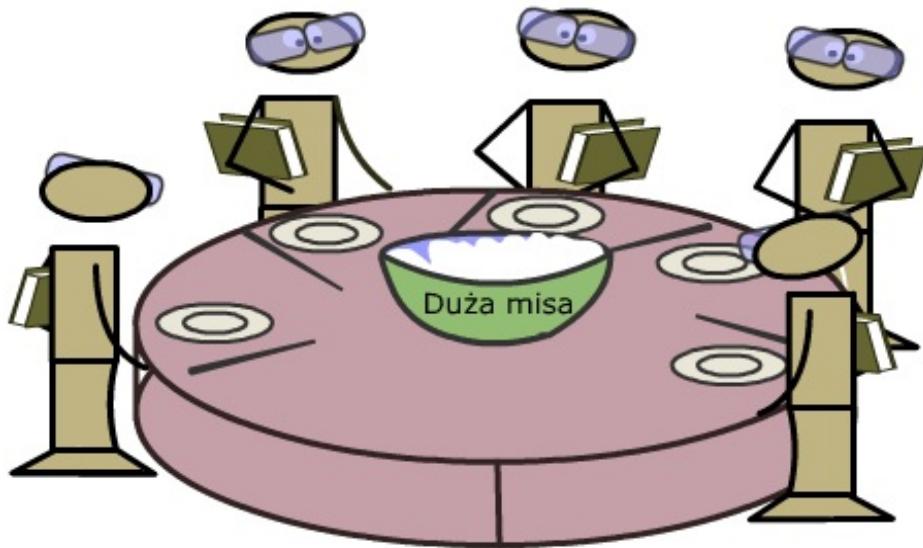
```
poczekaj(mutex);
liczba_czytelnikow:=liczba_czytelnikow+1;
if liczba_czytelnikow=1 then poczekaj(pisarz);
zasignalizuj(mutex);
...
proces czyta sobie i czyta, i czyta ...
...
poczekaj(mutex);
liczba_czytelnikow:=liczba_czytelnikow-1;
if liczba_czytelnikow=0 then zasignalizuj(pisarz);
zasignalizuj(mutex);
```

Po wykonaniu przez proces pisarza operacji *zasignalizuj(pisarz)* możliwe są dwie drogi: można wznowić działanie pojedynczego pisarza lub czekających czytelników, zależnie od wyboru planisty (scheduler-a).

PROBLEM GŁODUJĄCYCH FILOZOFÓW

Uwaga, wybieramy się na wycieczkę. Znajdujemy się teraz w miejscu bliżej nieokreślonym, w którym jest tylko jeden mebel - stół. Przy tymże stole siedzi pięć osób. Wszyscy mają przed sobą miseczki, a na środku stołu stoi misa z ryżem. Ponadto, na stole znajduje się pięć pałeczek rozmieszczonych pomiędzy miseczkami nieznanych nam osób. Po dokładniejszych oględzinach okazuje się, że ci ludzie są filozofami. A wiecie dlaczego? Ponieważ przez cały czas myślą, a gdy są głodni sięgają po pałeczki, aby zjeść trochę ryżu, potem znowu myślą. I tu dochodzimy do naszego problemu. Jak nasz drogi filozof ma wziąć dwie pałeczki, skoro sąsiad też może chcieć coś zjeść? Przecież filozof nie będzie wyrywał siłą koleżanek pałeczki z ręki.

Mogą być przypuszczać, że problem filozofów nie ma wiele wspólnego z synchronizacją - lecz tak nie jest. Problem obiadujących filozofów jest klasycznym problemem konieczności przydziału wielu zasobów do wielu procesów w warunkach zagrożenia głodzeniem i zakleszczeniem.



Pięciu współczesnych obiadujących filozofów

Przedstawimy proste rozwiązanie tego problemu, oparte na semaforach, nie pozbawione jednak niebezpieczeństw. Operacja *poczekaj* oznacza podniesienie przez filozofa pałeczki, zaś operacja *zasygnalizuj* oznacza odłożenie pałeczki. Nasza pałeczka wygląda natomiast następująco:

```
var pałeczka: array[0..4] of semaphore; // wartość początkowa=1
```

Po wielu zmaganiach umysłowych będziemy wreszcie mogli zobaczyć jak wygląda struktura n-tego filozofa.

```
repeat
    poczekaj(pałeczka[n]);
    poczekaj(pałeczka[n+1 mod 5]);
    ...
    jedzenie, jedzenie, jedzenie, np. ryżu
    ...
    zasygnalizuj(pałeczka[n]);
    zasygnalizuj(pałeczka[n+1 mod 5]);
    ...
    myślnie nie boli, więc filozof myśli
    ...
until false
```

Struktura procesu n-tego filozofa

Rozwiązanie przedstawione powyżej nie jest najlepsze. Daje co prawda gwarancję, że dwaj sąsiadzi nie będą jedli jednocześnie, jednakże w przypadku gdy każdy z filozofów w tym samym czasie weźmie pałeczkę (by było to możliwe każdy z filozofów musi wziąć pałeczkę leżącą po tej samej stronie, np. prawej) tak, że wszystkie pałeczki zostaną podniesione (elementy tablicy pałeczka są równe "0"), nie będzie możliwe podniesienie przez którykolwiek z filozofów pałeczki drugą ręką. Zatrzymamy się w pętli nieskończonej - zakleszczymy się.

Naturalnie możliwa jest eliminacja takich zakleszczeń. Przykładowe sposoby eliminacji tego zagrożenia:

- do stołu zasiada czterech filozofów, podczas gdy jest wolnych pięć miejsc;
- filozof może podnosić pałeczki wtedy i tylko wtedy, gdy po prawej i po lewej stronie są one dostępne;
- wprowadzenie asymetrii - filozofowie nieparzyści zaczynają podnosić pałeczki z lewej strony, zaś parzyści ze strony prawej.

INTERFEJS SYSTEMU PLIKÓW

W trakcie tej lekcji chcielibyśmy przedstawić Wam jeden z najważniejszych modułów systemu operacyjnego z punktu widzenia użytkownika - czyli podsystem obsługi plików.

To od systemu plików w dużej mierze zależy szybkość działania pamięci masowych oraz bezpieczeństwo danych na nich umieszczonych. Dodatkowo, o ile użytkownik systemu operacyjnego raczej nie ma wpływu na algorytmy zarządzania pamięcią, kolejkowania procesów czy też zapobiegania zakleszczeniom (opisane w poprzedniej lekcji), o tyle najczęściej może sam zadecydować, jaki system plików ma wykorzystywać dla swojej stacji roboczej, czy też na serwerze którym się opiekuje.

Mamy nadzieję, że wiedza o właściwościach i możliwościach poszczególnych systemów plików pomoże w prawidłowym i świadomym wyborze systemu plików.

PLIKI I ICH ORGANIZACJA

Pojęcie pliku

Dane bezpośrednio potrzebne procesorowi do wykonywania jego zadań są umieszczone w pamięci operacyjnej systemu. Jest to jednak pamięć ulotna i dane w niej zapisane są tracone po wyłączeniu zasilania. Nie nadaje się więc ona do trwałego magazynowania informacji. Ponadto, rozmiar pamięci operacyjnej jest stosunkowo niewielki i dodatkowo ograniczony od góry poprzez szerokość szyny adresowej komputera. Dlatego też informacje magazynuje się w sposób trwały na różnych nośnikach takich jak magnetyczne dyski i taśmy lub dyski optyczne.

Plik jest pojęciem logicznym i definiuje podstawową logiczną jednostkę magazynowania informacji. Jest to ciąg bitów danych, opatrzony nazwą i atrybutami. Dane są zapisane w odpowiednim formacie, zależnie od zawartości pliku. Nazwa z reguły składa się z głównej części nazwy i rozszerzenia (oddzielonego kropką). Podział ten jest charakterystyczny dla systemu MS-DOS (gdzie musiał być konsekwentnie przestrzegany), w innych systemach operacyjnych (np. Windows, Linux) nazwa może składać się z wielu części. Z punktu widzenia użytkownika systemu operacyjnego plik jest całością, w rzeczywistości przechowywany jest on na fizycznym nośniku w innej formie (np. jako zestaw powiązanych ze sobą bloków). Sposób przechowywania plików, a także możliwe struktury na jakie dzielony jest fizyczny nośnik, zależą w dużej mierze od systemu operacyjnego i noszą nazwę systemu plikowego (z ang. *file system*).

Najważniejszą cechą pliku zapisanego na dysku jest jego nieulotność - po zaniku systemu zawartość nie jest tracona. Jednakże z punktu widzenia teorii informatyki ta cecha jest nieistotna (co więcej - czasem nieprawdziwa, istnieją systemy plików tworzone w pamięci RAM). Dla informatyka najważniejszą cechą pliku jest jego sekwencyjność, czyli fakt, że dane w pliku są zapisane kolejno i kolejno muszą być z niego czytane. W każdym pliku istnieje znacznik, nazywany kursorem, mówiący o tym, który następny bajt będzie z niego odczytyany / do niego zapisany.

Kiedyś pojęcie pliku było jednym z fundamentalnych pojęć informatyki - wszystkie urządzenia wejścia / wyjścia przez system operacyjny były traktowane jako pliki. Aktualnie istnieje duża grupa tzw. urządzeń blokowych (o swobodnym dostępie, jak karta graficzna), a pojęcie pliku ograniczyło się głównie do plików tworzonych na pamięciach masowych.

RODZAJE I TYPY PLIKÓW

Dane zapisane w pliku zależą od jego twórcy i mogą być bardzo różne. Ze względu na charakter zapisanych informacji pliki dzieli się na:

- programy;
- dane.

Dane zapisane w plikach mogą być różnego rodzaju. Mogą to być dane liczbowe, teksty, teksty z dodatkowymi informacjami o formatowaniu, obrazy grafiki komputerowej, nagrania dźwiękowe, zapisy sekwencji wideo itd.

Struktura pliku, czyli sposób jego wewnętrznej organizacji, zależy od rodzaju zapisanych informacji. Mówimy, że struktura pliku zależy od jego typu.

Przykładowe typy plików (dla systemu Windows) zostały zebrane w tabeli poniżej.

Typ pliku	Rozszerzenie nazwy	Funkcja
Wykonywalny	exe, com, bin	Gotowy do wykonania program w języku maszynowym
Kod źródłowy	c, pas, asm, vb, cpp	Kod źródłowy wyrażony w różnych językach
Wsadowy	bat	Polecenia dla interpretera poleceń
Tekstowy	txt	Dane i dokumenty tekstowe
Obraz	png, gif, jpg, bmp, tiff	Plik binarny, zawierający obraz rastrowy
Archiwalny	zip, arj, rar, 7z	Plik skompresowany odpowiednim algorytmem kompresji

Wewnętrzna organizacja struktury pliku jest uzależniona od typu pliku i od wymagań programu z niego korzystającego.

Z plikiem związanych jest kilka *atrybutów* (cech). Należą do nich m.in.:

- **nazwa** – ciąg znaków identyfikujących plik w sposób zrozumiały dla człowieka;
- **typ** – określa „rodzaj” przechowywanych informacji, na przykład: wykonywalny, wynikowy, kod źródłowy, tekstowy, archiwalny itd. Typ rozpoznawany jest najczęściej przez tak zwane rozszerzenie, choć we współczesnych systemach już tak nie musi. Plik wykonywalny może mieć dowolne rozszerzenie, lub nie mieć go w ogóle (jak to jest najczęściej w przypadku Unixów);
- **położenie** – pozwala na fizyczną lokalizację na nośniku;
- **rozmiar** – zawiera wielkość pliku najczęściej wyrażoną w bajtach;
- **ochrona** – informacje o tym kto i w jaki sposób może sprawować kontrolę i odczytywać informację z pliku;
- **data** – data utworzenia, modyfikacji, ostatniego użycia, wydrukowania itp.

W różnych systemach mogą występować różne zestawy atrybutów. Przykładowo - w systemach Unix i pochodnych, to czy plik jest wykonywalny czy nie, nie jest rozpoznawalne poprzez rozszerzenie, lecz jako dodatkowy atrybut.

OPERACJE NA PLIKACH

Zazwyczaj to system operacyjny zapewnia mechanizmy dostępu i zarządzania plikami, choć nie jest powiedziane, że podobne mechanizmy nie mogą być również wbudowane w oprogramowanie użytkowe (np. sieciowe czy wirtualne systemy plików).

Każdy sensowny system operacyjny powinien zapewnić następujące operacje na pliku:

- tworzenie - utworzenie nowego, najczęściej pustego pliku, nadanie mu nazwy oraz udostępnienie do dalszych operacji;
- zapis - wypełnienie pliku danymi. Plik nie ma określonego sztywno rozmiaru - jego wielkość dostosowywuje się do ilości zapisanych w nim danych;
- odczyt - operacja odwrotna do zapisu - pobranie danych z pliku do pamięci. Przy odczytce dane w pliku pozostają nienaruszone;
- usuwanie - fizyczne usunięcie pliku z dysku wraz z danymi w nim zawartymi;
- skracanie - może nieco niefortunna nazwa. Skracanie polega na usunięciu z pliku wszystkich zawartych w nim danych, bez usuwania pliku jako takiego. W efekcie otrzymujemy na dysku pusty plik o długości 0;
- przenoszenie - zmiana położenia pliku w strukturze katalogów;
- kopiowanie - wykonanie kopii zawartości pliku pod inną nazwą lub w innym położeniu w strukturze katalogów.

W różnych systemach operacyjnych mogą występować dodatkowe operacje na plikach.

Należy podkreślić, że plik jest strukturą logiczną, tak samo jak jego struktura wewnętrzna. Zapis pliku na dysku musi natomiast być zgodny z fizyczną organizacją nośnika danych. Z tego powodu zapis pliku na dysku odbywa się całymi blokami, które często nie pokrywają się z logiczną wielkością pliku. W takim wypadku część przestrzeni dyskowej pozostaje niewykorzystana, a zjawisko to znane jest pod nazwą fragmentacji.

SYSTEM PLIKÓW

Pierwsze komputery, pracujące w trybie wsadowym pobierały dane i zwracały wyniki obliczeń w postaci kart perforowanych, taśm magnetycznych itp. Nie posiadały więc żadnego mechanizmu przechowywania jakichkolwiek zbiorów danych. Kolejny poziom ewolucji komputerów i systemów operacyjnych umożliwiał już jednak zapamiętywanie i odczytywanie danych użytkowników tak, aby były one dostępne po np. zakończeniu procesów, wyłączeniu urządzenia itd. Obecnie w zasadzie każde urządzenie komputerowe potrafi odczytywać i zapisywać dane z jakiejś formy pamięci stałej (taśmach, dyskietkach, dyskach twardych, dyskach optycznych, pamięci flash itd.). Narzędziem umożliwiającym organizację danych na nośnikach, jest system plików (czyli zbiorów danych), często zintegrowany z systemem operacyjnym z racji głębokich powiązań z jądrem systemu i wywołaniami pracującymi w trybie specjalnych uprawnień (kernela).

Głównym celem systemów plików (SP) jest jak wspomniano przechowywanie danych w sposób zorganizowany, umożliwiający łatwe ich zapisanie oraz odnalezienie i odczytanie. System plików to mechanizm pozwalający na uporządkowane przechowywanie oraz bezpośredni dostęp do informacji (dane i programy) zawartej w plikach. Dodatkowo, system plików pozwala na zarządzanie plikami (kopiowanie, przenoszenie, usuwanie) oraz organizuje je w logiczną strukturę. We współczesnych systemach operacyjnych bezpośrednie operowanie na danych w plikach zarezerwowane jest tylko dla systemu operacyjnego, aplikacje mają dostęp tylko do operacji na plikach i mają zabroniony bezpośredni dostęp do nośnika danych.

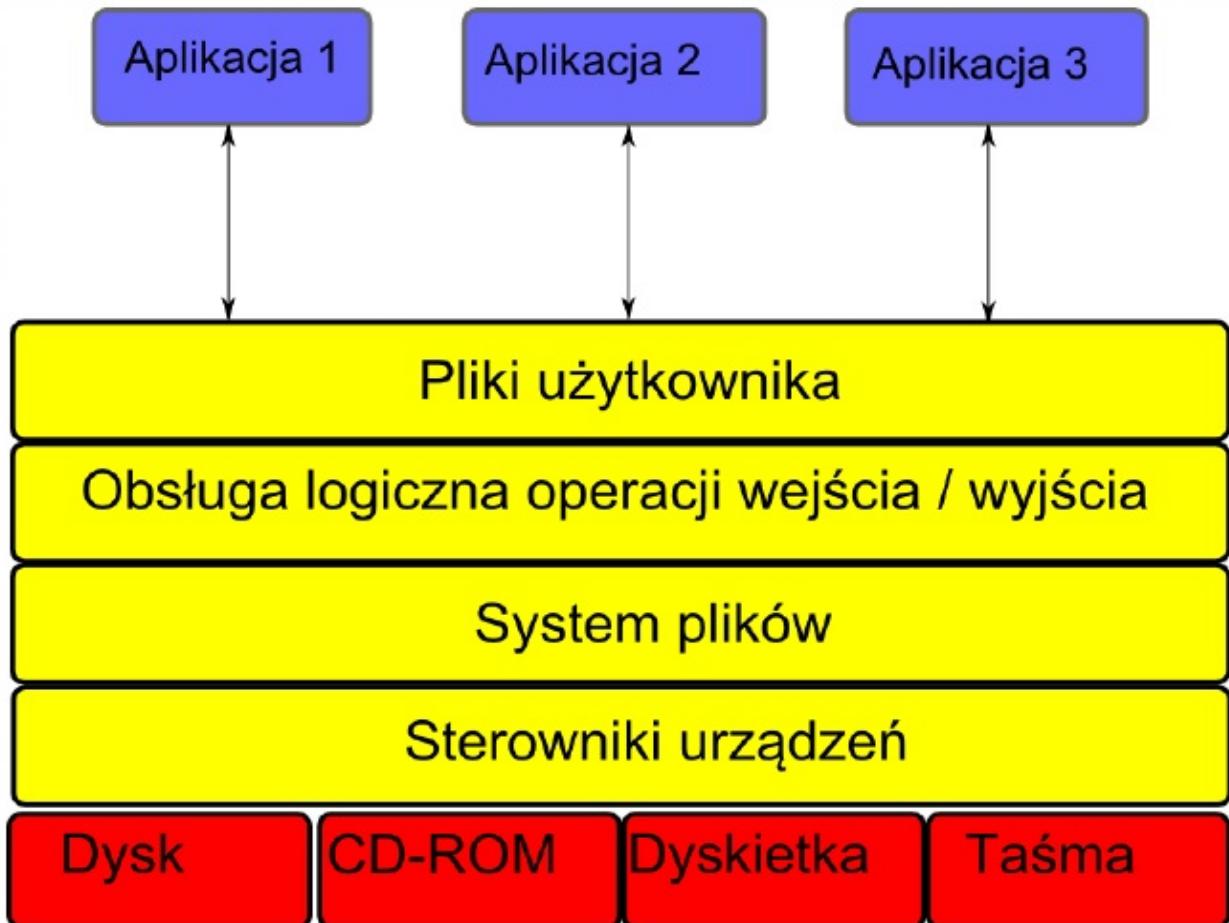
System plików jest jednym z bardziej widocznych elementów systemów operacyjnego. W jego skład wchodzi zbiór plików oraz struktura katalogów, która pozwala na łatwą organizację i udostępnianie informacji o wszystkich plikach zapisanych w systemie. W niektórych systemach występuje jeszcze dodatkowy element służący do wyodrębnienia w sposób fizyczny lub logicznych dużych zbiorów katalogów.

Głównym zadaniem systemu pliku w systemie komputerowym jest zapewnienie prawidłowego i wydajnego dostępu do plików zapisanych w pamięci pomocniczej oraz zapewnienie prawidłowej ochrony przechowywanej informacji.

Ważnymi cechami SP są:

- długoterminowość (przechowywanie danych po wylogowaniu się użytkownika je tworzącego, wyłączeniu zasilania itp.)
- możliwość wykorzystywania danych przez różne procesy
- możliwość organizacji plików w struktury odzwierciedlające wzajemne relacje między nimi (np. w strukturę katalogów i podkatalogów)

System plików wraz z biblioteką funkcji służących do jego obsługi można przedstawić za pomocą schematu warstwowego jak poniższym rysunku:



Schemat warstwowy systemu plików

Aplikacje i programy użytkowników komunikują się z systemem poprzez wywołania systemowe. W zależności od programisty, pliki mogą być traktowane jako binarne, tekstowe, przechowujące rekordy itd. Wywołania systemowe przetwarzane są przez warstwę odpowiedzialną za komunikację z urządzeniem, przetwarzającą informacje określające plik użytkownikom (nazwę, położenie, atrybuty) na informacje o fizycznym lokalizacji danych w nim zawartych na nośniku. Warstwa ta zarządza również ewentualnym buforowaniem pliku (lub jego części) w pamięci operacyjnej komputera. Odwołania do fizycznych parametrów nośnika (np. numer sektora) są przekazywane do sterownika urządzenia fizycznego, który dba o przekazanie odpowiednich sygnałów do układów elektronicznych nośnika (np. pozycjonowania głowicy w dysku twardym).

Zazwyczaj pliki organizowane są w grupy, nazywane katalogami.

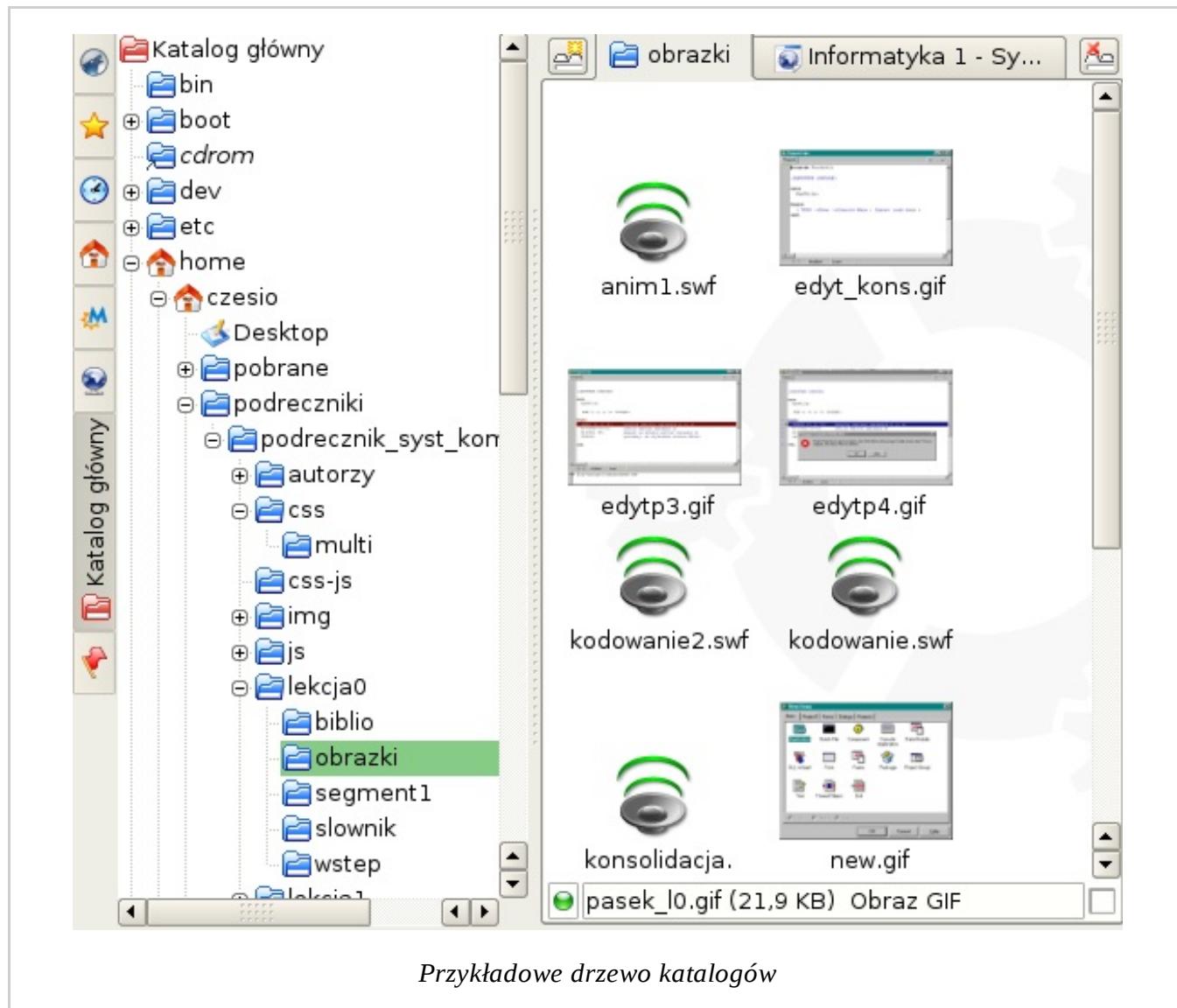
STRUKTURA KATALOGÓW

Katalog określa położenie pliku na dysku. Operacje wykonywane na plikach (takie jak odnajdywanie, odczyt czy usuwanie) są wykonywane przy wykorzystaniu katalogów. Dlatego też organizacja struktury katalogu jest dla systemu bardzo istotna.

Najprostszym sposobem organizacji katalogu jest struktura jednopoziomowa (w praktyce niestosowana). W takiej strukturze informacje o wszystkich plikach ujęte są w jednym poziomie. Katalog o strukturze jednopoziomowej jest łatwo utworzyć i obsługiwać. Ma on jednak sporo wad. Uwypuklają się one przede wszystkim w momencie, gdy rośnie liczba plików lub różnych

użytkowników systemu. W takiej strukturze każdy plik musi mieć swoją unikatową nazwę, a różni użytkownicy nie mogą w ten sam sposób nazywać swoich plików. Trudności pojawiają się również w trakcie wyszukiwania informacji o zadanym pliku.

Tych wad pozbawiona jest, najczęściej dziś wykorzystywana, struktura katalogów zorganizowana na kształt drzewa.



W takiej strukturze nie występuje problem jednoznaczności nazw (w ujęciu globalnym) oraz problem oddzielenia plików różnych użytkowników, gdyż każdy z nich może przechowywać swoje pliki w osobnej części struktury drzewiastej.

W katalogu drzewiastym drzewo ma swój katalog główny (ang. *root*), a każdy plik w systemie oprócz swojej nazwy ma jednoznaczną ścieżkę określającą jego położenie w drzewie katalogów.

Katalog, w którym znajduje się plik, jest nadzędny (bezpośrednio) względem tego pliku. Katalog jest podzędny (bezpośrednio) względem innego katalogu w którym się znajduje.

Format podanej ścieżki zależy od systemu operacyjnego. W większości systemów (Unix, Linux, Mac OS) katalog jest reprezentowany przez ukośnik (ang. *slash* "/"), pełna ścieżka do pliku o nazwie *edytp3.gif* z rysunku powyżej wygląda następująco:

```
/home/czesio/podreczniki/podrecznik_syst_komputerowe/lekcja0/obrazki/edytp3.gif
```

W przypadku systemów Windows oraz DOS katalog jest reprezentowany przez lewy ukośnik (ang. *backslash* "\), przy oznaczaniu pełnej ścieżki uwzględnia się również literę dysku logicznego (volumenu). Tak więc ścieżka dostępu do pliku o tej samej nazwie w Windows mogłaby wyglądać następująco:

```
C:\Documents and Settings\czesio\moje dokumenty\podreczniki\podrecznik_syst_komputerowe\lekci
```

Oprócz pojęcia ścieżki bezwzględnej, w systemach opartych na drzewie istnieje również pojęcie względnej ścieżki do pliku - czyli ścieżki jaką z aktualnego katalogu możemy dostać się do punktu docelowego. W tym celu poruszamy się w górę drzewa do pierwszego katalogu wspólnego, korzystając przy tym ze pomocą specjalnego oznaczenia katalogu nadzawanego - dwóch kropek. Wracając do naszego przykładu z rysunku, jeśli aktualnym katalogiem jest

```
/home/czesio/podreczniki/podrecznik_syst_komputerowe/css/multi
```

to pierwszym wspólnym katalogiem jest

```
/home/czesio/podreczniki/podrecznik_syst_komputerowe
```

a względową ścieżką do pliku edyt3p będzie

```
../../../../lekcja0/obrazki/edytp3.gif
```

W systemie Windows to przejście będzie wyglądało podobnie, tylko ukośniki zmieniamy na odwrotne ukośniki.

DYSK LOGICZNY

Wspomnieliśmy, iż w systemach DOS / Windows przy podawaniu pełnej ścieżki podaje się na początek nazwę (literę) dysku logicznego. W pozostałych systemach operacyjnych takie dyski również istnieją - tyle, że nie są traktowane w sposób szczególny, a po prostu przypisane do jednego z katalogów w drzewie katalogów. Czym zatem są owe dyski logiczne?

Dysk logiczny, nazywany też partycją (ang. partition) czy wolumenem (ang. volume), jest wydzieloną strefą na fizycznym nośniku, o własnej strukturze wewnętrznej oraz systemie plików

Podziału na partie dokonuje się w celu:

- uzyskania kilku rozłącznych obszarów na jednym dysku fizycznym;
- zwiększenia rozmiaru strefy ponad rozmiar pojedynczego dysku fizycznego (w wypadku partycji montowanych na kilku dyskach fizycznych).

Każda partycja zawiera wszystkie zapisane w niej dane (zorganizowane w pliki umieszczone w katalogach). Dane umieszczone na jednej partycji są zupełnie niezależne od danych umieszczonych na innych partycjach.

Niektóre operacje na plikach (takie, jak zmiana nazwy, przeniesienie pliku w obszarze jednej partycji, czy skasowanie pliku) nie operują bezpośrednio na danych zapisanych w pliku, a jedynie zmieniają wpisy w odpowiednich dla danego systemu plików strukturach kontrolnych. Dzięki temu wspomniane operacje z jednej strony wykonywane są bardzo szybko, a z drugiej specjalisci mogą bez problemu odzyskiwać skasowane z systemu dane bezpośrednio po ich usunięciu - bo fizycznie cały czas pozostają one na dysku.

ZABEZPIECZENIA

Najprostsze systemy plików nie uwzględniają konieczności ochrony poszczególnych plików i katalogów przed dostępem - każdy użytkownik systemu komputerowego może czytać i zmieniać w zasadzie każdy plik umieszczony na pamięci masowej. Podejście takie charakteryzuje się prostotą koncepcyjną, lecz jego stosowanie w praktyce, poza komputerami domowymi, można określić jako mało rozsądne ...

Po pierwsze - skoro każdy użytkownik danego systemu komputerowego może czytać wszystkie umieszczone na nim pliki, to nie ma możliwości, by dane tworzone przez jednego pracownika były niedostępne dla innych. Co więcej - nie ma też możliwości stwierdzenia, kto dany plik utworzył oraz kto go modyfikował.

Najgorszą cechą systemów plików bez zabezpieczeń jest jednak możliwość uszkodzenia systemu operacyjnego przez użytkownika. Skoro każdy plik może być modyfikowany (w tym usunięty) przez każdego użytkownika, to mało doświadczony (lub złośliwy) użytkownik komputera jest w stanie skasować pliki należące do systemu operacyjnego. Czym to się kończy? Łatwo się domyśleć... System operacyjny przestaje działać i cały komputer przestaje być użyteczny.

Systemem plików pozbawionym zabezpieczeń jest opisany w następującym segmencie FAT.

W poważnych zastosowaniach stosuje się więc systemy plików pozwalające na zabezpieczanie zarówno pojedynczych plików jak i całych katalogów przed dostępem. Najczęściej przy tym stosuje się jeden ze schematów zabezpieczeń:

- prostsza i szybsza metoda - to wykorzystanie pojedyńczych uprawnień nadawanych każdemu plikowi i katalogowi na dysku. Takie podejście jest stosowane przez wszystkie popularne systemy plików przeznaczone dla Unix-ów. Każdy plik i katalog, oprócz swojej nazwy, jest opisany przez dwie dodatkowe liczby: identyfikator użytkownika do którego należy oraz identyfikator grupy użytkowników.
Dodatkowo mamy zestaw 9 bitów uprawnień, po trzy - dla użytkownika, dla grupy i dla pozostałych. Te trzy bity to: bit pozwolenia na odczyt pliku, jego modyfikacje (zapis, kasowanie) oraz uruchomienie (w przypadku katalogów - na wejście do katalogu).

```
drwxr-xr-x  3 czesio czesio  4096 2007-04-09 13:05 .
drwxr-xr-x 25 czesio czesio  4096 2007-04-09 12:37 ..
-rw-r-----  1 czesio czesio   343 2007-02-11 00:40 README
-rwxr-x---  1 czesio czesio 22562 2007-04-03 22:21 tempest
-rwxr-x---  1 czesio czesio 19722 2007-02-11 01:54 tempest.c
drwxr-xr-x  2 czesio czesio  4096 2007-04-09 13:05 tempest-dane/
-rw-r-----  1 czesio czesio   388 2007-02-11 00:37 tempest.desktop
-rw-r-----  1 czesio czesio  6234 2007-02-04 00:56 vroot.h
```

Przykład uprawnień do plików w systemie Linux

Na rysunku powyżej widzicie wyświetlzoną przykładową listę uprawnień (to pierwsze 10 znaków). Litera d oznacza katalog, a następnie mamy trzy grupy po trzy litery : rwx. Jeśli litera nie występuje, to oznacza że uprawnienie nie zostało przyznane. I tak, dla pliku o nazwie README, jego właściciel ma prawo na odczyt i modyfikacje, użytkownicy należący do grupy do której należy plik - mają prawo do odczytu, ale już nie mogą pliku modyfikować, natomiast pozostały nie mają prawa ani na odczyt, ani na modyfikację pliku;

- bardziej zaawansowana i elastyczna jest druga metoda - listy dostępu (ACL), stosowana w systemach plików HPFS i NTFS dla Windows. W tym przypadku nie każdy plik i katalog ma nadawane niezależnie prawa. Jeśli dany plik nie ma powiązanej z nim listy uprawnień (ACL), to przejmuje on listę z katalogu nadziedznego. Jeśli katalog również nie ma przyznanej takiej listy, to

sprawdzany jest katalog wyższego poziomu - i tak aż do momentu gdy lista zostanie odnaleziona. Takie podejście nazywa się dziedziczeniem uprawnień - pliki i katalogi dziedziczą uprawnienia po katalogu do którego należą.

W dowolnym miejscu drzewa plików można zdefiniować nową listę uprawnień dla plików. Lista uprawnień składa się z jednego lub więcej wpisów, z których każdy ma dwa pola:

- identyfikator użytkownika lub grupy;
- zestaw uprawnień (prawo do odczytu, zapisu, modyfikacji pliku, oraz tworzenia nowych plików w danym katalogu).

Do jednego pliku / katalogu uprawnienia można nadawać niezależnie dla kilku / kilkunastu użytkowników lub grup użytkowników. Jak widać, taka struktura jest zdecydowanie bardziej elastyczna. Jej wadą jest mniejsza wydajność, bo przy odwołaniu się do każdego pliku należy przeszukać w góre drzewo plików, w celu odnalezienia list dostępu i sprawdzenia uprawnień, które można nadać użytkownikowi.

KSIĘGOWANIE

Zabezpieczanie przed nieautoryzowanym dostępem nie daje jeszcze całkowitej pewności bezpieczeństwa zapisanych danych. Bezpieczeństwo to także pewność, że w wypadku np. awarii zasilania bądź też zawieszenia się systemu operacyjnego nie stracimy danych na dysku. Skoro operacje zapisu, tworzenia lub usuwania plików lub katalogów nie są wykonywane natychmiast, a także nic nie zabrania systemowi operacyjnemu normalnego wywłaszczenia zadania zapisu na dysk (w informatyce mówi się, iż nie są to operacje atomowe), to może się zdarzyć, że na dysku znajdą się nie do końca zapisane pliki, bądź - co gorzej - nie do końca zmienione wpisy w strukturach kontrolnych systemu plików - metadanych.

System plików musi przechowywać, oprócz danych zapisywanych w plikach, także dane o samej strukturze plików i strukturze systemu plików (tzw. metadane). Stąd biorą się problemy ze spójnością podczas awarii systemu, gdy nastąpi ona między operacjami na metadanych i operacjami na danych (w trakcie jednej operacji np. zapisu).

Dlaczego? Oto prosty przykład: chcielibyśmy zapisać listę wypłat naszych pracowników na dany miesiąc w pliku "scorge.xls". Aby tego dokonać wybieramy odpowiednią opcję z menu, w okienku które się pojawiło wpisujemy nazwę pliku i naciskamy przycisk "Zapisz". Od tego momentu dla nas - jako użytkowników - wszystko przebiega niezauważenie, ale w praktyce wykonywane są następujące operacje przez system operacyjny:

1. Na dysku wyszukiwany jest pierwszy wolny blok, gdzie możemy zapisywać.
2. W tablicy alokacji plików (lub w odpowiednim węźle) dodawany jest wpis oznaczający przypisanie danej nazwy pliku do aktualnego katalogu oraz przed chwilą znalezionego bloku (co to jest tablica alokacji plików oraz i-node wyjaśnimy przy opisie konkretnych systemów plików w następnych segmentach).
3. Sterowanie jest przekazywane do aplikacji - zaczyna ona przesyłać dane na dysk.
4. W momencie, gdy w aktualnym bloku zabraknie miejsca, wyszukiwany jest następny wolny i odpowiednio modyfikowane są metadane... i tak aż do końca zapisu.

Teraz wyobraźmy sobie, że Zakład Energetyczny wyłączy nam prąd pomiędzy krokiem 2 a 3. Wtedy okaże się, że struktury kontrolne zawierają informację o pliku którego tak na prawdę nie ma ... pojawił się nam plik-widmo. Aby uniknąć podobnych sytuacji, stosuje się księgowanie.

Księgowanie lub kronikowanie (ang. journaling) to w informatyce termin związany z konstrukcją baz danych oraz systemów plików. Przy użyciu księgowania dane nie są od razu zapisywane na dysk, tylko zapisywane w swoistym dzienniku/kronice (ang. journal). Dzięki takiemu mechanizmowi działania zmniejsza się prawdopodobieństwo utraty danych: jeśli utrata zasilania nastąpiła w trakcie zapisu - zapis zostanie dokonczony po przywróceniu zasilania, jeśli przed - stracimy tylko

ostatnio naniesione poprawki, a oryginalny plik pozostanie nietknięty.

Transakcje zapewniają zapis jednej operacji dyskowej (zapisu, tworzenia lub usuwania pliku (katalogu)) do dziennika jako operacji atomowej. Jest ona zapisywana w całości i tylko jeśli wszystkie jej kroki przejdą pomyślnie może zostać uznana za wykonaną - jest potwierdzana i usuwana z dziennika.

W momencie uruchomienia systemu po awarii wszystkie operacje w całości zapamiętane w dzienniku są wykonywane, a te które nie są zapisane w całości są wymazywane z dziennika i ignorowane (na pewno nie zostały nawet rozpoczęte na dysku).

W przypadku systemów plików bez dziennika, aby przywrócić porządek po nagłej awarii, trzeba uruchamiać specjalny program poprawiający błędne wpisy, inaczej grożą nam:

- wycieki wolnego miejsca (po przerwanej alokacji nowych bloków lub przy usuwaniu pliku został on już skasowany, ale jego bloki nie są jeszcze zaznaczone jako usunięte);
- złe dane o wolnym miejscu w grupie (np. w Linux-owym Ext3 źle będzie działał algorytm przydzielania wolnego miejsca);
- *plik1* zawiera wpisy pośrednie mogące pokazywać na bloki, których system już nie zdążył zająć przed awarią, następnie *plik2* je zajmuje, a my teraz zwalniamy *plik1* i *plik2* traci dane!

Dla systemów plików bez kroniki może się też zdarzyć sytuacja, że mimo uruchomienia programu sprawdzającego - wszystkich błędów nie uda się skorygować. A nawet jeśli uda się - to sprawdzenie całego dysku dla systemu bez księgowania może trwać bardzo długo (czas liczy się w dziesiątkach minut dla dużych dysków twardych).

System plików z kroniką po restarcie zwykle przywraca spójność dysku w czasie rzędu kilku, kilkunastu sekund. Restart systemu po awarii w systemach z kronikowaniem nie zależy od wielkości partycji, co jest szczególnie istotne w przypadku wielkich dysków.

Rozróżniamy dwa podstawowe typy kronikowania:

- tylko metadanych (zapewnia spójność metadanych, ale może nie pamiętać ostatnio wpisanych danych tuż przed awarią);
- metadanych i danych (odtworzy wszystko co zostało w całości wprowadzone do dziennika).

DOSTĘPNE SYSTEMY PLIKÓW

FAT

System plików FAT (ang. *File Allocation Table*) jest nierozerwalnie związan z MS DOS (i jeszcze wcześniejszego systemu dla maszyn 8-bitowych - CP/M). FAT jest jego natywnym (przewidzianym przez twórców) systemem plików.

W formacie FAT partycja (poza początkowymi zarezerwowanymi sektorami) jest podzielona na klastry (jednostki alokacji pliku). Każdy klaster składa się z jednego lub kilku sektorów, klastry są numerowane. System operacyjny na podstawie numeru klastra oblicza numer logiczny sektora (numer sektora od początku partycji), a na tej podstawie numer ścieżki, głowicy i sektora na ścieżce, identyfikując jednoznacznie sektor i dokonując odczytu lub zapisu wybranego sektora.

Klaster jest dla FAT jednostką niepodzieloną - nie można go wykorzystać w części, można jedynie w całości przydzielić jednemu plikowi.

Pliki w systemie FAT organizowane są w katalogi. Każdy wpis w katalogu zawiera nazwę pliku, atrybuty oraz numer pierwszego przydzielonego mu klastra. Natomiast lokalizację dalszej części pliku definiuje wpis w tablicy alokacji plików.

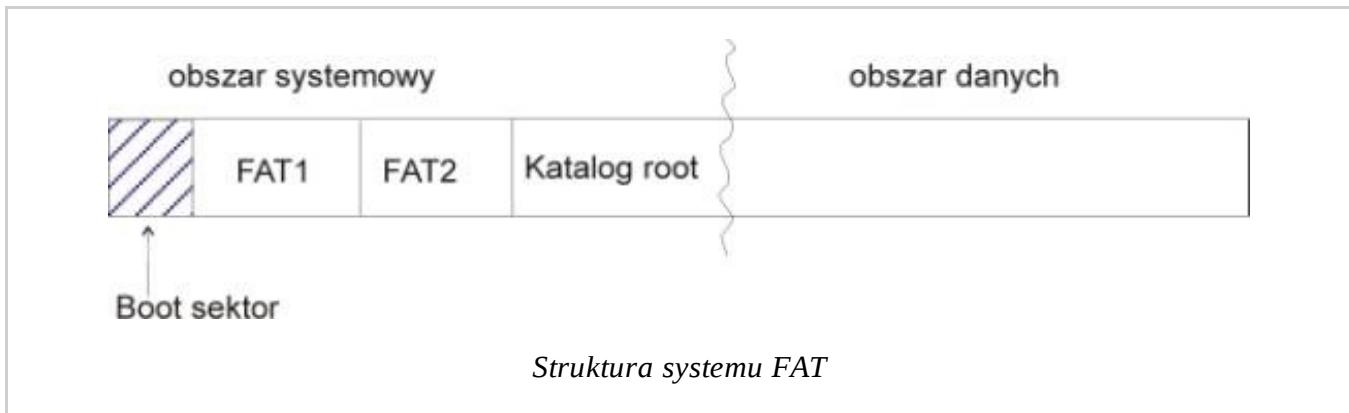
Tablica alokacji plików to przechowywana zaraz za boot sektorem struktura, zajmująca kilka sektorów, która zawiera tyle pozycji, ile jest na dysku klastrów. Każda pozycja w tablicy odpowiada jednemu klastrowi.

Teraz, jeśli w katalogu mamy zamieszczony wpis o tym, gdzie znajduje się pierwszy klaster danego pliku, to w tablicy alokacji plików pod numerem odpowiadającym numerowi pierwszej części pliku jest umieszczony numer kolejnego klastra przydzielonego plikowi lub liczba z zakresu FFF8h-FFF7h, jeśli to jest ostatni klaster pliku. Jeżeli dany klaster jest wolny, to w FAT odpowiada mu wpis 0000h, a FFF7h oznacza uszkodzony klaster.

Tablica FAT umożliwia szybkie odszukanie miejsca dla nowego pliku lub dalszej części oraz łatwe odszukiwanie kolejnych części plików. Wadą systemu FAT jest to, że dla partycji o dużej pojemności klaster jest równie duży i znaczna jego część pozostaje pusta, więc tracimy miejsce na dysku, i w efekcie, w przypadku dużych dysków twardych jesteśmy w stanie efektywnie wykorzystywać tylko ok. 60% ich pojemności. Ponadto w trakcie zapisywania i kasowania plików ulegają one fragmentacji (kolejne fragmenty pliku mogą leżeć w znacznej odległości od siebie), a przestawianie głowic dysku zajmuje czas. Inną wadą jest to, że każde założenie, skasowanie, każda zmiana wielkości pliku pociąga za sobą konieczność zmiany tablicy FAT, co przy niepoprawnym działaniu komputera może doprowadzić do utraty wszystkich danych na partycji. Pewnym zabezpieczeniem jest przechowywanie (realizowane w sposób automatyczny) dwóch kopii tablicy alokacji plików.

Dokładny podział każdej partycji w systemie FAT wygląda następująco:

- obszar zastrzeżony (z boot sectorem) – a w nim tablica BPB (Bios Parameter Block) oraz program ładowający system operacyjny (boot sector) dla partycji systemowej. Blok BPB zawiera informacje potrzebne do wyliczenia położenia i rozmiaru pozostałych regionów;
- tablica alokacji plików opisana wyżej. Na partycji może być kilka kopii tablicy FAT (zazwyczaj dwie);
- katalog główny i jego podkatalogi zawierające nazwy plików, atrybuty, informacje o czasie utworzenia i modyfikacji, wskaźnik na pierwszy klaster z danymi;
- region danych - to ten obszar jest podzielony na klastry, oraz zawiera rzeczywiste dane przechowywane na dysku.



System plików FAT występuje w trzech wersjach, różniących się jedynie liczbą bitów wykorzystywanych do adresowania jednostki alokacji (klastra):

FAT₁₂

Najstarsza z nich, FAT12, w praktyce już nie jest używana. Była przeznaczona głównie dla dyskietek. W stosowaniu jej na nośnikach o większej pojemności przeszkadzał fakt adresowania jednostki alokacji (najmniejszej jednostki organizacji logicznej dysku) przy pomocy zaledwie 12 bitów, tyle co na obsłużenie jedynie 4096 jednostek alokacji. FAT12 ze względu na to ograniczenie, nigdy nie uzyskał większej popularności.

To nie jedyne sztucznie wprowadzone kiedyś ograniczenie, które potem, wraz z rozwojem komputerów, powodowało wiele problemów z utrzymaniem wstępnej zgodności oraz z obsługą nowego sprzętu. Starsi czytelnicy zapewne pamiętają słynne zdanie Billa Gatesa - 640 KB wystarczy każdemu ... przez które komputery osobiste oparte na architekturze Intel'a przez długi czas nie mogły w sposób bezproblemowy wykorzystywać pamięci o większej pojemności.

FAT₁₆

Wraz z systemem DOS 2.0 pojawiła się nowa wersja FAT16, w którym na pamiętanie jednostki alokacji przeznaczono 16 bitów. Oznaczało to, że system plików mógł opisać tylko 2^{16} , czyli 65536 klastrów. Klastry były rozmiarowo równe fizycznym sektorom dysku twardego (512 bajtów), jednakże ograniczało to pojemność do 32 MB. Większy dysk twarty trzeba było dzielić na partycje. Zatem postanowiono zwiększyć rozmiar jednostek alokacji. Jednakże gdy wielkość dysków doszła do 1 gigabajta, jednostki alokacji osiągnęły rozmiar 32 kilobajtów. Tak duża wielkość spowodowała duże marnotrawstwo przestrzeni dyskowej (plik zawierający 10 bajtów informacji zajmował na dysku 32 kilobajty miejsca). Drugim mankamentem systemu plików FAT było ograniczenie wielkości dysku do 2,1 GB. Wymienione cechy spowodowały konieczność stworzenia następcy – systemu FAT32. Inną cechą systemu FAT16 jest nieroróżnianie wielkości liter w nazwach plików, oraz ograniczenie długości nazwy plików do 12 znaków.

Każdy plik zajmuje na dysku całkowitą liczbę klastrów bez względem na jego wielkość. Przy małych plikach (kilka bajtów) następuje więc znaczna strata miejsca na dysku. Wielkość klastra w systemie FAT16 jest ustalana automatycznie w czasie instalacji systemu i zależy od wielkości partycji, na której jest zakładany. Zależność ta została pokazana w tabeli. Należy zaznaczyć, że komputery z systemem FAT16 pracujące pod kontrolą systemu operacyjnego DOS nie obsługują partycji większych od 2GB.

Wielkość klastra w systemie FAT16:

Wielkość partycji	Liczba sektorów na klaster	Wielkość klastra
-------------------	----------------------------	------------------

0 MB–32 MB	1	512 bajtów
33 MB–64 MB	2	1 KB
65 MB–128 MB	4	2 KB
129 MB–255 MB	8	4 KB
256 MB–511 MB	16	8 KB
512 MB–1 GB	32	16 KB
1 GB–2 GB	64	32 KB

FAT₃₂

W roku 1997 rozmiar partycji FAT16 stał się zbyt mały. Klaster nie mógł być większy niż 32KB, co dawało partycje o pojemności do 2GB. Microsoft wprowadził nową wersję FAT32, zawierającą teoretycznie 32-bitowe numery klastrów. W rzeczywistości FAT32 wykorzystuje jedynie 28-bitów dla numeracji. Oznacza to możliwość adresowania do 268.435.438 klastrów, co teoretycznie pozwala na partycje o rozmiarze wielu TeraBajtów. W praktyce, ze względu na ograniczenia programów narzędziowych firmy Microsoft, FAT32 wprowadza ograniczenie do 4.177.920 klastrów, co dawało partycje o rozmiarze ok. 124.55 GB

Wielkość klastra w systemie FAT32 pokazano w tabeli poniżej

Wielkość partycji	Wielkość klastra
Poniżej 8 GB	4 KB
Od 8 GB do 16 GB	8 KB
Od 16 GB do 32 GB	16 KB
od 32 GB	32 KB

VFAT

FAT miał jeszcze jedno dokuczliwe ograniczenie. Nazwa pliku w tym systemie nie mogła mieć więcej niż 8 znaków plus 3 znaki rozszerzenia i mogła składać się tylko ze znaków ASCI o kodach niższych niż 127 (co wykluczało stosowanie polskich liter).

W Windows 95/98 oraz Windows NT/2000/XP wprowadzono obsługę długich nazw plików, przechowywanych w specjalnych (rozszerzonych) wpisach w katalogu. Tak więc, jeśli mamy dysk z systemem plików FAT i nakładką VFAT, informacje o każdym pliku przechowywane są w dwóch niezależnych położeniach, co może powodować problemy przy dostępie do tak sformatowanego dysku z poziomu systemu operacyjnego nie obsługującego VFAT.

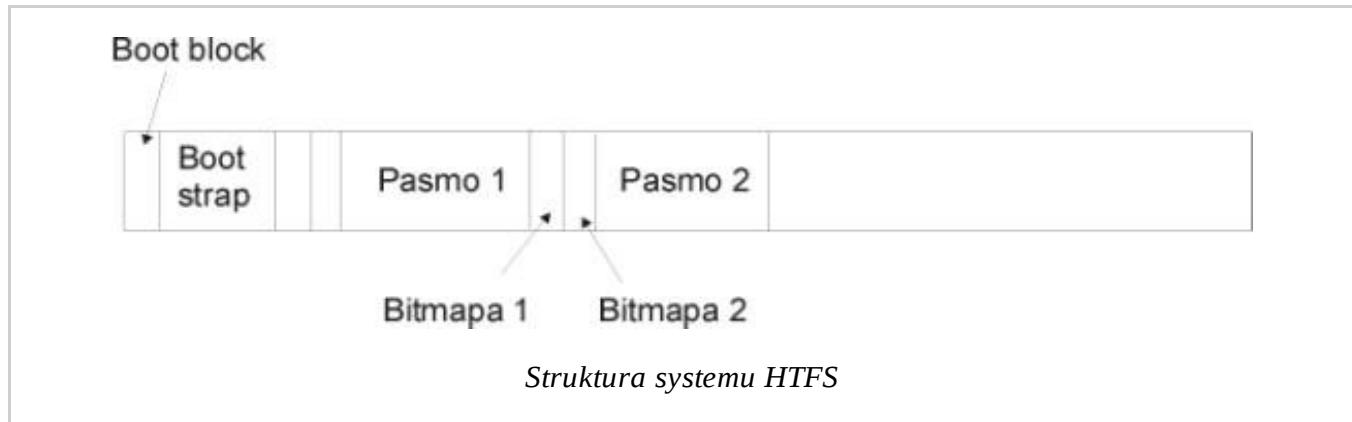
FAT - PODSUMOWANIE

Dziś można by powiedzieć, że FAT to już historia, w dodatku żaden z powszechnie stosowanych systemów operacyjnych nie wykorzystuje FAT jako swojego natywnego systemu plików. Można by -

gdyby FAT nie stał się standardowym systemem plików dla aparatów fotograficznych, pen-drive i tym podobnych urządzeń. Dlatego też warto przeczytać powyższy opis nie tylko ze względów historycznych.

Natomiast nie warto zupełnie stosować FAT (nawet FAT32) jako podstawowego systemu plików na dyskach twardych. Sam system plików, nawet w najnowszej wersji, nie zawiera informacji o uprawnieniach do danego pliku, co powoduje, że wszystkie pliki na dysku są dostępne dla każdego. Tak więc - nie da się zabezpieczyć systemu z dyskami FAT przed nieautoryzowanym dostępem. Przyjęta metodologia zapisu danych jest szybka - ale nieefektywna w sensie wykorzystania przestrzeni dyskowej, co więcej - ciągle wolniejsza niż zaawansowane systemy plików takie jak ReiserFS. Ostatnią niemiłą cechą FAT jest jego podatność na uszkodzenia.

HPFS



System HPFS (ang. *High Performance File System*) jest wspólnym dziełem firm IBM i Microsoft. Na szeroką skalę został zastosowany w systemie operacyjnym OS/2. System HPFS zapisuje na dysku boot block, super block i spare block. Struktury te są wykorzystywane do startu systemu oraz zarządzania systemem plików. Ponadto system rezerwuje obszar tzw. bitmap (2KB), które są rozmieszczone w odstępach co 8MB. Każda bitmapa zawiera jeden bit dla każdej jednostki alokacji w paśmie. System operacyjny zapisuje odpowiednie bitmapy i zaznacza, które sektory są wolne, a które zajęte. HPFS planuje rozmieszczenie plików na dysku w ten sposób, aby jego ewentualne powiększenie nie powodowało defragmentacji dysku. Dodatkowo nazwy plików i katalogów są przechowywane w postaci b-drzewa (b-tree).

Więcej o podobnych metodach zapisywania informacji o plikach powiemy przy omawianiu Unixowych systemów plików. Natomiast, z punktu widzenia użytkownika, zastosowanie b-drzewa jest o tyle istotne, że powoduje zwiększenie wydajności poszukiwania plików na dysku.

W porównaniu do FAT, system HPFS nie tylko cechuje się większą wydajnością poszukiwania plików, lecz również zdecydowanie wyższym poziomem bezpieczeństwa. W tym przypadku mamy do czynienia z pamiętaniem uprawnień do każdego z plików i katalogów, co umożliwia bezpieczne skonfigurowanie systemu operacyjnego. Bezpieczne zarówno w sensie odporności na włamania, jak i zawarcie w nim zabezpieczeń przed kasowaniem plików systemowych przez nieuprawnionych do tego użytkowników.

HPFS jest natywnym systemem plików dla systemu OS/2 firmy IBM. Co istotniejsze - jest również bezpośrednim protoplastą systemu NTFS - aktualnego standardu w Windows.

NTFS

NTFS (ang. *New Technology File System*, w wolnym tłumaczeniu "system plików nowej generacji") jest to standardowy system plików systemu Microsoft Windows NT i jego następców (Windows 2000, Windows XP, Server 2003 i Vista).

NTFS pochodzi od HPFS, zawiera wszystkie jego cechy oraz dodatkowo listy kontroli dostępu (ACL) i dziennik operacji dyskowych (*journal*).

NTFS nie posiada ograniczenia systemu plików FAT dotyczącego maksymalnego rozmiaru pojedynczego pliku, co umożliwia na przykład przechowanie obrazu płyty DVD na dysku twardym, bez dzielenia go na mniejsze pliki. Maksymalny rozmiar pliku dla NTFS to teoretycznie: 16 EB, w praktyce 16 TB (co i tak jest wielkością aż niewyobrażalną). Maksymalny rozmiar pojedyńczej partycji jest też ogromny - wynosi 256 TB.

Jego podstawowe cechy to:

- wprowadzenie mechanizmu księgowania (ang. *journal*). Jest to dziennik zmian, dostępny od wersji NTFS 3.0 w Windows 2000. W skrócie, mechanizm ten pozwala na bezpieczniejsze zapisywanie danych oraz utrzymywanie ich spójności na dysku, nawet w przypadku awarii zasilania w trakcie zapisu. Więcej o tym mechanizmie powiemy przy opisywaniu systemów plików stosowanych w Unix;
- szyfrowanie plików i katalogów (od NTFS 3.0 w Windows 2000) przy pomocy nakładek tworzących EFS - *Encrypting File System*. W zasadzie nie jest to cecha samego systemu plików, tylko szyfrowanie przez niezależny, zewnętrzny program. Dlatego też nie jest możliwe zaszyfrowanie plików systemowych. Ponadto, szyfrowanie nie jest dostępne w każdej wersji systemu operacyjnego. Dla Windows XP i następnych z szyfrowania można korzystać tylko w wersjach Professional lub wyższych (i ich odpowiednikach, np. Vista Business). Dodatkowo, występują problemy ze zgodnością systemów szyfrowania pomiędzy poszczególnymi wersjami Windows;
- kompresja danych w locie - to kolejny podsystem niezależny od samego NTFS. Niestety, szyfrowanie wyklucza kompresję i odwrotnie;
- prawa dostępu dla grup i użytkowników - dostęp do tej funkcji jest ograniczony w Windows XP Home Edition i późniejszych jej odpowiednikach. Pełne wykorzystanie praw dostępu, wraz z możliwością wykonania inspekcji praw dostępu z zapisem do dziennika, możliwe jest w Windows 2000 (wszystkie wersje dla komputerów PC), Windows XP Professional, Windows Server 2003 i nie-domowych wersjach Windows Vista.

System NTFS również występuje w kilku wersjach, co do których numeracji zawsze są problemy. Według numeru wersji wpisanego w nagłówku wolumenu NTFS, system ten miał dotychczas wersje o następującej numeracji wewnętrznej:

- 1.0
- 1.1
- 1.2 - pierwsza popularna, wykorzystywana przez Windows NT 3.51 oraz NT 4.0
- 3.0 - Windows 2000
- 3.1 - Windows od XP do Vista.

Ponieważ sam NTFS jest systemem zamkniętym (nie są publikowane jego specyfikacje), często przyjmuje się oznaczenie wersji Windows. I tak wobec NTFS używanego w Windows 2000 używa się także określenia NTFS 5, gdzie 5 odpowiada wewnętrznej numeracji wersji Windows (Windows 2000 -> Windows NT 5.0).

Pomimo numerycznej różnicy wersji 3.0 i 3.1, NTFS w obu wersjach jest zgodny w przód i wstecz w kluczowych rzeczach i system Windows 2000 - pomimo wbudowanej obsługi NTFS 3.0, nie 3.1 - nie ma kłopotów z odczytem i zapisem wolumenów NTFS 3.1 utworzonych lub podniesionych przed Windows XP i Windows Server 2003. Dla przykładu, systemowe narzędzie CHKDSK może

bezawaryjnie sprawdzić i w razie potrzeby usunąć błędy z wolumenów NTFS wersji 3.1. Jedną z nielicznych funkcji, których zgodność pomiędzy NTFS 3.0 i 3.1 jest zachwiana, jest EFS. Ogólnie rzecz ujmując - do szyfrowania danych w środowisku domowym lub firmowym lepiej jest używać programów niezależnych producentów. Stosowanie wbudowanych mechanizmów ma sens jedynie w przypadku komputerów o tej samej wersji systemu operacyjnego połączonych w domenę.

NTFS wykorzystuje 64 bity do numerowania klastrów. Wielkość klastra w systemie NTFS nie jest stała i może być definiowana przez użytkownika.

Wielkość partycji	Liczba sektorów na klaster	Wielkość klastra
Do 512 MB	1	512 bajtów
513 MB–1024 MB	2	1 KB
1025 MB–2048 MB	4	2 KB
2049 MB–4096 MB	8	4 KB
4097 MB–8192 MB	16	8 KB
8193 MB–16 384 MB	32	16 KB
16 385 MB–32 768 MB	64	32 KB
>32 768	128	64 KB

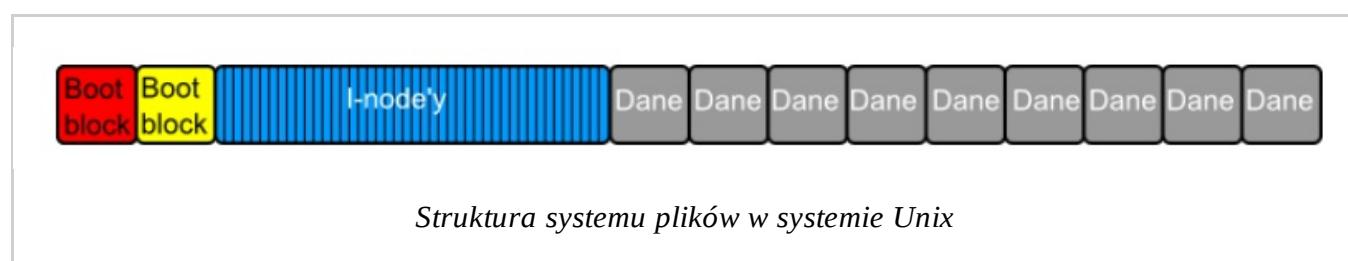
System ten jako standard kodowania znaków stosuje standard Unicode. Pozwala to na używanie w nazwach plików narodowych znaków diakrytycznych, których długość może mieć 255 znaków (w tym spacje i kropki).

Po sformatowaniu nośnika w systemie NTFS tworzone są specjalne pliki, tzw. metadata, a wśród nich Master File Table. MFT zawiera rekordy o wielkości 1KB, w których umieszczana jest informacja o rozmieszczeniu na dysku poszczególnych plików. W zależności od rozmiaru zbioru jest mu przyporządkowana różna liczba rekordów. Pierwszych 17 rekordów MFT ma specjalne znaczenie dla systemu, w nich zapisane są informacje bezpośrednio związane z samym systemem plików. Wszystkie pozostałe zawierają już informacje o plikach i katalogach.

Nowy NTFS może również montować do dowolnego pustego katalogu inny katalog znajdujący się na dysku lokalnym lub zdalnym (mechanizm ten znany jest od dawna m.in. w systemach unixowych.)

EXTENDED FILE SYSTEM - EXT

System plików używany przez różne dystrybucje Linuxa nazwany jest Rozszerzonym Systemem Plików (Extended File System – ext). W zależności od wersji, może to być ext,ext2, ext3 lub ext4. System ten jest rozszerzeniem systemu plików zaimplementowanego w Minixie, który z kolei swoje źródła ma w UFS czyli systemie plików Unixa, którego ogólna struktura przedstawiona została na rysunku:



Na początku woluminu znajduje się boot block, który zawiera program ładowający system operacyjny. Znajdujący się za nim blok przechowuje informację o systemie plików, takie jak liczba bloków, adres początku listy wolnych bloków oraz liczba tzw i-node'ów. Blok ten zwany jest superblokiem (superblock) a informacje w nim zawarte, jako że krytyczne dla działania systemu operacyjnego przechowywane są w kilku miejscach na dysku i mogą być odtworzone w przypadku awarii.

Za superblokiem znajduje się lista i-node'ów, które są strukturami przechowującymi informacje o plikach i katalogach. I-node'y są zbliżone ideowo do rekordów w tablicy MFT (zawierają określenie atrybutów, zezwoleń, ID użytkownika i grupy, rozmiaru, czasy tworzenia, modyfikacji itd. – pole informacja i listę bloków w których umieszczone są dane zapisane w pliku - wskaźniki). Każdy z i-node'ów jest identyfikowany w ramach systemu plików przez swój unikalny numer (nie zawiera jednak informacji o nazwie pliku, który opisuje). W unixowym systemie plików, nazwy plików są wiązane z poszczególnymi i-node'ami dzięki katalogom (katalog jest i-nodem, który przechowuje listę plików w nim zawartych, wiążąc nazwy z numerami i-node'ów).

Organizacja systemu plików bazując na i-node'ach w znacznym stopniu zmniejsza wymagania co do objętości pamięci operacyjnej komputera obsługującego dany wolumin. Z racji swojej specyfiki niepotrzebne jest przechowywanie całej tablicy i-node'ów (która standardowo zajmuje około 1% całkowitej przestrzeni woluminu) w czasie działania systemu, wystarczające jest przeczytanie zawartości i-node'a opisującego plik, na którym ma zostać wykonana jakaś operacja. W związku z tym zajętość pamięci operacyjnej określa iloczyn rozmiaru poszczególnego i-node'a (64 bajty) i liczby otwartych plików oraz katalogów.

Drugą korzyścią z identyfikacji plików poprzez i-node'y jest możliwość stosowania tzw. **twardych odnośników** (hard links), czyli konstrukcji odnoszących się do tego samego i-node'a. Efektem hard link'ów jest np. obecność jednego pliku (tego samego) w dwóch lokalizacjach (folderach) na raz. Ponieważ konkretne dane zawarte w pliku są silniej skojarzone z i-nodem niż z nazwą pliku możliwe jest usunięcie używanego pliku (np. biblioteki) i podmienienie jest nową wersją. Dzięki temu większość modyfikacji i uaktualnień bibliotek systemowych nie wymaga restartu systemu operacyjnego – odnośniki do starych bibliotek są usuwane (nawet jeżeli jakieś programy z nich obecnie korzystają) i na ich miejsce tworzone są nowe. I-node, który jest skojarzony ze starą wersją biblioteki zostanie automatycznie usunięty kiedy wszystkie używające go programy zostaną zamknięte. Nowo uruchamiane programy (od momentu wgrania nowej wersji biblioteki) będą korzystały z uaktualnionych danych.

System plików ext2 został zastąpiony przez ext3, w którym głównym dodatkiem było wprowadzenie dziennika wykonywanych operacji (księgowania). W systemach z jądrem w wersji 2.6.28 dystrybuowany jest jego następca, ext4, który oferuje lepszą wydajność niż ext3 i możliwość tworzenia woluminów o rozmiarze 1 EiB (exabajta, czyli 1024 petabajty).

PODSUMOWANIE

Bez pamięci masowej trudno byłoby sobie wyobrazić pracę systemu komputerowego. Dane w pamięci masowej przechowywane są w postaci plików. Za kontrolę i obsługę plików jest odpowiedzialny system plików. Od niego zależy sposób i jakość wykorzystania nośnika pamięci oraz komfort i sposób pracy użytkownika. Niektóre systemy operacyjne mogą współpracować tylko z jednym systemem plików, inne potrafią wykorzystać wiele z nich. W najpopularniejszych systemach rodziny Windows stosowanych w komputerach klasy PC stosowane są przede wszystkim systemy FAT i NTFS. System NTFS ma znacznie większe możliwości i pozwala m.in. na zastosowanie odpowiednich mechanizmów zabezpieczeń systemu na poziomie pojedynczego pliku i katalogu. Systemy z rodziny Unixów korzystają znów z pochodnych systemu ext - oba rozwiązania są świetne.

Obok wymienionych systemów plików często można spotkać również systemy CDFS i UDF przeznaczone do zapisu i odczytu danych z nośników CD i DVD. Podobnie nie wspomnialiśmy o sieciowych / klastrowych systemach (jak NFS czy HFS) - ale wydaje się, że informacji w tym

podręczniku i tak jest już sporo ;)

SYSTEM OPERACYJNY - ROZWIĄZANIA

W trakcie tej lekcji chcemy Wam przedstawić systemy operacyjne od strony użytkowej. Zapoznamy Was z historią rozwoju systemów operacyjnych, jak również poruszymy zagadnienia ochrony i bezpieczeństwa systemów.

Lekcja ta zawiera krótki i ogólny opis kilku wybranych systemów operacyjnych oraz ich porównanie. Dzięki niej uzyskacie szerszą perspektywę na dostępne na rynku systemy operacyjne oraz ich najważniejsze cechy.

HISTORIA ROZWOJU SYSTEMÓW OPERACYJNYCH

Systemy operacyjne w czasie swego istnienia przeszły znaczące zmiany. Pierwsze z nich były bardzo proste i mało wydajne. Dzisiejsze systemy operacyjne są systemami o dużym stopniu skomplikowania, zajmują nierzadko wiele megabajtów pamięci, a ich wydajność jest niewyobrażalna w porównaniu z możliwościami ich odpowiedników sprzed lat.

Do historycznych już dzisiaj rozwiązań w dziedzinie systemów komputerowych możemy zaliczyć wymienione w [lekcji 5](#) systemy wsadowe czy systemy wieloprogramowe. Ich stosowanie wymagało przede wszystkim wiedzy ... głębokiej wiedzy informatycznej i znajomości budowy komputera.

Reprezentantami tego typu systemów był m.inn. MS DOS. Jednakże wiele osób (w tym piszący te słowa ;)) nie uznaje rozwiązań z tej grupy za pełnoprawne systemy operacyjne - ze względu na fakt, iż zadania zarządzania zasobami sprzętowymi są przez nie realizowane w sposób symboliczny. Z tego powodu nie będziemy się nim dalej zajmowali.

Innym historycznym systemem o zbliżonej do MS-DOS funkcjonalności był CP/M - przeznaczony dla 8-bitowych komputerów.

W latach sześćdziesiątych wśród producentów komputerów pojawił się trend ujednolicania produkowanych przez nich linii sprzętu. Dotąd bowiem jednostki z różnych linii (mainframe, komputery do prostych zastosowań) były wzajemnie niekompatybilne, czyli niemożliwym było uruchomienie na nich tego samego programu bez zmiany jego struktury (w zasadzie napisania go od nowa). Z racji niewygody i dużych kosztów utrzymania tego typu rozwiązania, najpierw firma IBM, a za nią kolejni producenci zdecydowali się na ujednolicenie architektoniczne swoich produktów. W ten sposób powstał IBM System/360, seria komputerów różniących się wydajnością i ceną ale tej samej architekturze. Komputery tej serii obsługiwane były przez system operacyjny OS/360. Najważniejszą cechą tego systemu było wprowadzenie możliwości wykonania (części lub całego) innego programu w czasie, kiedy bieżący program oczekiwany na zakończenie operacji wejścia wyjścia (np. pobrania danych z taśmy) zwane multiprogrammingiem. Oczywiście system operacyjny musiał zarządzać dostępem do komponentów komputera (szczególnie pamięci) aby wiele uruchomionych jednocześnie programów nie nadpisywały wzajemnie swoich danych. Zarządzanie to było wspomagane przez odpowiednie rozwiązania sprzętowe wprowadzone w serii System/360.

Sukces OS/360 i ogromne zainteresowanie multiprogrammingiem zapoczątkował prace nad systemami operacyjnymi umożliwiającymi pełną wieloprocesowość. Zaowocowało to projektem systemu CTSS na uniwersytecie M.I.T, a następnie szeroko zakrojonym projektem MULTICS (1969 r.) realizowanym przez MIT, GE i Bell Labs. System ten był projektowany w celu umożliwienia korzystania z jednego komputera typu mainframe (GE 645) tysiącom użytkowników jednocześnie (w oryginalnym projekcie docelową grupą byli mieszkańcy miasta Boston). Projekt ten nie odniósł sukcesu komercyjnego, jednakże system był wykorzystywany przez takie jednostki jak NSA (Agencja Bezpieczeństwa Narodowego

USA), firmy Ford i GM, Air Force Data Services Center u USA, Industrial Nucleonics oraz Kanadyjski Departament Obrony Narodowej działający do 30 października 2000. System ten posiadał wiele rozwiązań, które zostały wykorzystane w SO następnej generacji i wykorzystywane są do dzisiaj.

Jednakże zarówno OS/360, jak MULTICS nigdy nie zdobyły ogromnej popularności - natomiast twórcy MULTICS postanowili wykorzystać swoje doświadczenia w tworzeniu systemu dla mainframe i opracować system operacyjny dla prostego i taniego (jak na ówczesne czasy) komputera DEC PDP-7 - i tu przechodzimy do pierwszego rozwiązania, które istnieje do dziś, czyli Unix-ów.

RODZINA UNIX

Pierwsza wersja systemu UNIX powstała w Bell Labs firmy AT&T w stanie New Jersey w 1969 na komputery architektury PDP-7 i PDP-9 firmy DEC. Jej głównymi autorami byli Ken Thompson i Dennis Ritchie. Początkowy cel stworzenia nowego systemu operacyjnego został osiągnięty i autorzy rozpoczęli prace nad drugą wersją systemu na komputery PDP-11. Tę wersję uruchomiono w 1971 roku. W tym czasie ukazał się Unix Programmer's Manual, który był de facto opisem ówczesnego Uniksa. Drugie wydanie systemu Unix pojawiło się w 1972 roku, w którym wprowadzono pojęcie łącznika komunikacyjnego łączącego procesy. Minikomputery PDP-11 uzyskały wielką popularność i wraz z nimi także Unix. Dlatego wkrótce napisano trzecią wersję systemu operacyjnego Unix.

Autorzy zrezygnowali z pisania systemu w assemblerze i przenieśli całość kodu do stworzonego przez Dennisa Ritchie wysokopoziomowego języka C, co zaowocowało przenośnością systemu, oraz - paradoksalnie - popularyzacją języka C (dziś najpopularniejszego języka programowania). W 1975 roku pojawił się Unix Szóstego Wydania - inaczej Wersja Szósta - rozpowszechniana nieodpłatnie w uczelniach dla zastosowań akademickich. Przyczyniło się to do gwałtownego rozwoju systemu, powstawania poza AT&T rozszerzeń i oprogramowania. Legendą tego okresu stała się napisana przez profesora Uniwersytetu Nowej Południowej Walii w Australii, Johna Lionsa, tzw. Lions Book zawierająca pełny kod V.6 wraz z komentarzem wersji po wersie.

W 1978 ukazała się wersja 7 (Version 7) - była pierwszą szeroko dostępną, choć AT&T nie zapewniał obsługi i serwisu. Jej zestaw funkcji i programów był, aż do czasu ukazania się specyfikacji IEEE i POSIX, główną bazą unifikującą rodzące się różne odmiany Uniksa. Jednym z ważniejszych wydarzeń we wczesnej historii systemu Unix był projekt agencji DARPA przyjęcia jednego systemu operacyjnego na wszystkich komputerach ARPANET. DARPA zdecydowała, że tym systemem będzie wersja Berkeley systemu Unix. W ramach pierwszego kontraktu, który trwał w latach 1981-1982, powstał system 4BSD oraz 4.1BSD. Sukces zaowocował kolejnym dwuletnim kontraktem, którego owocem był 4.2BSD, będący pierwszym systemem operacyjnym zawierającym obsługę TCP/IP.

Wraz z początkiem lat 80 AT&T zdecydowała się na komercjalizację Uniksa. W 1980 roku licencję na rozwijanie jego kodu sprzedano firmie Santa Cruz Operations, która na zlecenie Microsoftu, miała przygotować Xenix, jego wersję Uniksa. W 1983 roku ukazała się pierwsza wersja komercyjna Uniksa z AT&T, System V.

Wraz z komercjalizacją AT&T przestała udostępniać kod źródłowy systemu poza licencjami komercyjnymi. Spowodowało to protesty wśród wielu inżynierów akademickich (UCB, MIT), którzy do tej pory pisali własne rozszerzenia systemu i uczestniczyli w jego rozwoju (vide: BSD). Blokada nałożona na kod stała się przyczyną powstania na bazie społeczności użytkowników i niezależnych twórców Uniksa ruchu wolnego oprogramowania. Założona w 1983 roku przez Richarda Stallmana z MIT [Free Software Foundation](#) postawiła sobie za cel stworzenie wolnego systemu uniksowego bez kodu pochodzącego z AT&T.

Popularność systemu i mnogość jego odmian wymusiła potrzebę standaryzacji API systemowego, który każdy z systemów deklarowany jako „kompatybilny z Unixem” musiał obsługiwać. Standard ten, wprowadzony przez IEEE nosi oficjalną nazwę IEEE 1003 / oraz ISO/IEC 9945. W świecie komputerów jest on bardziej znany pod nazwą **POSIX**. Ze standardem tym zgodna jest większość wspólnie

występujących systemów operacyjnych wywodzących się z Unix-a, czyli zarówno rozwiązań serwerowych - A/UX, AIX, BSD/OS, HP-UX, Solaris, INTEGRITY, jak i produkty masowe takie jak Mac OS-X firmy Apple, czy systemy niszowe jak IRIX, MINIX, QNX, RTEMS, UnixWare, VxWorks). Częściowo zgodny z POSIXem jest m.in. Linux i FreeBSD, oraz Android i iOS.

Więcej informacji o historii i rozwoju Unix-a możecie znaleźć między innymi pod adresem <http://www.levenez.com/unix>

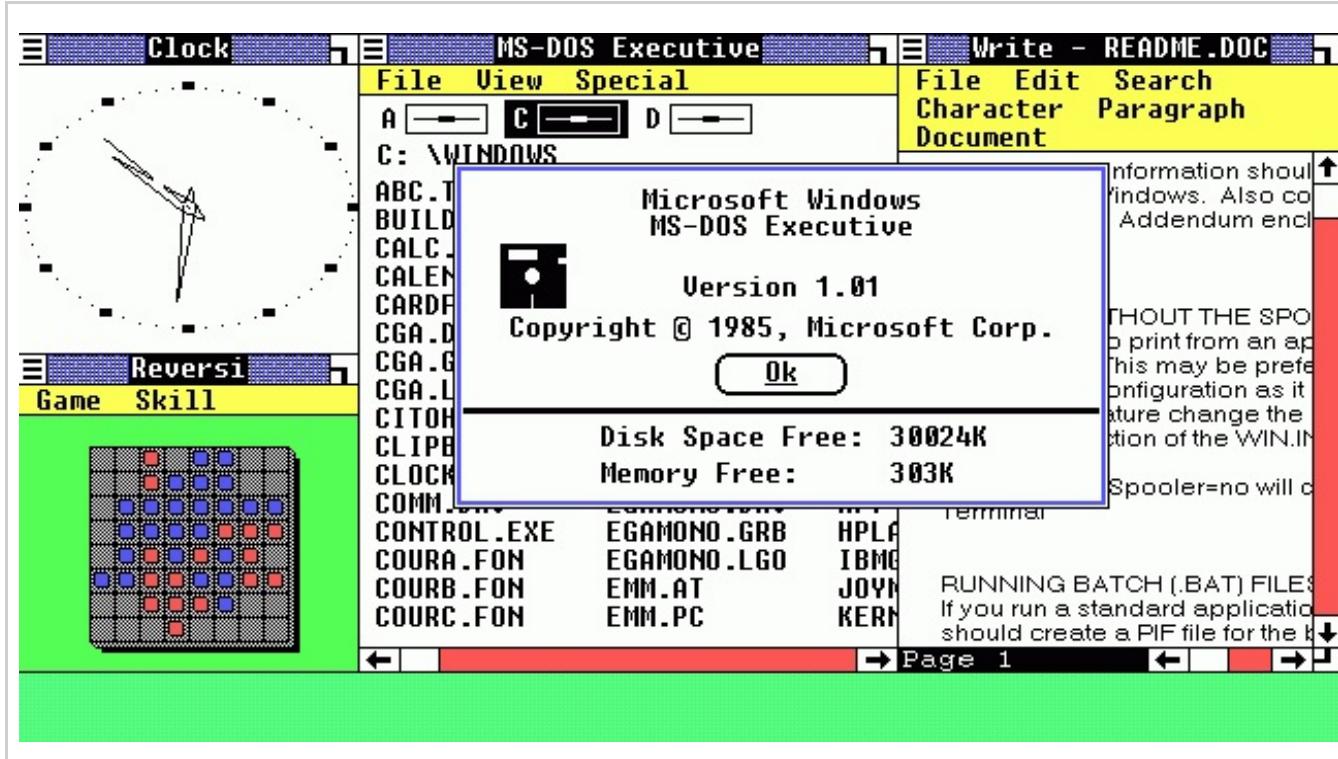
RODZINA WINDOWS

Historia rozwoju systemów Windows dzieli się na dwie odnogi. Punktem startowym dla obu z nich był MS DOS, opracowany i wypuszczony w 1981 roku.

Ze względu na fakt, iż nie był to system operacyjny sensu stricto (co przeszkadzało w jego zastosowaniu w wielu dziedzinach) w zasadzie od samego początku istnienia IBM PC (a szczególnie od pojawienia się modelu PC AT) był poszukiwany dla niego następcą, tym bardziej, że niedługo później pojawiły się rozwiązania systemów operacyjnych dla innych platform (takich jak MacOS firmy Apple, Kickstart+Workbench przeznaczony dla Amiga czy TOS stosowany w Atari ST) bijące na głowę funkcjonalnością i jakością MS DOS.

Firma Microsoft zdecydowała się na poszukiwanie tego rozwiązania dwutorowo - z jednej strony zajęła się rozbudową nakładek na DOS, które potem przekształciły się w tzw. domowe wersje Windows, druga - rozpoczęta we współpracy z IBM miała doprowadzić do zbudowania zupełnie nowego systemu operacyjnego. Ten swoisty dualizm skutkował pojawianiem się na rynku systemów domowych (o znacznie bardziej chwytniej dla oka stylistyce, lecz ograniczonymi w kwestiach pracy w sieci i bezpieczeństwa) oraz wersji profesjonalnych (uboższych graficznie, lecz stabilniejszych podczas pracy w sieci i wyposażonych w lepsze zabezpieczenia).

Pierwsza niezależna wersja Microsoft Windows (opatrzona numerem 1.0), obecna na rynku od listopada 1985, nie oferowała dużego zakresu funkcjonalności i nie zyskała szerszego uznania. Windows 1.0 nie był systemem operacyjnym, lecz przedłużeniem - nakładką na system MS-DOS (jak praktycznie wszystkie wersje domowe tego systemu).



Microsoft Windows w wersji 2.0, sprzedawany od listopada 1987, nieco lepiej przyjął się wśród użytkowników. Wersja 2.03, datowana na styczeń 1988, wprowadziła zmiany w wyglądzie - zamiast okien ułożonych sąsiadującą możliwie było nachodzenie poszczególnych aplikacji na siebie. Skutkiem wprowadzenia tej zmiany był pozew wytoczony Microsoftowi przez firmę Apple Computer, który zarzucał firmie Billa Gatesa naruszenie praw autorskich należących do Apple. Wersja 3.0 nakładki Windows, sprzedawana od roku 1990, była pierwszym produktem rodziny, który osiągnął szerszy sukces. Wyznacznikiem popularności było sprzedanie 2 milionów kopii nakładki w trakcie pierwszych sześciu miesięcy obecności rynkowej. Wprowadzone ulepszenia dotyczyły przede wszystkim interfejsu użytkownika i możliwości pracy wielozadaniowej. W sierpniu 1995 Microsoft rozpoczął sprzedaż systemu operacyjnego Windows 95, który kontynuował zmiany w interfejsie użytkownika i pozwalał na rzeczywistą pracę wielozadaniową w przypadku nowo pisanych, 32 bitowych aplikacji. Kolejnym systemem tej odnogi rodziny była wersja Windows 98, który na rynku pojawił się w czerwcu 1998. Dość ostrej krytyce poddana została jego powolność w porównaniu z Windows 95, lecz większość niedogodności została wyeliminowana w wydanej w roku 1999 wersji Windows 98 Second Edition. Ostatnim przedstawicielem tej gałęzi było Windows ME.

Niezależnie od wprowadzanych fajerwerków graficznych, systemy Windows z serii domowej ciągle w dużej mierze opierały się o jądro MS DOS ze wszystkimi jego ograniczeniami (między innymi z kooperatywną wielozadaniowością bez wywłaszczenia), nie miały możliwości obsługi sensownego systemu plików, nie pozwalały na zabezpieczenie komputera i były po prostu niestabilne. Dlatego też ich rozwój aktualnie zarzucono, wycofano je ze sprzedaży, a rolę systemów domowych przejęły początkowo sztucznie ograniczone wersje serii profesjonalnej, a na dzień dzisiejszy po prostu jeden produkt.

Z drugiej strony, w 1987 roku światło dzienne ujrzała pierwsza wersja wspólnego dzieła IBM i Microsoftu - OS/2 1.0. Był to wtedy system z tekstową powłoką, ale posiadający już bogate API zapewniające kontrolę grafiki i obsługę myszy. Interfejs graficzny wprowadzono wraz z wydaniem w listopadzie roku 1988 wersji 1.1.

W roku 1990 współpraca firm Microsoft i IBM nad OS/2 rozluźniła się. Rosnąca popularność środowiska graficznego Windows opartego o DOS skłoniła Microsoft do skupienia uwagi na własnym systemie. Doprowadziło to do podziału prac nad OS/2 – IBM miał zająć się wydaniem wersji 2.0, podczas gdy Microsoft miał skupić się nad wersją 3.0 znaną jako NT OS/2. Ostatecznie współpraca obu firm nad OS/2 została zerwana, zaś wersję 3.0 Microsoft wydał jako Windows NT.

Wydana samodzielnie przez IBM w roku 1992 wersja 2.0 systemu OS/2 zawierała nowy interfejs graficzny oraz 32-bitowe API. Prawdziwym 32-bitowym systemem był jednak dopiero OS/2 w wersji 3 wydanej w roku 1994. Wprowadził on ponadto lepszą obsługę multimedialnych i internetu. Potem pojawił się system OS/2 Warp, pierwszy system operacyjny na świecie wyposażony w standardzie między innymi w rozpoznawanie mowy. Niestety - IBM wycofał OS/2 ze sprzedaży 23 grudnia 2005 roku, a 31 grudnia 2006 roku zakończył wsparcie dla tego systemu. Według doniesień CNET między firmami IBM i Microsoft doszło do potajemnej transakcji wartości 800 milionów dolarów, która miała na celu "ustąpienie drogi" produktowi Microsoftu. OS/2 będąc znacznie lepszym systemem operacyjnym przegrał w nierównej walce.

W lipcu 1993 pojawił się Windows NT, oparty na nowym jądrze. Seria NT była postrzegana jako system dla profesjonalistów. Następnym przedstawicielem tej serii był system operacyjny Windows 2000. W październiku 2001 Microsoft przedstawił XP. Produkt ten łączył w sobie jądro przejęte z linii Windows NT oraz funkcjonalność systemu Windows 95 i jego następców. W prasie komputerowej pojawiło się sporo pochwał skierowanych pod adresem nowego systemu (pomijając jego szatę graficzną, dzięki której Windows XP był też nazywany Teletubies Windows). Oferowane były dwie wersje Windows XP, oznaczone mianami "Home" i "Professional", przy czym pierwsza - skierowana do użytkowników domowych - nie była wyposażona w szereg mechanizmów sieciowych i

zabezpieczających, które zostały wprowadzone w wersji Professional. Dodatkowym produktem linii XP była wersja "Media Center", która pojawiła się w roku 2003 i skierowana była głównie do entuzjastów filmów telewizyjnych i DVD, pozwalając na przykład na nagrywanie sygnału telewizyjnego i obsługę za pomocą pilota. W 2007 roku pojawiła się wersja Windows Vista, tym razem już w 6 odmianach, lecz wciąż oparta na jądrze NT, potem Windows 7 i Windows 8.

Na tym samym jądrze NT oparte są również specjalne edycje Windows przeznaczone do zastosowań serwerowych. Wprowadził je one wiele modyfikacji w stosunku do wersji przeznaczonej dla stacji roboczych, głównie w dziedzinie bezpieczeństwa i centralnego zarządzania siecią (domeny, active directory, etc...). Aktualnie najnowszym przedstawicielem systemów serwerowych jest Windows Server 2003 R2 (pojawił się w grudniu 2005). Następcą Windows Server 2003 był Windows Server 2008 "Longhorn". Współczesna wersja serwerowa Windows nosi oznaczenie Windows Server 2012.R2 i została wydana w 2013 roku.

PODSUMOWANIE

W dzisiejszych czasach wykorzystywany system operacyjny zależy od zastosowania systemu komputerowego, z którym jest związany. Aby sprostać odmiennym wymaganiom stawianym systemom w każdej grupie zastosowań, rozwinęły się one w kilku różnych kierunkach.

W komputerach osobistych zastosowanie mają głównie wieloprogramowe systemy operacyjne z podziałem czasu (np. Microsoft Windows, MacOS, Linux). W systemach tych główny nacisk nie został położony na maksymalizację wydajności i wykorzystania procesora, a na wygodę użycia i interakcję systemu z użytkownikiem. Systemy te posiadają przyjazny dla człowieka graficzny interfejs.

W zastosowaniach serwerowych (maszyny obliczeniowe, serwery baz danych itp.) wykorzystuje się głównie skomplikowane, wieloprogramowe, równoległe systemy z podziałem czasu lub systemy rozproszone. W systemach tych nacisk położony jest na szybkie i bezawaryjne wykonywanie zadań, do których dany system jest przeznaczony. W tym celu zwykle stosuje się w jednym systemie wiele procesorów oraz dubluje się np. zasoby dyskowe. Interfejs użytkownika w takich systemach ma mniejsze znaczenie. Jądro tych systemów oparte jest głównie na którejś z wersji Unixa.

SYSTEMY OPERACYJNE RODZINY UNIX

Pierwsza wersja systemu UNIX została opracowana w 1969 roku przez zespół pod nadzorem Kena Thomsona. Do czasów dzisiejszych Unix doczekał się wielu kolejnych, coraz bardziej udoskonalonych wersji (zarys historii systemu zamieściliśmy w [pierwszym segmencie](#) tej lekcji). Zazwyczaj stosuje się go komputerach „większych” niż komputery domowego użytku - stanowi on podstawę dużych serwerów (np. IBM z serii profesjonalnych). Unix i jego pochodne jest również praktycznie jedynym wyborem dla komputerów klastrowych.

W zastosowaniach domowo-biurowych oraz jako typowe stacje robocze, Unix w czystej postaci (np. Solaris czy Irix) został współcześnie wyparty albo poprzez MS Windows, albo przez system Linux - swojego młodszego brata.

Unix został zaprojektowany jako system z podziałem czasu. Cechują go także wielodostępowość i wielozadaniowość. W nowszych wersjach systemu Unix zastosowano stronicowanie na żądanie jako mechanizm wspierający zarządzanie pamięcią i podejmowanie decyzji o przydzielaniu procesora. Rodzaj schedulera, techniki zarządzania pamięcią czy same systemy plików mogą się różnić między wersjami Unix-ów - bo też są ich różne zastosowania (np. jako serwery, systemy czasu rzeczywistego, itp...)

Standardowym interfejsem użytkownika jest interpreter poleceń tzw. powłoka (ang. *shell*). Istnieje kilka odmian shell'a (najpopularniejszy z nich to Bash, ang. *Bourne-Again Shell*), które mogą być wykorzystywane zamiennie, w zależności od aktualnych potrzeb użytkownika. Sposób wydawania poleceń w Unix ma charakter tekstowy, choć pod kontrolą tego systemu mogą działać systemy graficzne (np. X-Windows).

```
jczerski@lucky /home/prac/tomcat/jboss-4.0.2/server/all/conf $ cat jndi.properties
# DO NOT EDIT THIS FILE UNLESS YOU KNOW WHAT YOU ARE DOING
#
java.naming.factory.initial=org.jnp.interfaces.NamingContextFactory
java.naming.factory.url.pkgs=org.jboss.naming:org.jnp.interfaces
jczerski@lucky /home/prac/tomcat/jboss-4.0.2/server/all/conf $ ls -la
total 312
drwxr-xr-x  4 jboss 501  4096 Jul 24  2005 .
drwxr-xr-x 11 jboss 501  4096 Jun 23  2005 ..
-rw-r--r--  1 jboss 501 16405 May  3  2005 jacobr.properties
-rw-r--r--  1 jboss 501  6552 May  3  2005 jboss-minimal.xml
-rw-r--r--  1 jboss 501 23219 May  3  2005 jboss-service.xml
-rw-r--r--  1 jboss 501   194 May  3  2005 jndi.properties
-rw-r--r--  1 jboss 501  9944 May  3  2005 log4j.xml
-rw-r--r--  1 jboss 501  7353 Jul 24  2005 login-config.xml
drwxr-xr-x  2 jboss 501  4096 Jul 24  2005 props
-rw-r--r--  1 jboss 501   550 May  3  2005 server.policy
-rw-r--r--  1 jboss 501 35767 May  3  2005 standardjaws.xml
-rw-r--r--  1 jboss 501 59919 May  3  2005 standardjboss.xml
-rw-r--r--  1 jboss 501 112238 May  3  2005 standardjbosscmp-jdbc.xml
drwxr-xr-x  2 jboss 501  4096 Jun 23  2005 xmddesc
jczerski@lucky /home/prac/tomcat/jboss-4.0.2/server/all/conf $ cat server.policy
/// ===== //
//                                //
// JBoss Security Policy          //
//                                //
/// ===== //
//
// $Id: server.policy,v 1.4 2003/08/27 04:31:53 patriotiburke Exp $
//
grant {
    // Allow everything for now
    permission java.security.AllPermission;
};

jczerski@lucky /home/prac/tomcat/jboss-4.0.2/server/all/conf $
```

Przykładowy wygląd ekranu shell'a w czasie pracy w systemie typu Unix.

ARCHITEKTURA SYSTEMU

System UNIX zbudowano trzymając się kilku ważnych zasad. Główną zasadą jest tak zwana reguła KISS (ang. *Keep It Simple, Stupid* – tłum: „to ma być proste, głupku”). Interpretacja tej reguły przy tworzeniu systemu Unix zaowocowała prostotą i przejrzystością architektury i budowy systemu. Trzymano się zasad, że każdy program ma wykonywać jedną i tylko jedną rzecz, do której został przeznaczony i ma ją wykonywać dobrze.

Do wykonywania bardziej skomplikowanych operacji można programy łączyć w tzw potoki - wtedy to wyjście z jednego z nich staje się wejściem do drugiego, ponadto wykonają się automatycznie jeden po drugim. Przykładowo:

```
ls -la | grep mop
```

Wyświetli nam wszystkie nazwy plików i podkatalogów, które zawierają następujące po sobie litery **mop**. Do wykonania tego zadania wykorzystaliśmy dwa programy - **ls** zwraca nam listę plików i podkatalogów, przełącznik **-la** oznacza że polecenie **ls** ma pokazać detale plików. Potem napotykamy pionową kreskę **-** która jest jednym z symboli łączenia w potok i oznacza mniej więcej: weź wynik i przekaż następnemu poleceniu jako wejście. Następne polecenie to **grep** - filtr wyrażeń regularnych, przy wywołaniu jak powyżej po prostu "przepuści" (wyświetli) na ekranie tylko linie zawierające litery **mop**. Jeśli chcielibyśmy wyszukiwać linie z literami **mop** nie w katalogach, lecz w pliku dajmy na to 'eleonora.txt', wywołanie zmieniłoby się tylko w pierwszej części - dotyczącej dostarczenia danych do wyszukiwania (zamiast **ls** w tym wypadku wykorzystamy plik):

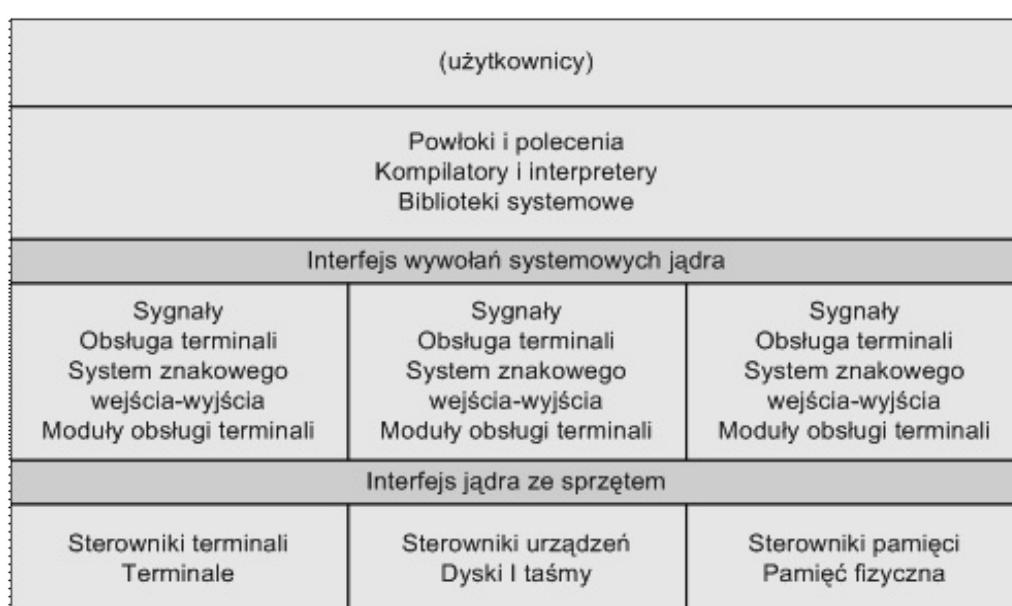
```
cat eleonora.txt | grep mop
```

Stosując przetwarzanie potokowe polecień, w połączeniu z dobrze dobranym zestawem najprostszych programów wykonujących podstawowe operacje, można wykonać dowolne, nawet najbardziej skomplikowane czynności. Drugim bardzo ważnym elementem konstrukcji systemu jest reguła „wszystko jest plikiem”. Oznacza ona, że zarówno pliki na dysku, partycje, całe dyski, nośniki taśmowe porty komputera czy urządzenia wejścia-wyjścia takie jak np. mysz są traktowane jako plik i istnieją w strukturze systemu plików. Szczególny związek z obsługą urządzeń zawarte są w jądrze systemu lub osobnych modułach do tego przeznaczonych dołączanych dynamicznie do jądra. Sam system plików jest zbudowany na zasadzie wielopoziomowego drzewa. W doborze większości algorytmów kierowano się prostotą, a nie szybkością lub wyrafinowaniem.

Większość kodu systemu została napisana w języku C, który opracowano specjalnie w tym celu. Dzięki zastosowaniu języka wyższego poziomu (wcześniej stosowano Asemblera), łatwiejsze stało się przenoszenie oprogramowania Unix z jednego systemu sprzętowego na inne.

Opracowywane systemy Unix od samego początku były dostępne bezpośrednio w postaci kodu źródłowego. Ten schemat postępowania znacznie ułatwił tropienie i usuwanie błędów czy niedociągnięć, jak również był podstawą do zapoczątkowanego w późniejszym czasie ruchu „Open Source” – otwartego oprogramowania z dostępnym kodem źródłowym.

Podobnie jak większość systemów operacyjnych, Unix składa się z dwóch oddzielnych części: jądra systemu i programów systemowych. Na system operacyjny można patrzeć jak na system złożony z warstw:



Warstwowa struktura systemu typu Unix..

Wszystko to, co znajduje się poniżej interfejsu funkcji systemowych i powyżej sprzętu fizycznego, tworzy jądro. W jądrze zrealizowany jest system plików, planowanie przydziału procesora, zarządzanie pamięcią i inne funkcje systemu operacyjnego, do których dostęp odbywa się za pomocą wywołań systemowych. Odwołania do systemu Unix można z grubsza podzielić na trzy kategorie: manipulowanie plikami, sterowanie procesami oraz manipulowanie informacją.

SYSTEM PLIKÓW

Plik w systemie Unix jest ciągiem bajtów. Jądro nie narzuca plikom żadnej struktury. Pliki są zorganizowane w drzewiastą strukturę katalogów. Katalogi też są plikami, które zawierają informacje o tym, jak odnaleźć inne pliki. Nazwą ścieżki do pliku jest napis, który identyfikuje plik przez określenie drogi prowadzącej do tego pliku w strukturze katalogowej.

Pod względem składni nazwę ścieżki tworzą poszczególne nazwy plików pooddzielane ukośnymi kreskami. Na przykład w `/usr/local/font` pierwsza kreska symbolizuje korzeń drzewa katalogów, zwany katalogiem głównym (ang. *root*). W przeciwieństwie do systemów MS Windows, gdzie każdy dysk sformatowany jest w jakimś systemie plików i stanowi oddzielnny korzeń dla struktury plików, w systemie plików Unix'a katalog główny może obejmować wiele dysków czy partycji. Następny element ścieżki `usr` oznacza podkatalog korzenia, `local` oznacza podkatalog katalogu `usr`, a `font` jest nazwą pliku lub katalogu docelowego. Na podstawie składni nazwy ścieżki nie można wywnioskować, czy nazwa `font` odnosi się do pliku zwykłego czy do katalogu.

W systemie Unix używa się zarówno bezwzględnych (rozpoczynających się od korzenia, czyli ukośnej kreski), jak i względnych (rozpoczynających się w katalogu bieżącym) nazw ścieżek. Plik może być znany pod więcej niż jedną nazwą w jednym lub większej liczbie katalogów. Takie zwielokrotnione nazwy są nazywane dowiązaniem (ang. *link*), przy czym wszystkie dowiązania są traktowane przez system operacyjny jednakowo. Nowsze wersje systemu dopuszczają również dowiązania symboliczne (ang. *symbolic link*), które są plikami zawierającymi nazwy ścieżek innych plików. Te dwa rodzaje dowiązań znane są również jako dowiązania twarda i miękkie. W odróżnieniu od dowiązania twardego, dowiązanie miękkie może wskazywać katalog i wykraczać poza granice systemu plików. Występująca w katalogu nazwa „..” jest twardym dowiązaniem do tego samego katalogu, nazwa „...” oznacza twarde dowiązanie do katalogu nadziedzinnego.

BEZPIECZEŃSTWO

Systemy rodziny Unix uznawane są jako systemy z dobrze zaprojektowaną i zaimplementowaną polityką bezpieczeństwa. Zapewnienie bezpieczeństwa danych w systemie wielodostępowym, jakim jest Unix, sprowadza się do rozwiązania dwóch kwestii: uwierzytelniania użytkowników oraz kontroli dostępu do zasobów.

Uwierzytelnianie w systemie Unix na ogół odbywa się za pomocą pliku haseł jawnie udostępnianego do czytania. Hasło użytkownika zostaje zmieszane z wartością losową, a wynik jest kodowany za pomocą jednokierunkowej funkcji transformacji. Użycie funkcji jednokierunkowej oznacza, że wydedukowanie oryginalnego hasła na podstawie pliku haseł jest niemożliwe. Gdy użytkownik przy próbie wejścia do systemu podaje swoje hasło dostępu, jest ono miesiane z tą samą wartością losową również przechowywaną w pliku haseł i poddawane tej samej transformacji jednokierunkowej. Hasło zostaje przyjęte, jeśli wynik tego przetwarzania zgadza się z wartością zapamiętaną w pliku.

Nadzorowanie dostępu w systemach uniksowych odbywa się za pomocą niepowtarzalnych identyfikatorów numerycznych przypisanych do każdego użytkownika (uid – ang. *user identifier*). Identyfikator grupy (gid – ang. *group identifier*) jest dodatkowym identyfikatorem, który można stosować do określania praw przynależnych więcej niż jednemu użytkownikowi. Kontrolowanie dostępu odnosi się do rozmaitych obiektów w systemie. Każdy plik osiągalny w systemie jest chroniony za pomocą standardowego mechanizmu kontrolowania dostępu. Ponadto inne obiekty używane wspólnie, takie jak sekcje pamięci dzielonej i semafory, korzystają z tego samego systemu dostępu.

Z każdym obiektem w systemie Unix, do którego dostęp jest kontrolowany przez użytkownika i grupę, jest skojarzony jeden identyfikator uid i jeden identyfikator gid. Procesy użytkowników też mają pojedyncze identyfikatory uid, ale mogą mieć więcej identyfikatorów gid. Jeżeli identyfikator użytkownika (uid) procesu zgadza się z identyfikatorem użytkownika obiektu, to proces ma prawa użytkownika, czyli właściciela (ang. *owner*) tego obiektu. W przeciwnym razie, jeżeli któryś grupowy identyfikator (gid) procesu pasuje do grupowego identyfikatora obiektu, to nadaje się prawa grupowe. W pozostałym przypadku proces ma do obiektu prawa „reszty świata”, czyli wszystkich użytkowników

nie będących właścicielem obiektu i nienależących do grupy obiektu.

System nadzoruje dostępy przez przypisywanie obiektom masek ochronnych, określających, które tryby dostępu (pisanie, czytanie lub wykonywanie) są udzielane procesem właściciela, grupy lub świata. Właściciel obiektu może mieć pełny dostęp do pliku (tj. prawo czytania, pisania i wykonywania), inni użytkownicy zrzeszeni w pewnej grupie mogą mieć prawo tylko do czytania natomiast pozostały użytkownicy mogą nie mieć w ogóle dostępu do pliku. Jedynym wyjątkiem jest uprzywilejowany identyfikator „super-użytkownika” (root) – reprezentującego administratora systemu. Procesowi z identyfikatorem tego użytkownika automatycznie udziela się dostępu do dowolnego obiektu w systemie z obejściem zwykłej procedury sprawdzania dostępu.

Prawa dostępu związane są z każdym obiektem systemu za pomocą atrybutów. Atrybuty dostępu, uid oraz gid, widoczne są na przykład w momencie wylistowania zawartości bieżącego katalogu za pomocą polecenia:

```
ls -l
```

```
-rw-rw-r-- 1 jczerski prac 747592 Mar 12 20:01 plik1.txt  
-rwxr-x--- 1 jczerski prac 239837 Mar 12 20:01 plik2.txt  
dr-xr-xr-x 3 jczerski prac 4096 Mar 12 20:01 katalog1
```

```
0123456789 x użytkownik grupa rozmiar data_modyfikacji nazwa
```

Na pozycji 0 umieszczony jest znacznik określający typ obiektu. Znak „-” oznacza zwykły plik, „d” oznacza katalog, „l” link symboliczny. Następne 9 pozycji służy do zapisu masek dostępu kolejno dla użytkownika, grupy oraz reszty świata, po 3 znaki każda. Pierwszy znak „r”, jeżeli występuje, oznacza prawo do odczytu obiektu, kolejny „w” - prawo do zapisu a ostatni „x” - prawo do wykonania. Jeżeli zamiast litery występuje znak „-”, oznacza brak danego prawa dla użytkownika grupy lub reszty świata. W następnej kolejności znajdują się: liczba obiektów powiązanych z danym obiektem (1 w przypadku plików, 0 lub więcej w przypadku katalogów), właściciel obiektu, grupa, rozmiar, data oraz nazwa.

X-WINDOWS

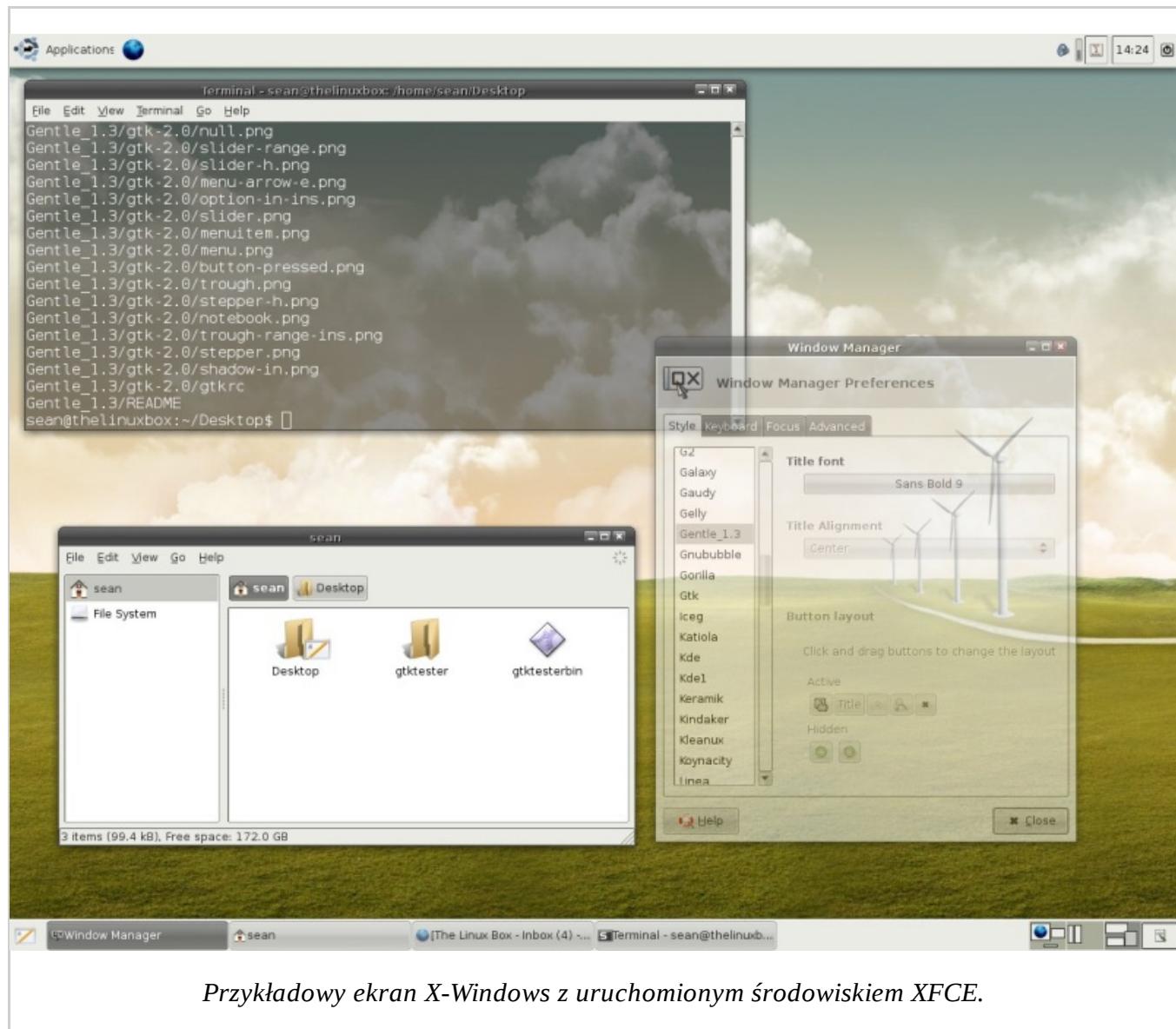
X-Windows jest graficznym interfejsem użytkownika stworzonym w latach 80 i wykorzystywany w systemach z rodziny Unix.

System X tworzy okna, na których program może tworzyć obraz, oraz zajmuje się obsługą urządzeń wejściowych (myszki, klawiatury, tabletu). Serwer X może rysować tylko najprostsze obiekty (odcinki, wielokąty, elipsy, wyświetlać bitmapy, stawiać pojedyncze piksele), nie dostarcza natomiast żadnego interfejsu użytkownika, czyli przycisków, rozwijanych menu, pasków przewijania itp. Rysowaniem i obsługą tych elementów musi zająć się program, najczęściej jest to biblioteka widgetów. System X nie zajmuje się również obsługą okien, nie dostarcza żadnych wbudowanych mechanizmów do ich przesuwania, zmiany rozmiaru, zamknięcia i uruchamiania programów itd., nie rysuje także pasków tytułowych dla okien – tym wszystkim musi zająć się osobny program, tzw. menedżer okien (ang. *window manager*).

Pierwszymi bibliotekami widgetów były Athena, Motif oraz OpenLook, dzięki którym udało się stworzyć proste środowiska graficzne. Dzisiejsze GNOME i KDE (korzystające kolejno z: bibliotek GDK i GTK+ oraz Qt) to całe rozbudowane środowiska graficzne starające się tworzyć pewną użytkową całość.

X-Windows System zaprojektowany jest w architekturze klient-serwer, co oznacza, że składa się z

dwóch elementów: z serwera (zajmującego się wyświetlaniem grafiki oraz obsługą klawiatury i myszy) oraz z klienta (czyli oprogramowania żądającego od serwera zmian zawartości okna). Oba elementy komunikują się ze sobą przy pomocy protokołu przezroczystego sieciowo, co oznacza, że nie ma znaczenia, czy oba elementy systemu działają na tym samym komputerze, czy na dwóch różnych maszynach. Jedna maszyna może wykonywać program, druga może wyświetlać wynik działania programu. Przykładem może być sytuacja, w której użytkownik pracuje zdalnie na odległym komputerze korzystając z X-Windows. Jego lokalny komputer (ten, przy którym siedzi) działa jako serwer X-Windows zapewniając graficzny interfejs użytkownika. Na zdalnym komputerze działa uruchomiony przez użytkownika program, będący klientem serwera X-Windows działającego na maszynie lokalnej. Z serwera na maszynie lokalnej program na maszynie odległej otrzymuje instrukcje sterujące użytkownika i na maszynę lokalną przesyła wyniki swojego działania.



PODSUMOWANIE

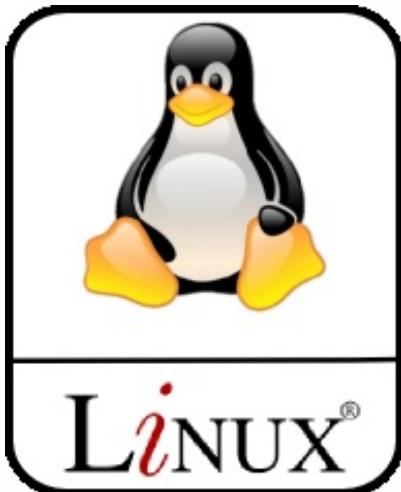
System Unix świetnie nadaje się jako system operacyjny dla różnej wielkości serwerów. Umożliwia on zarówno udostępnienie użytkownikom wielu usług sieciowych (takich, jak dostęp do sieci Internet, współużytkowanie plików, obsługa poczty elektronicznej), jak również zapewnia odpowiednie bezpieczeństwo. Stosując mechanizmy bezpieczeństwa zaimplementowane w Uniksie można zapewnić bezpieczeństwo usług świadczonych w sieci komputerowej, kontrolę dostępu do tych usług, prywatność

zasobów gromadzonych przez użytkowników na serwerze oraz inne elementy wchodzące w skład tak zwanej polityki bezpieczeństwa. Cechy te sprawiają, że system Unix – w różnych swoich odmianach – znalazł zastosowanie w bankach, instytucjach finansowych, agendach rządowych i wielu innych obszarach wymagających niezawodności i bezpieczeństwa.

Przejrzysty projekt systemu Unix zaowocował wieloma naśladowcami i modyfikacjami, z których w ostatnim czasie wyjątkowo aktywna stała się gałąź Linux.

LINUX

POCZĄTEK



Historia Linux-a sięga swoimi korzeniami systemu Unix. Wcześniej pisaliśmy o tym, że istnieje wiele różnych odmian Unixa. Jednym z jego klonów był także Minix - mały system, który choć nie miał wszystkich zalet Unix-a, jednak mógł być z powodzeniem używany na mniejszych maszynach klasy PC. Jego możliwości nie były wielkie - w zasadzie był wykorzystywany jedynie do uczenia studentów budowy systemów operacyjnych na żywym przykładzie z dostępnym kodem źródłowym. W zajęciach z tym systemem uczestniczył niejaki Linus Torvalds - fiński student z uniwersytetu w Helsinkach. Po głębokim (niewątpliwie) namyśle postanowił, że zamiast nieprzydatnego w praktyce, oraz ograniczonego licencyjnie Minix-a można było stworzyć pełnoprawny system operacyjny zgodny z Unixami, ale dystrybuowany w zgodzie z zasadami wolnego oprogramowania.

Rozpoczął prace nad takim systemem. Pierwsza wersja Linux-a (0.01)

mogła obsługiwać dyski oraz mały system plików. Nie miała jednak jeszcze nawet zaimplementowanej obsługi napędu dyskietek. Krótko po pojawienniu się wersji 0.02, 5 października 1991 roku, Torvalds ogłosił istnienie systemu wraz z zaproszeniem do współpracy wszystkich chętnych. Wersja 0.02 posiadała już możliwość uruchomienia bash'a (Bourne Again Shell - popularna powłoka tekstowa) a także gcc (darmowa, genialna kolekcja kompilatorów, głównie wykorzystywana jako kompilator języka C i C++). Niestety poza tym prawie nic nie działało.

Lecz na grupie comp.os.minix pojawiły się pierwsze listy dotyczące Linux-a. Przynęta zadziałała - nowe jądro ściągnęło 10 osób, z czego pięć dokonało własnych poprawek. Linux zaczął się błyskawicznie rozwijać. Zaraz po tym jak wyszła wersja 0.03, pojawiła się 0.10, a w marcu 1992 roku - 0.95.

Pod koniec roku 1991 z systemu korzystało 100 osób, a samo źródło liczyło 10 tys. linii kodu! W I kwartale 1992 roku liczba użytkowników powiększyła się dziesięciokrotnie, a objętość kodu - ponad trzykrotnie. W grudniu roku 1993 pojawiła się wersja 0.99pl14 licząca 100 tys. linii i 20 tys. użytkowników, rok później pojawiło się pierwsze stabilne jądro oznaczone 1.0 oraz pierwsze firmy zajmujące się dystrybucją systemu - Red-Hat i Caldera.

Sama nazwa Linux nie jest dziełem Linusa choć pochodzi od jego imienia. Torvalds proponował nazwę FREAX. Jak twierdził zawierała w sobie słowo free (wolny), freak (dziwak) i X jak Unix. Lecz nazwa nie podobała się adminowi pierwszego serwera ftp na którym Linux był umieszczony - więc zaproponował nazwę Linux, zaaprobowaną przez Torvaldsa. Spontaniczny charakter prac oraz pełen dostęp do kodu źródłowego systemu wpłynęło na powstanie kilku wydań - ułatwiających instalację systemu, oraz łączących jądro z innymi użytkowymi programami. Z czasem wyłoniły się trzy główne historyczne odmiany (dystrybucje) systemu Linux: Debian, Red Hat oraz Slackware.

W 1999 roku nastąpiło wielkie zauroczenie systemem i gazety o tematyce komputerowej zaczęły coraz mocniej rozpisywać się na temat Linux-a. Wiele komercyjnych firm odkryło przy tym nowe, doskonałe źródło dochodu, a entuzjastyczne wypowiedzi na temat systemu Linux stały się normą. Od tego roku system stał się znany, rozpoznawalny oraz stał się poważną alternatywą dla rozwiązań komercyjnych. Dynamiczne tempo rozwoju systemu sprawia, iż staje się on z dnia na dzień coraz doskonalszy i bardziej powszechny. Ogromna rzesza programistów z całego świata, pracujących dla przyjemności, chętnie dokłada swoje trzy wartościowe grosze do tego, by system był coraz lepszy i wydajniejszy.

Należy przy tym pamiętać, iż nazwa Linux odnosi się raczej do samego jądra systemu, które jest dostępne na zasadach GNU General Public Licence (lub "copyleft") utworzonej przez Free Software Foundation. Celem tej organizacji, założonej przez Richarda Stellmana, jest promowanie "wolnego"

oprogramowania o wysokiej jakości. Licencja GPL oznacza możliwość bezpłatnego kopiowania, użytkowania i rozpowszechniania binariów, jednak pod warunkiem, iż rozprowadzane są wraz z kodem źródłowym (dotyczy to także wersji komercyjnych Linux-a).

Dystrybucje Linux-a są zbiorem, w którym znajduje się zarówno jądro systemu, jak i potrzebne do pracy oprogramowanie. Różnią się one sposobem przygotowania systemu, mechanizmami łączącymi pakiety w jedną całość, programem instalacyjnym, przeznaczeniem oraz rodzajem i ilością dołączanych programów.

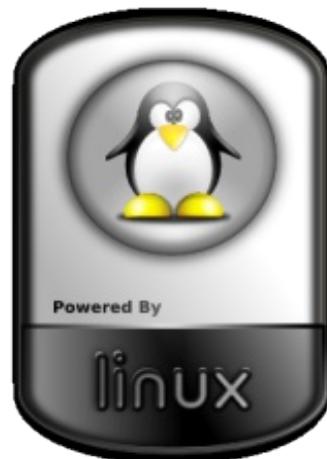
Początkowe idee, w świetle materialistycznej rzeczywistości współczesnego świata, trochę się niestety zacierają i nieuniknione, czysto komercyjne działania wielu firm tworzą nową erę rozwoju Linux-a... jaka ona będzie - zobaczymy. Należy pamiętać, że ducha darmowego oprogramowania tworzonego przez zapaleńców nie jest łatwo zniszczyć - świadczą o tym ciągle powstające w pełni niekomercyjne dystrybucje.

Linux nie jest jedynym przedstawicielem darmowych, otwartych systemów operacyjnych. Istnienie takich rozwiązań jak FreeBSD, BeOS czy GNU Hurd - daje nam wybór ... tak istotny dla zapaleńców (przecież muszą mieć jakiś temat do kłótni odnośnie wyższości świąt Bożego Narodzenia nad Wielkanocą ;)). Niemniej, ze wszystkich darmowych rozwiązań Linux jest najdojrzały i najbardziej "do przyjęcia" dla mniej doświadczonych użytkowników.

DZISIAJ

Dziś Linux ma już za sobą wiek młodzieńczych wad i stał się pełnoprawnym systemem operacyjnym. Można go wykorzystywać z powodzeniem jako zamiennik Windows czy MacOS w codziennej pracy czy w zastosowaniach domowych. Wyposażony w wygodny, okienkowy interfejs użytkownika, graficzne instalatory, olbrzymią ilość dokumentacji w internecie oraz sporą liczbę publikacji książkowych pozwala na w miarę komfortową przesiadkę z innego systemu operacyjnego.

Podstawowy system Linux jest standardowym środowiskiem dla aplikacji i programów użytkownika, które jednak nie narzuca żadnych standardowych sposobów korzystania z dostępnych funkcji, pojmowanych jako całość. W skład tzw. dystrybucji wchodzi aktualnie znacznie więcej, niż to zwyczajowo wymaga się od systemu operacyjnego. W zasadzie każda dystrybucja jest kompletnym zestawem całego potrzebnego oprogramowania, poczynając od jądra, poprzez programy narzędziowe i użytkowe, oprogramowanie biurowe i graficzne, na narzędziach developerskich i serwerach wszystkich usług sieciowych kończąc. Aby zarządzać tym gigantycznym zbiorem programów powstała potrzeba zbudowania jeszcze jednej warstwy usług - tzw. systemu zarządzania pakietami. Dystrybucja Linux-a obejmuje wszystkie standardowe części systemu Linux wraz ze zbiorem narzędzi administracyjnych upraszczających wstępne instalowanie systemu oraz dalsze jego ulepszanie i konfigurowanie, jak również umożliwiających instalowanie i usuwanie dodatkowego oprogramowania. Większość z nich zawiera także własne narzędzia do administrowania systemem plików, tworzenia kont użytkowników i zarządzania nimi, administrowanie siecią itp...



W zastosowaniach domowych jego główną zaletą pozostaje cena - jest za darmo, bądź też w przypadku niektórych dystrybucji - za niewielką opłatą. Większość oprogramowania dla Linux-a jest również darmowa. Emulator Wine (lub jego płatna wersja CrossOffice) pozwala na uruchamianie aplikacji dla Windows, lecz zazwyczaj nie ma takiej potrzeby. Przeciętny użytkownik znajdzie bez problemu darmowe odpowiedniki potrzebnych mu programów, w dodatku znajdzie je i zainstaluje dużo łatwiej niż ma to miejsce w przypadku aplikacji komercyjnych.

Linux jest także systemem bezpiecznym. W każdej dystrybucji wbudowany jest firewall oraz narzędzia szyfrujące, nowoczesne systemy plików z księgowaniem takie jak **ReiserFS** czy **Ext3** są bezpieczniejsze i wydajniejsze nawet od **NTFS**, a wirusy ... cóż, to nie problem :) W praktyce nie ma wirusów grożących temu systemowi, a więc nie ma też potrzeby instalowania oprogramowania antywirusowego.

Szeroko znana stabilność systemu odnosi się głównie do zastosowań serwerowych. Tam nie występuje problem zawieszania się systemu, niedostępności pewnych usług czy też konieczności przestojów serwisowych. Do normy należy (autor sam nadzorował kilka takich serwerów) nieprzerwana praca serwera liczona w latach. W zastosowaniach biurowo-domowych już nie jest aż tak różowo. Zarówno poszczególnym aplikacjom, jak i samym środowiskom graficznym, zdarza się zawiesić - nie częściej niż w przypadku Windows, ale się zdarza. Ponadto funkcjonalność darmowych aplikacji zazwyczaj jest mniejsza, są trudniejsze w obsłudze i nie wyposażone w rozbudowany system pomocy.

Jego olbrzymią zaletą jest fakt, iż każdy z Was, w każdej chwili może zupełnie za darmo sprawdzić, co potrafi ten system. Jest mnóstwo argumentów uzasadniających, dlaczego warto się z nim zapoznać. Linux ciągle szybko się rozwija i coraz więcej firm z niego korzysta. W przyszłości będzie więc zapewne łatwiej pozyskać pracę znając więcej, niż jeden system operacyjny. Warto także chociaż po to, aby kształtować i doskonalić swoją wiedzę informatyczną. W Linux-ie zrozumiesz na jakiej podstawie funkcjonuje sprzęt i jak ogólnie działa każdy (nie tylko Linux) system operacyjny. Nie do pominięcia jest również satysfakcja związana z użytkowaniem systemu bardziej zaawansowanego i trudniejszego w obsłudze niż standardowy Windows.

Jednak nie należy zapominać, że jest to system mniej dopracowany, ustandaryzowany i wygładzony niż jego komercyjni konkurenci - Windows i MacOS. Wymaga wiedzy, wymaga od użytkownika świadomości własnych czynów, oraz nie jest pozbawiony problemów z obsługą nowego sprzętu. Mimo pojawienia się graficznych konfiguratorów, ciągle czasami szczegółowe dostrojenie różnych usług i serwerów wymaga ręcznej edycji plików konfiguracyjnych. Tak więc - nie posiadając odpowiedniej wiedzy informatycznej, zapału i czasu można szybko zniechęcić się i już nigdy nie siegnąć po ten system. Ale moim zdaniem warto spróbować- choćby po to by wiedzieć, że Wam nie odpowiada.

JĄDRO LINUX

Jądro Linux (ang. *Linux kernel*) to najważniejsza i centralna część systemu operacyjnego Linux. Kod jądra Linux-a jest objęty licencją GNU General Public License - każdy może go pobrać z sieci, przejrzeć i modyfikować do woli (pomijając zastrzeżenie, że trzeba być naprawdę dobrym programistą by je zrozumieć).

Jądro Linux-a jest w dużym stopniu zgodne ze standardami ANSI i POSIX, obsługuje wielozadaniowość, wielowątkowość, wielobieżność (wielodostęp), pamięć wirtualną, biblioteki współdzielone, ładowanie na żądanie, współdzielony kod wykonywalny (ang. *copy-on-write*), dobre zarządzanie pamięcią i obsługę sieci TCP/IP. Współcześnie jest ono jądrem hybrydowym, choć wywodzi się z jądra monolitycznego (a więc w większej części ładowanym na stałe do pamięci przy uruchamianiu systemu) z ładowalnymi modułami. Sterowniki urządzeń i rozszerzenia jądra zwykle pracują w trybie uprzywilejowanym, z pełnym dostępem do sprzętu, nieliczne jednak działają w trybie użytkownika. W przeciwieństwie do typowych jąder monolitycznych, sterowniki urządzeń są zwykle komplikowane jako moduły, które można załadować i wyładować na działającym systemie. Podobnie, sterowniki mogą być wywłaszczone w określonych warunkach. Ta funkcja została dodana w celu poprawnej obsługi przerwań sprzętowych i systemów wieloprocesorowych.

Aktualnie większość systemów korzysta z jąder w wersji 3.1 i wyższych (wersje stabilne oznaczane są parzystymi liczbami w drugim i trzecim członie numeru wersji, niestabilne liczbami nieparzystymi).

Zazwyczaj dystrybucja dostarcza nam prekompilowane i skonfigurowane jądro do natychmiastowego zastosowania, niemniej - zawsze można je zamienić na takie, które skompilujemy i skonfigurujemy

samodzielnie. Jest to jedno z pierwszych zadań, których podejmują się ludzie zaczynający swoją przygodę z Linux-em, a przy okazji - jedna z pierwszych porażek ;) Niemniej po lekturze paru artykułów na sieci, oraz serii czasochłonnych prób, każdy z Was, drodzy Studenci, jest w stanie przygotować sobie własny kernel, dostosowany do sprzętu na którym jest uruchomiony, jak garnitur sztyły na miarę.

GRAFICZNE ŚRODOWISKA UŻYTKOWNIKA

W początkowych latach istnienia systemu główną powłoką (interfejsem użytkownika) była tekstowa konsola. Mimo jej wygody (zdecydowanie większej niż konsola Windows) dziś jest to już interfejs przestarzały. Dlatego też i w systemie Linux w zasadzie od samego początku pojawił się [X-Windows](#). Sam X-Windows udostępnia jednak tylko podstawową obsługę okien, odbiegając funkcjonalnością od tego, do czego przyzwyczaił nas MacOS czy Windows. Dlatego w dniu dzisiejszym praktycznie nikt nie używa już X-Windows w czystej postaci, lecz w połączeniu z jednym graficznego środowisk pracy.

Użytkownik przyzwyczajony do MS Windows może nie zdawać sobie sprawy, że Linux oferuje wybór również w tej dziedzinie, nie skazując nas na jedyny słuszny interfejs, przygotowany przez dostawcę systemu. Większość dystrybucji oferuje nam wprawdzie na początek jedno lub więcej wstępnie skonfigurowanych środowisk, lecz żadna nie odbiera nam prawa do instalowania innych, co pozwala na poszukiwanie pulpitu odpowiadającego nam najlepiej.

W Linux-ie sami możemy podjąć decyzję, czy chcemy pracować w środowisku o bardzo rozbudowanej funkcjonalności (jednak wymagającym najczęściej nowoczesnego sprzętu), czy może wystarczy nam prosty tzw. menedżer okien, oferujący tylko niezbędne funkcje, takie jak dostęp do menu aplikacji, zegar i zakładki z uruchomionymi programami. Nieważne czy lubimy bajery i błyski, czy raczej prosty i funkcjonalny desktop, na pewno znajdziemy coś dla siebie. Do tych prostych środowisk, paradoksalnie przeznaczonych raczej dla zaawansowanych użytkowników, możemy zaliczyć:

- **FVWM** - jeden z najstarszych menedżerów okien, uznawany za najbardziej konfigurowalny (oferuje nawet własny język skryptowy). Bardzo ładny temat/nakładkę na FVWM - FVWM-Crystal, rozwija Polak, Maciej Delmanowski;
- **Fluxbox**, **Windowmaker**, **IceWM** - trzy popularne, lekkie menedżery okien, oferujące jedynie najbardziej podstawowe funkcje, jednak dzięki temu pracujące wydajnie również na naprawdę starym sprzęcie;
- **XFCE** - stosunkowo młode środowisko, oferujące mniejszą funkcjonalność niż (opisane dalej) GNOME i KDE, jednak zdecydowanie mniej wymagające pod względem sprzętu. Fani XFCE cenią sobie w tym środowisku prostotę i elegancję.

Jednak zdecydowana większość użytkowników dokonuje wyboru pomiędzy dwoma głównymi konkurentami, GNOME i KDE. Oba stały się już osobnymi i potężnymi projektami programistycznymi, integrującymi międzynarodowych użytkowników. Oba projekty wychodzą z założenia dostarczenia użytkownikowi zintegrowanego systemu zarządzania i pracy w systemie operacyjnym. Oba posiadają pulpity, na których można umieszczać ikony, oba mają paski zadań, oba pozwalają na swobodne przełączanie się między aplikacjami oraz na dokonywanie podstawowych konfiguracji systemu. Oba także obsługują mechanizmy tematów (skórek) - dzięki którym można dostosować ich wygląd do własnych upodobań. Czym więc się różnią?

KDE



[KDE](#) jest najbardziej zaawansowanym technicznie środowiskiem użytkownika dla systemu Linux. W zasadzie stanowi system sam w sobie. Oprócz tego co znacie z Windows (a więc pulpitu, menu start, ikon, itp) KDE daje nam możliwość w zasadzie dowolnego konfigurowania ekranu, wyglądu i położenia poszczególnych elementów, skrótów klawiszowych, położenia przycisków, itp. Nie ma najmniejszego problemu by KDE wyglądało dokładnie tak samo jak Windows z jednej strony, z drugiej - upodobnienie go do MacOS także jest możliwe.

Przy tworzeniu KDE ogromny udział mieli ludzie o tzw. dobrym guście, co oznacza, że w większości przypadków KDE jest po prostu eleganckie. Ponadto istnieją także specjalne strony, takie jak www.kde-look.org, gdzie można znaleźć ogromną ilość dodatkowych tapet, ikon, motywów, efektów, wygaszaczek ekranu, i tak dalej.

Z technicznego punktu widzenia, KDE to nie tylko ładnie wyglądające okienka. To ogromna ilość rozwiązań, które ułatwiają jego używanie, takich jak wyświetlanie tekstu i grafiki na pulpicie, grupowanie okien jednej aplikacji, jednolity system pomocy i konfiguracji, pełna internacjonalizacja, dostęp do wielu pulpitów, i wiele innych (część z tych rozwiązań zostało przejęte przez Windows XP czy Vista, niektóre czekają jeszcze na zaimplementowanie w systemie z Redmond - np w Windows 10 jedną z nowości ma być dostęp do wielu pulpitów ;)). To także zestaw podstawowych aplikacji o świetnej jakości, jak np. zintegrowany system zarządzania informacjami osobistymi (ang. PIM, *Personal Information Manager*) Kontakt - łączący obsługę poczty, kalendarza, listy zadań i książki adresowej, przeglądarka www i manager plików Konqueror, program do nagrywania płyt k3b, centrum obsługi muzyki Amarok, edytor stron www Quanta, środowisko programistyczne KDevelop. Niektóre z aplikacji tworzonych dla KDE jeszcze nie osiągnęły poziomu innych, alternatywnych rozwiązań (np. KOffice jest mniej funkcjonalne niż OpenOffice), ale zawsze można znaleźć coś dla siebie. Jest nawet dostępny klon GaduGadu, o nazwie Kadu. KDE udostępnia także system osadzania "aplikacji w aplikacji" KParts, co oznacza że programiści mogą łatwo w swoich programach wykorzystywać te już istniejące.

GNOME



[GNOME](#) to środowisko nieco skromniejsze w porównaniu z KDE, jeśli chodzi o funkcjonalność, jednak wiele osób preferuje je ze względu na bardziej przejrzysty interfejs i brak skłonności do przerostu formy nad treścią. Korzystając z niego, można stwierdzić, że w zasadzie ma te same funkcje, tylko wszystko w GNOME jest jakby skromniejsze, mniej krzykliwe, czasem może też mniej konfigurowalne, takie bardziej podobne do MacOS niż do Windows.

Liczba aplikacji pisanych specjalnie dla GNOME także jest mniejsza, co nie znaczy, że nie ma tych najważniejszych. Rolę PIM pełni Evolution, przeglądarka www to Galeon. Ponadto w GNOME można uruchomić aplikacje KDE (podobnie jak w KDE aplikacje pisane dla GNOME). GNOME jest także częściej wybierane przez twórców poszczególnych dystrybucji jako podstawowe środowisko graficzne. Mimo tego, można odnieść wrażenie, iż jest to cichszy, bardziej szary brat młodszego KDE, lekko pozostający w jego cieniu.

DYSTRYBUCJE LINUX-A

Teoretycznie system Linux może zainstalować każdy, pobierając najnowsze wersje niezbędnych części systemu z ogólnie dostępnych serwerów ftp, po czym je komplilując. We wczesnych czasach systemu Linux czynność tę często musiał wykonać sam użytkownik. Jednak w miarę dojrzewania systemu poszczególne osoby i grupy dołożyły starań, aby zadanie to uczynić mniej bolesnym, dostarczając standardowego, wcześniej skompilowanego i skonfigurowanego, co więcej - spójnego - zbioru

pakietów do łatwego instalowania.

Początkowo, dopiero co powstające dystrybucje można było porównać do zestawu archiwów. Każdy wchodzący w ich skład pakiet był po prostu archiwum, które odpowiedni program rozpakowywał i kopiował wchodzące w jego skład pliki do odpowiednich, na sztywno przypisanych katalogów. Współczesne dystrybucje mogą w tej dziedzinie znacznie więcej. Mogą dokonać konfiguracji pakietu, mogą automatycznie pobrać z internetu i zainstalować wymagane dla danego programu pakiety z bibliotekami (tzw. kontrola zależności), mogą samodzielnie skompilować kod źródłowy, dostarczają narzędzi wyszukiwania oprogramowania w repozytoriach sieciowych, i tak dalej ...

Pierwszą uznaną za kompletną dystrybucję był zbiór pakietów Linux-a nazwany SLS - ale to już daleka i głęboka historia. Poniżej znajdziecie krótki opis najistotniejszych dystrybucji współcześnie wykorzystywanych. Jest to jedynie wycinek całości - jednak wydaje nam się, że ujeliśmy w nim te najważniejsze.

DEBIAN I POCHODNE

Debian



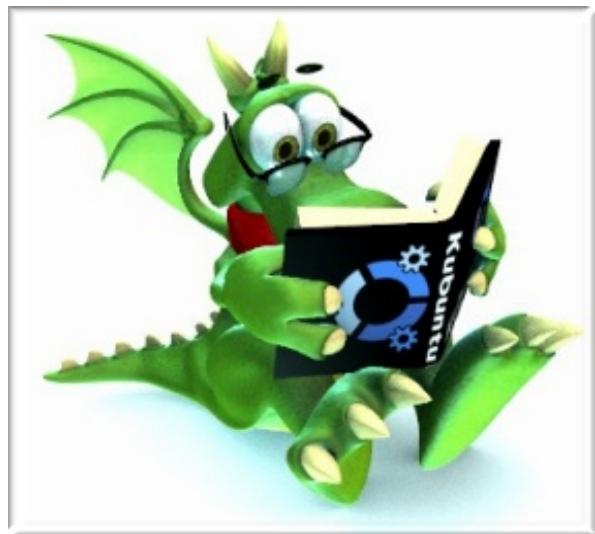
Sam [Debian](#) jest największym działającym do dziś projektem stworzenia wolnej dystrybucji systemu operacyjnego GNU/Linux przez ochotników na całym świecie, bez wsparcia jakiejkolwiek firmy (sic!) - wszystko opiera się na [Umowie Społecznej Debiana](#) oraz wolontariacie jego deweloperów. Wewnątrz Debiana istnieją również projekty mające na celu stworzenie dystrybucji systemu GNU/Hurd, BSD a nawet dystrybucji wolnego oprogramowania na platformę Windows. Debian cieszy się opinią stabilnego systemu o wysokiej jakości i łatwego do aktualizacji. Jest jednak często krytykowany za zbyt rzadkie wydania wersji stabilnych oraz za trudny w obsłudze instalator. Debian korzysta z pakietów DEB. Podstawowym instalatorem pakietów jest dpkg - instalator niskiego poziomu obsługiwany z linii poleceń, lub jego bardziej zaawansowany odpowiednik - APT, w którym wiele czynności jest zautomatyzowanych (pobieranie pakietów, rozwiązywanie zależności między pakietami). Do wygodniejszego zarządzania pakietami, Debian dysponuje nakładkami na powyższe narzędzia - dselect oraz nowszy - aptitude. Pierwsza wersja 1.x pojawiła się w 1996 roku.

Dla początkującego użytkownika Debian na pierwszy rzut oka może wydawać się trudny i odstraszający - ciągle ciąży na nim piętno systemu tworzonego "dla zapaleńców przez zapaleńców". Większość narzędzi systemu ciągle jest obsługiwana z konsoli i coraz częściej przy jego rozwoju góruje podejście zachowawcze, ostrożne przy włączaniu do systemu nowoczesnych rozwiązań. Ponadto czasem uciążliwy jest purytanizm jego twórców odnośnie licencji oprogramowania - np. w głównej dystrybucji nie ma binarnych sterowników do kart graficznych NVidia i ATI (mimo że są one udostępniane za darmo, to ich kod źródłowy nie jest dostępny - vide nie jest to wolne oprogramowanie), nie ma wsparcia dla formatu mp3 (ograniczenia licencyjne), itp. Z drugiej strony Debian jest postrzegany jako system stabilny, o przemyślanej strukturze pakietów i dużej dbałości o spełnienie współprzeciwieństwa między nimi.

Ubuntu



[Ubuntu](#) (polska strona [tutaj](#)) i jego bliźniak - [Kubuntu](#) są (udaną) próbą "ucywilizowania" Debiana, dostosowania go do potrzeb początkującego użytkownika. Obie te dystrybucje są przeznaczone przede wszystkim do zastosowań biurowych i domowych (ang. *desktop*). Bazują na dystrybucji Debian (wersja Sid). Projekt sponsorowany jest przez firmę Marka Shuttlewortha Canonical Ltd. Samo słowo ubuntu pochodzi z języków plemion Zulu i Xhosa zamieszkujących południową Afrykę i oznacza "człowieczeństwo wobec innych" (ang. *humanity towards others*). Głównym celem dystrybucji jest dostarczenie zwykłym użytkownikom kompletnego, otwartego systemu, który będzie łatwy w obsłudze oraz jednocześnie stabilny, niezawodny i nowoczesny. Dodatkową zaletą jest fakt, że z płyta instalacyjnej jest zrobiona w stylu tzw. LiveCD - co oznacza, że można z niej uruchomić system operacyjny i go "wypróbować" (uruchamiać aplikacje, przeglądać, itp) zanim się go zainstaluje. Ubuntu wydawane jest w regularnych, możliwe sześciomiesięcznych odstępach czasowych. Aktualnie możecie spotkać się z dwoma wersjami systemu: Ubuntu 14.04 "Thrusty Tahr" (Triumfalna Turkawka) to wersja o przedłużonym wsparciu, oraz 14.10 " Utopic Unicorn " (numeracja wersji ubuntu pokazuje datę wydania : 7 - to rok, 04 - miesiąc).



Ubuntu i jego pochodne korzystają z repozytorium pakietów Debiana, ale w przeciwieństwie do niego - wyposażono je w wygodne narzędzia instalacyjne oraz zarządzania systemem. Główna różnica między Ubuntu a Kubuntu to wykorzystywane środowisko graficzne: Ubuntu opiera się o Gnome, Kubuntu wykorzystuje KDE. Oprócz nich opracowano kilka innych wariantów, mniej popularnych, takich jak Xubuntu oraz Edubuntu.

Knoppix

[Knoppix](#) to kolejna dystrybucja Linux-a oparta na Debiane (większość oprogramowania pochodzi od edycji Sarge, choć kilka programów pochodzi od edycji Sid), możliwa do uruchomienia bezpośrednio z CD-ROM lub płyty DVD, bez instalacji na twardym dysku (LiveCD). Jest ona rozwijana przez niemieckiego inżyniera Klausza Knoperra. Podczas pracy Knoppix-a tworzony jest wirtualny system plików w pamięci RAM, natomiast wszystkie partycje na dysku twardym montowane są tylko do odczytu. Dlatego nie ma ryzyka przypadkowej utraty danych. Po uruchomieniu można go zainstalować na dysku za pomocą prostego w obsłudze instalatora Knoppix-installer.

Główną zaletą Knoppix-a jest wspomniana możliwość pracy po uruchomieniu z CD. Aktualnie jest wiele tego typu rozwiązań, ale to Knoppix był pierwszym szeroko znany. Często wykorzystuje się go jako przenośne stanowisko pracy - wystarczy nam komputer z napędem CD oraz płytę z Knoppix-em i pamięć USB - by zawsze nasz system był z nami ;)

RED HAT I POCHODNE

Red Hat



[Red Hat](#) – jedna z najstarszych i najpopularniejszych dystrybucji Linux-a, tworzona przez firmę Red Hat. Jej wielka popularność w początkowym okresie wynikała z systemu zarządzania pakietami **rpm** (ang. *RedHat Package Manager*), który został przejęty przez wiele innych dystrybucji. Współcześnie Red Hat zaprzestał dalszego rozwoju Red Hat Linux (ostatnią wersją jest Red Hat Linux 9) tworząc dwie gałęzie - darmową, opartą na ruchu wolontariuszy [Fedora Core](#) oraz komercyjną, przeznaczoną głównie dla serwerów, dystrybucję Red Hat Enterprise Linux.

Dzięki swoim cechom Red Hat Linux był szeroko stosowany w zwykłych systemach odpowiedzialnych

za kierowanie ruchem w sieci, w serwerach wydruków, ppp, poczty, FTP, serwerach baz danych lub stacjach roboczych, a także w najbardziej zaawansowanych zastosowaniach wieloprocesorowych, szczególnie zaś tam, gdzie wymaga się wysokiej niezawodności działania. Red Hat Linux, jako system zapewniający stabilność pracy w sieci, znalazł szerokie zastosowanie na rozwijającym się dynamicznie rynku internetowym. Dzięki swej wieloplatformowości i bezproblemowej obsłudze licznych protokołów sieciowych jest przydatny w integrowaniu zasobów udostępnianych przez systemy operacyjne różnych producentów.

Red Hat Enterprise Linux skierowany jest przede wszystkim do odbiorców komercyjnych takich jak przedsiębiorcy i firmy, dla których oferuje wsparcie i pomoc techniczną oraz aktualizacje za pomocą Red Hat Network. Dystrybucja Red Hat Enterprise Linux powstała w wyniku wprowadzenia w roku 2003 nowej strategii działania firmy Red Hat. Celem nowej polityki było oddzielenie w pełni komercyjnej dystrybucji Linux-a, przeznaczonej dla przedsiębiorców Red Hat Enterprise Linux, od wolnodostępnej, skierowanej przede wszystkim dla użytkowników domowych (pozbawionej komercyjnego wsparcia), Fedora Core.

Fedora Core (niekiedy nieprawidłowo określana jako Fedora Linux) to nazwa następcy wolnej dystrybucji Red Hat Linux rozwijanej przez Projekt Fedora i finansowanej głównie przez Red Hat. Ostatnio utworzono Fundację Fedora, mającą w zamierzeniu koordynować prace nad Fedorą Core w większym stopniu niezależnie od Red Hat-a. Do dziś grono wiernych użytkowników Red Hat-a używa Fedory, zapewne dlatego, że jest to dystrybucja w miarę wypośrodkowana - z jednej strony nie zawiera najnowszych wersji jądra i oprogramowania, z drugiej - jest dość wygodna w użyciu.

Mandriva (dawniej Mandrake)



Mandriva powstała początkowo jako klon Red Hat Linux ładnych parę lat temu, przy założeniu, że ma być zawsze nowsza, łatwiejsza w obsłudze i bardziej "elegancka" niż pierwowzór (nic dziwnego - firma Mandrake została założona we Francji... ;)). Trzeba przyznać, że firmie udało się utrzymać kurs i dziś dystrybucja Linux-a firmy Mandriva jest uznawana za łatwą w użytkowaniu nawet dla niezbyt doświadczonego użytkownika. Ciągle też charakteryzuje się dbałością o graficzny interfejs użytkownika oraz obsługą dużej ilości modeli sprzętu popularnego w zastosowaniach biurkowych. To w niej można znaleźć np. standardowo włączone najnowsze rozszerzenia 3D pulpitu, przy których nawet Vista nie prezentuje się specjalnie oszałamiająco.

Mandriva jest dystrybuowana zarówno jako wersja bezpłatna (free) jak i w postaci płatnych PowerPacków. W przypadku tych drugich płaci się głównie za dodatkowe, dołączone oprogramowanie, które nie jest dostępne jako OpenSource - takie jak np. Cedega pozwalająca na uruchamianie większości gier tworzonych dla Windows, oraz za wsparcie techniczne. Ponadto są dostępne wersje LiveCD (nazywana się Movie) oraz uruchamialna z pamięci USB (Flash) - obie płatne. Dla chcących spróbować - jest za darmo dostępna wersja MandrivaOne, uruchamiająca się z płyty CD.

Aurox



Aurox był tworzony w Polsce dystrybucją systemu GNU/Linux. Przygotowali ją Robert Główczyński i Jarosław Górný. Wywodził się z dystrybucji Red Hat Linux (obecnie Fedora Core) i korzystał z pakietów RPM. Pierwsza Wersja Auroxa ukazała się w listopadzie 2002 roku, regularne wydania zakończono w 2006 roku.

INNE DYSTRYBUCJE

Gentoo



Gentoo - nazwa powstała od pingwina białobrewego, który po angielsku nazywa się *Gentoo Penguin*. Jest to dystrybucja raczej dla zaawansowanych użytkowników, nie polecamy jej jako pierwszego wyboru. Gentoo to w założeniu system najbardziej konfigurowalny i pozostawiający użytkownikowi pełną swobodę decyzji odnośnie każdej jego części. Dzięki systemowi zarządzania pakietami Portage cały system, lub tylko niektóre jego fragmenty mogą być automatycznie optymalizowane dla naszego i tylko naszego komputera - co zwiększa wydajność. O wyjątkowości Gentoo świadczy już sam proces instalacji. Polega on na niskopoziomowej konfiguracji wszystkich elementów systemu. Wybieramy więc podstawowe narzędzia systemowe np. program logujący, cron czy też jądro (w Gentoo mamy ich kilka do wyboru), a następnie instalujemy potrzebne oprogramowanie, głównie poprzez komplikację ze źródeł (choć niekiedy można też skorzystać z dostarczonych pakietów binarnych). Instalacja trwa długo - bywa że i kilka dni (sic!), jednak czas ten nie jest zmarnowany, gdyż Gentoo pozwala nam zbudować system dobrze dopasowany do naszych potrzeb. W zawiłościach niskopoziomowego procesu instalacji pomoże nam doskonały przewodnik użytkownika (również po polsku).

Inność i wyjątkowość Gentoo polega przede wszystkim na sposobie instalacji pakietów. Do tej pory omawiane dystrybucje korzystały z już skompilowanych programów (czyli takich, jakie otrzymujecie też w przypadku Windows - gotowe exe-ki w połączeniu z instalatorem). Gentoo natomiast korzysta z tego, że znakomita większość oprogramowania dla Linuxa to Open Source.

Sercem Gentoo jest Portage - potężne narzędzie, które w założeniu komplikuje każdy instalowany program na naszym komputerze bezpośrednio z kodu źródłowego, uwzględniając np. optymalizacje dokładnie dla naszego procesora. Jeśli porównamy to z klasycznym podejściem z innych dystrybucji - gdzie zazwyczaj otrzymujemy uniwersalne (czyli nieoptymalizowane) wersje programów - to można się domyśleć co będzie szybsze... Portage zapewnia również ciągłą aktualność, nie trzeba czekać na wydanie najnowszej wersji programu dla naszego systemu, ani na kolejne wersje systemu jako takiego. W przypadku Gentoo w zasadzie ciężko mówić o wersji - nie ma takowej, zawsze system jest po prostu najnowszy, o ile dba o to jego administrator.

Gento wymaga jednak posiadania pewnej wiedzy o GNU/Linux-ie, a także czytania (ze zrozumieniem) dokumentacji, przeglądania forum i "grzebania" w plikach konfiguracyjnych (w systemie nie ma tekstowych ani graficznych konfiguratorów). Nie obędzie się również bez poświęcenia zasobów systemu (głównie procesora i pamięci RAM) na częstą komplikację oprogramowania. Przy obecnych komputerach aktualizacja systemu nie przeszkadza jednak w korzystaniu z niego w środowisku graficznym. Gentoo nie jest dystrybucją dla każdego. Wymaga poświęcenia dużej ilości czasu i bycia świadomym tego co robimy i co chcemy uzyskać. Po pewnym czasie odwdzięcza się nam jednak wygodą zarządzania oprogramowaniem i prostą aktualizacją. Dobrze postawione Gentoo działa długo, jest to więc dobry wybór dla cierpliwych, świadomych i (wbrew pozorom) leniwych użytkowników komputerów ;-)

Slackware



Slackware jest najstarszą aktywnie rozwijaną dystrybucją Linux-a, choć warto zauważyć, że Debian jest młodszy zaledwie o 31 dni. W 1992 Peter MacDonald założył firmę Softlanding Linux System, która wydała wspomnianą we wstępie SLS - pierwszą wszechstronną dystrybucję zawierającą takie elementy jak środowisko graficzne X czy protokół TCT/IP. Założony przez Patricka Volkerdinga Slackware oparty został w większej mierze właśnie o SLS. Pierwsza wersja, 1.00, została wydana 17 lipca 1993 roku. Slackware prezentuje odmienne podejście niż inne popularne dystrybucje, takie jak Debian, Ubuntu, Red Hat Linux, Gentoo Linux, SuSE czy Mandriva. Najtrajniej można ją określić mianem "uniksowej", ze względu na politykę dostarczania jedynie stabilnych wersji oprogramowania, organizację skryptów startowych w stylu BSD (ang. *Berkley Software Distribution* - jedna z wersji "czystego" Unixa) i brak dedykowanych narzędzi konfiguracyjnych wyposażonych w GUI, które można znaleźć w innych odmianach Linux-a. Realizacja zarządzania pakietami w Slackware również jest nietypowa ale zgodna z zasadami minimalizmu i prostoty. Pakiety są to zwykłe archiwa tar spakowane programem gzip, zawierające dodatkowe informacje o ich zawartości oraz opcjonalnie skrypt instalacyjny. System ten obsługuje instalację, uaktualnianie i usuwanie pakietów równie prosto, jak w innych dystrybucjach. Nie usiłuje jednak

śledzić czy pilnować tego, co określa się mianem zależności (tj. upewnienia się, że system posiada wszystkie biblioteki i programy, których nowy pakiet potrzebuje do prawidłowego działania).

SuSE



SUSE, prosta w obsłudze dystrybucja Linux-a jest przeznaczona zarówno dla niedoświadczonego użytkownika jak i doświadczonych znawców tego systemu. Oryginalnie SUSE Linux była niemieckim tłumaczeniem dystrybucji Slackware Linux. S.u.S.E została założony w 1992 roku jako grupa konsultingowa, która spośród wielu innych rzeczy regularnie wydawała nowe wersje oprogramowania zarówno dla SLS jak i Slackware oraz pisała instrukcje obsługi dla systemów UNIX/Linux. Pierwszy raz CD zawierający SLS/Slackware został wydany dwa lata później w marcu pod dźwięczną nazwą S.u.S.E Linux 1.0 z jadrem linux 1.0. Z biegiem czasu wprowadzono do systemu kilka rozwiązań firmy Red Hat Linux, np. pakietowanie typowe dla Slackware (tar.gz) zastąpiono pakietami typu RPM.

SuSE oryginalnie była akronimem niemieckich słów *Gesellschaft für Software und Systementwicklung GmbH* (ang. *Company for Software and System Development Ltd*). Od momentu kiedy została przejęta przez firmę Novell, oficjalna nazwa tego systemu brzmi "SUSE Linux". Istnieje jeszcze nieoficjalna plotka, że nazwa została nadana na cześć niemieckiego pioniera komputerowego Konrada Zuse. Rozwijana w Niemczech dystrybucja w styczniu 2004 roku została przejęta przez Novella. Do wersji 7.0 w system włączona była obsługa języka polskiego; została przywrócona dopiero przez Novella. Dystrybucja zawiera graficzny program instalacyjny, narzędzie do konfiguracji systemu i zarządzania pakietami w jednym o nazwie YaST2. Doświadczeni użytkownicy znajdą wiele aplikacji umożliwiających programowanie w różnych językach, w tym Mono (odpowiednik środowiska .net firmy Microsoft). SUSE jest dostępne w dwóch wersjach - do pobrania za darmo oraz do zakupu w wersji box, zawierającej pakiety komercyjne oraz instrukcję obsługi i kilkadziesiąt dni pomocy technicznej przy instalacji. Świeżo zainstalowany system nie wymaga konfiguracji i jest gotowy od razu do pracy – co czyni tę dystrybucję przyjazną nowicjuszom. Dzięki firmie Novell SUSE Linux stał się realną alternatywą dla systemów z rodziną Windows. Na rynek wyszły bowiem różne wersje tego systemu, przeznaczone zarówno dla firm jak i dla komputerowych laików. Wersją dla firm jest SUSE Linux Enterprise Desktop lub jego wersja serwerowa SUSE Linux Enterprise Server. Do zastosowań domowych Novell poleca openSUSE, tak w wersjach pobieranych z Internetu jak i pudełkowych, zawierających na płytach instalacyjnych oprogramowanie o zamkniętym kodzie. Dystrybucja jest bardzo popularna i używana głównie jako system biurkowy. Warto również wspomnieć o bardzo dobrej obsłudze laptopów.

Podsumowując, SUSE jest dobrym narzędziem dla ludzi, którzy chcą po prostu zainstalować system i mieć święty spokój, których nie interesuje zbytnio konfiguracja, strojenie parametrów i częste uaktualnienia, i raczej nie grzebią po sieci w poszukiwaniu ciągle nowych programów.

PLD



Kolejną polską i tworzoną głównie przez Polaków dystrybucją Linuxa jest **PLD**. Jednakże jest ona zupełnie inna niż konkurencyjny Aurox. Po pierwsze - powstała znacznie wcześniej. Po drugie - nie jest klonem jakiekolwiek wcześniejszej dystrybucji, tylko projektem opracowanym od początku. W PLD można znaleźć wiele połączonych rozwiązań pochodzących z różnych dystrybucji, przykładowo do zarządzania pakietami można wykorzystać zarówno RPM (jak w Red Hat Linux, jednak wykorzystanie jego możliwości jest znacznie efektywniejsze), jak i APT (z Debiana) czy też własny system zarządzania pakietami i zależnościami poldek.

W przeciwieństwie do Auroxa, PLD nie jest przeznaczona dla początkujących. Jest trudna, przeznaczona głównie do zastosowań serwerowych, dodatkowo - bardzo bezpieczna. Ogromny nacisk twórcy położyli na zabezpieczenie systemu przed włamaniem oraz na obsługę najnowszych technologii sieciowych (przykładowo - PLD jako pierwsza używa protokołu IP w wersji 6 jako domyślnego). Konfiguracja większości usług i programów jest możliwa jedynie przez edycję plików konfiguracyjnych.

Co wybrać?

Czytając powyższy opis można dojść do wniosku, że ... *osiołkowi w żłoby dano*... Którą dystrybucję wybrać?

Dobrej odpowiedzi na to pytanie nie ma, a w zasadzie jest jedna, nic nie mówiąca - wybierz tą, która Ci najbardziej odpowiada ;)

Możemy jednak udzielić paru wskazówek:

- jeśli chcesz tylko zobaczyć Linux-a w działaniu - skorzystaj z Mandriva One lub Knoppix'a z systemem startowanym z płyty CD. Dzięki temu, bez instalowania czegokolwiek, natychmiast będziesz mógł zobaczyć system w działaniu. Jednak nie licz, że w takiej sytuacji będzie on szybki, lub że będzie obsługiwał wszystkie Twoje urządzenia periferyjne;
- jeśli chcesz spróbować korzystać z Linux-a na dłuższą metę, ale nie wiesz nic o systemach Unixowych - możesz zainstalować Mandrivę, Fedora, SUSE czy Ubuntu na twardym dysku. Każdy z nich da się zainstalować obok Windows i używać z nim na zmianę. Każdy jest wyposażony w sympatyczny, graficzny konfigurator, a proces instalacji nie jest trudniejszy niż instalacja Windows. Który z nich? Na dzień dzisiejszy najpopularniejszy jest Ubuntu i Kubuntu. Zależy Ci na nowinkach - wybierz Mandrivę, chcesz mieć spokój i stabilność działania - wybierz SUSE lub Fedora, chcesz mieć system wolny od jakichkolwiek ograniczeń czy opłat - wybierz Ubuntu (lub Kubuntu dla KDE);
- jeśli jesteś zainteresowany dogłębnym poznaniem działania systemów operacyjnych ogólnie, a w szczególności Linux-a - polecamy Gentoo ... sam proces instalacji nauczy Cię wiele;
- jeśli szukasz stabilnego systemu na serwer - dobrym wyjściem może okazać się zakup którejś z płatnych, serwerowych wersji SUSE, Red Hat czy Mandrivy. Debian lub Gentoo również świetnie się sprawdzą - pod warunkiem odpowiedniej wiedzy administratora.

Powyższe uwagi są jedynie wskazówkami. To, że np. Ubuntu jest oparte na pakietach deb nie oznacza że nie można w nim instalować rpm. Można, tylko trochę więcej trzeba wiedzieć i trochę więcej operacji wykonać by uzyskać ten efekt. Ostateczny wybór i tak będzie należał do Was - i to właśnie jest zgodne z duchem Linux-a :)

RODZINA SYSTEMÓW OPERACYJNYCH MS WINDOWS

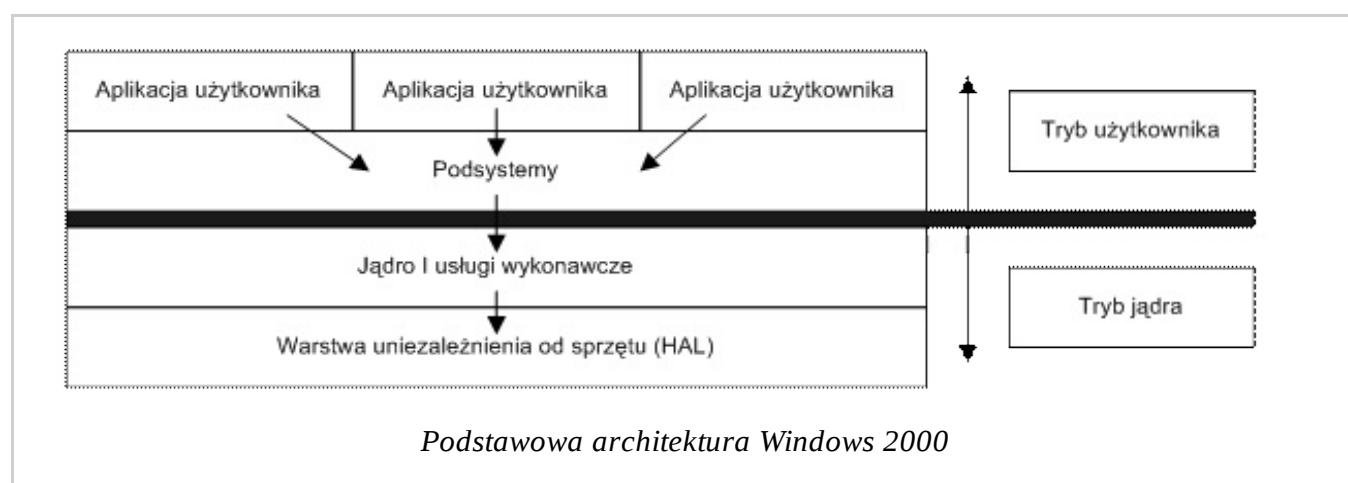
WINDOWS 2000

W lutym 2000 roku, Microsoft wprowadził na rynek system Windows 2000, który rozwijany był pod nazwą Windows NT 5.0. Z powodzeniem trafił on zarówno do segmentu serwerowego, jak również na rynek stacji roboczych. Najbardziej znaczącą funkcjonalnością nowych Okienek była usługa Active Directory, która miała zastąpić model domeny Windows Server z systemu NT 4.0, a zbudowana została w oparciu o technologie DNS, LDAP oraz Cerber.



Wiele funkcji zaczerpnięto z systemu Windows 98, jak na przykład ulepszonego menadżera urządzeń, odtwarzacz Windows Media Player czy DirectX, dzięki czemu po raz pierwszy wiele nowych gier można było uruchomić na jądrze NT. System doczekał wersji DirectX 9.0c (obecnie egzystującej w Windows XP SP2).

Windows 2000 to 32-bitowy, nowoczesny, wielozadaniowy z wywłaszczeniem, wielowątkowy system operacyjny z serii NT. Jako że system był opracowany w wersjach dla różnych procesorów, dużą wagę przyłożono do odpowiedniego wydzielenia obsługi sprzędu - tak powstała warstwowa architektura systemu wraz z warstwą uniezależnienia od sprzętu (HAL – *Hardware Abstraction Layer*).



W trybie jądra uruchamiane są usługi wykonawcze, w trybie użytkownika - programy użytkownika. Usługi trybu jądra są chronione przez procesor, a usługi trybu użytkownika - przez system operacyjny. W momencie, gdy ochrona zawodzi, następuje wyjątek trybu jądra, co przedstawia się jako znany niebieski ekran. W Windows 2000 możliwie największa część systemu operacyjnego działa w trybie użytkownika. W ten sposób redukuje się ilość nadzwyczaj krytycznego kodu jądra, usiłując utrzymać nad nim nadzór. Programy trybu użytkownika komunikują się z jądrem poprzez specjalne interfejsy programowe aplikacji (API). System ten został prawie w całości napisany w języku C++. Elementy, których nie napisano w C++, stworzono w języku niskiego poziomu procesora i to właśnie one składają się na warstwę HAL. HAL jest jedynym elementem systemu operacyjnego bezpośrednio kontaktującym się ze sprzętem.

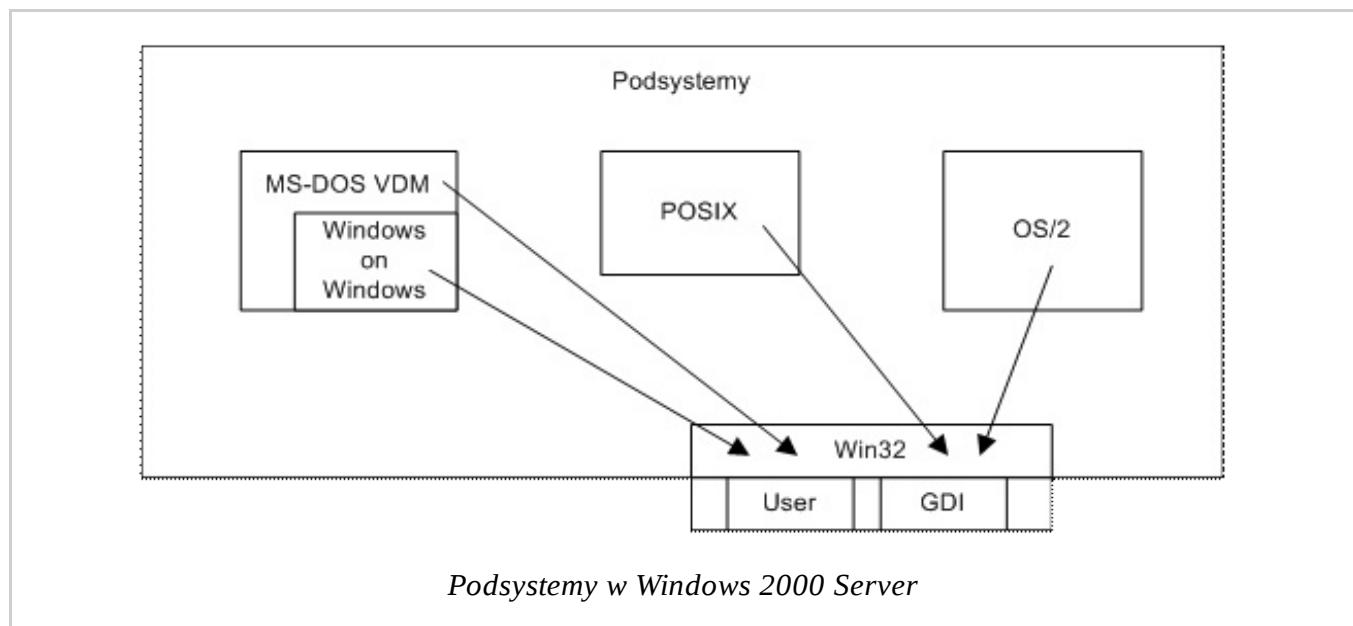
Z punktu widzenia architektury jądro jest centralną częścią systemu operacyjnego. Jądro Windows 2000 wywodzi się z modelu mikrojądra, lecz w praktyce ze względu na wydajność, swoją wewnętrzną budową bardziej przypomina jądro hybrydowe.

System operacyjny musi spełnić wiele różnych zadań, aby zapewnić aplikacjom odpowiednie

środowisko pracy. Model mikrojądra narzuca wykonywanie przez jądro tylko podstawowych działań, takich jak przełączanie zadań. Pozostałe usługi nadal są realizowane, ale już poza obszarem jądra. Jedną z tych wyniesionych poza jądro usług jest egzekutor systemu (Windows 2000 Executive) skupiający w sobie olbrzymi zbiór różnych funkcji przeznaczonych do obsługi odrębnych zadań. Egzekutor pracuje w całości w trybie jądra. Posiada pełny dostęp do jądra i urządzeń wejścia-wyjścia. Większość aplikacji użytkownika nie korzysta z egzekutora bezpośrednio, polegając na podsystemie.

Podsystem może być postrzegany jako środowisko stykające się z jądem i aplikacjami. Umożliwia uruchamianie programów, zapewniając im potrzebne usługi i interfejsy programowe API, równocześnie odwołując się do jądra z prośbą o wymagane zasoby. Podsystem jest właściwie tłumaczem pomiędzy grupą aplikacji a jądem. Występują dwa rodzaje podsystemów. Podsystem Win32 jest klasą sam dla siebie, wszystkie inne podsystemy są właściwie tłumaczami między emulowanymi przez nie podsystemami a Win32.

Win32 jest najczęściej używanym i najistotniejszym podsystemem. Pierwotnie był takim podsystemem jak każdy inny, lecz ze względu na to, że korzystają z niego wszystkie funkcje użytkownika, niektóre jego elementy pracują w trybie jądra, będąc częścią egzekutora. Są to biblioteki User i GDI, sterujące odpowiednio interfejsem użytkownika i monitorem graficznym.



Podsystem Win32 zawiera również podsystemy MS-DOS i 16-bitowego Windows (starsze systemy Windows 3.1 i 3.11), co pozwala na uruchamianie w specjalnej maszynie wirtualnej DOS programów MS-DOS.

Podsystem POSIX planowo miał umożliwiać uruchamianie aplikacji systemu Unix pod kontrolą Windows. POSIX (*Portable Operating System Interface*) jest standardem określającym usługi systemowe i wywołania API innych systemów operacyjnych dostępnych aplikacjom.

Do rodziny systemu Windows 2000 wchodzi 6 różnych produktów:

Windows 2000 Terminal – środowisko udostępniane przez Windows 2000 Server jako usługi terminalowe ze współdzielonymi procesami i pamięcią.

Windows 2000 Professional – prosta wersja systemu wykorzystywana raczej jako stabilna wersja „domowa” (konkurencja dla Windows Millennium Edition). System potrafi obsłużyć maksymalnie 2 procesory i 4 GB RAM.

Windows 2000 Server – podstawowa wersja „serwerowni” z rodziny 2000. System obsługuje maksymalnie 4 procesory i 4 GB RAM, zawiera Active Directory, Intellimirror, Kerberos oraz usługi

terminalowe.

Windows 2000 Advanced Server – wersja analogiczna do wersji Server co do zawartości, potrafi obsłużyć do 8 procesorów i 8 GB RAM.

Windows 2000 Datacenter Server – analogicznie do wersji "Advanced Server", obsługuje maksymalnie 32 procesory i 64 GB RAM.

Windows 2000 Advanced Server, Limited Edition - specjalna wersja systemu wydana w 2001 r. przeznaczona dla procesorów Itanium firmy Intel.

Systemy rodziny Windows 2000 stosowane były często do użytku domowego w wersji Professional, jednak głównym celem ich zastosowania, a dokładniej wersji serwer i wyższych, są małe i średnie wielkości serwery. System ten jest dość prosty w instalacji i obsłudze. Zarządzanie serwerem odbywa się głównie przez graficzne środowisko.

Windows XP

Microsoft Windows XP (nazwa robocza Whistler) to wersja systemu operacyjnego Windows z rodziny Microsoft Windows NT (oparty na jądrze NT, następca Windows 2000) firmy Microsoft, wydana oficjalnie 25 października 2001 roku. Nazwa XP pochodzi od angielskiego słowa *eXPerience*, czyli doświadczenie, doznanie. Ta wersja systemu była przeznaczona głównie dla użytkowników indywidualnych potrzebujących wygodnego i wydajnego systemu operacyjnego dla swojego komputera domowego. Do systemu XP wprowadzono wiele nowych elementów i ulepszeń, jak choćby wbudowana zapora ogniodziałająca chroniąca komputer przed atakami z sieci. Windows XP uważany jest za system stabilny i wygodny w użytkowaniu, i ciągle często można go spotkać na komputerach użytkowników - mimo oficjalnie zakończonego wsparcia przez firmę Microsoft.



Wygląd przykładowego ekranu w systemie MS Windows XP.

Następcą Windows XP była wersja VISTA wypuszczona w styczniu 2007 roku - lecz nie zyskała ona uznania użytkowników, głównie poprzez wolną i niestabilną pracę - dlatego nie będziemy się nim tu zajmować.

WINDOWS 7

Wersja 7 systemu Windows pojawiła się na rynku w 2009 jako następca nieudanego systemu Vista - i na dzień dzisiejszy jest najpopularniejszą wersją tego systemu. System był dystrybuowany w 5 wersjach (jedna w dwóch wariantach):

- Windows 7 Starter – jest to najbardziej podstawowa wersja systemu Windows 7. Preinstalowana na komputerach, które w minimalnym stopniu są w stanie współpracować z najnowszym systemem, głównie na netbookach. Dystrybucja i preinstalacja była prowadzona przez partnerów OEM – system był dostarczany wraz ze sprzętem. Najbardziej widocznym limitem tej wersji jest liczba uruchomionych aplikacji – jednocześnie mogą być uruchomione zaledwie trzy. Wersja ta ma udoskonalony pasek zadań, może także dołączyć do istniejącej grupy domowej (ang. HomeGroup).
- Windows 7 Home Basic – następna wersja systemu Windows 7, która także była dostarczana wraz ze sprzętem przez partnerów OEM. Jest to wersja przeznaczona na rynki rozwijające się. Zawiera dodatkowo Windows Mobility Center, które pozwala m.in. tworzyć połączenia sieciowe typu ad hoc oraz z urządzeniami przenośnymi. Pozwala na udostępnianie połączenia internetowego innym komputerom w sieci (ang. Internet Connection Sharing, ICS) oraz zawiera

bogatszy interfejs graficzny.

- Windows 7 Home Premium – wersja przeznaczona dla użytkowników domowych, zarazem pierwsza, którą można kupić jako osobny produkt (do samodzielnej instalacji). Pozwala nie tylko dołączyć do grupy domowej, ale także tworzyć takie grupy. Zawierała część funkcji multimedialnych ze wszystkich dostępnych w różnych wersjach, m.in. Media Center i bardziej rozbudowaną obsługę plików multimedialnych. Inne funkcje to m.in. interfejs Aero Glass, obsługa technologii MultiTouch oraz rozpoznawanie pisma odręcznego.
- Windows 7 Professional – następna wersja możliwa do kupienia, kierowana do bardziej wymagających użytkowników, ale także małych firm. Wersja ta pozwala dołączyć do domeny, obsługuje szyfrowanie EFS, pliki offline, zaawansowane opcje kopii zapasowej – czyli jedne z ważniejszych funkcji przeznaczonych do pracy w sieci firmowej. Z mniej ważnych można wskazać możliwość podłączenia się przez Remote Desktop do innej stacji roboczej czy Location Aware Printing (szczególnie przydatna w komputerach przenośnych – automatycznie rozpoznaje, do jakiej sieci podłączony jest komputer, np. domowej czy firmowej, i odpowiednio ustawia domyślną drukarkę).
- Windows 7 Enterprise – wersja sprzedawana dużym przedsiębiorstwom (klienci „enterprise”). Do nabycia jedynie poprzez kanał Microsoft Volume Licensing. Obsługuje techniki: DirectAccess (działanie podobne do VPN), BitLocker i BitLocker To Go (pomaga chronić dane na komputerach i modułach pamięci przenośnej), BranchCache (skracza czas oczekiwania na pobranie pliku w sieci), AppLocker (określa możliwe do uruchomienia aplikacje). Wersja ta obsługuje tzw. MUI (ang. Multilingual User Interface), czyli inne wersje językowe systemu, i VDI (ang. Virtual Desktop Infrastructure).
- Windows 7 Ultimate – wersja o możliwościach wersji Enterprise, jednak możliwa do nabycia w kanałach konsumenckich (nie wymaga Volume Licensing).

Z technicznego punktu widzenia Windows 7 nie zmieniał zasadniczo technologii stosowanych już w XP i Vista - podstawową różnicą było to, że system jest stabilny i szybki. Miłe dla oka dodatki, takie jak interfejs Aero, czy ulepszone zarządzanie oknami, przycisk Start, nie zmieniają faktu, iż wersja 7 jest raczej ewolucją niż rewolucją.

RODZINA SYSTEMÓW DLA KOMPUTERÓW APPLE

MACINTOSH MACOS

MacOS to system operacyjny dla komputerów Macintosh firmy Apple. Był to pierwszy ogólnodostępny system operacyjny wyposażony w graficzny interfejs użytkownika (GUI). Wcześniejsze komputery Apple pracowały w oparciu o interfejs tekstowy. Idea graficznego interfejsu została zaczerpnięta przez inżynierów Apple z laboratoriów firmy Xerox w Palo Alto, gdzie w 1974 roku stworzono Alto, pierwszy komputer z graficznym interfejsem.

Pierwsza wersja systemu weszła do użytku publicznego w 1984 roku wraz z modelem Macintosh - reklamowana przy pomocy klipu nawiązującego do literatury Georga Orwella. Upowszechnione przez nią takie standardy graficznego interfejsu jak okna, rozwijalne menu, kursor myszy, kosz na niepotrzebne pliki stały się od tej pory podstawą współczesnych systemów operacyjnych. Zachęcony sukcesem Apple, w tym samym kierunku zaczął podążać Microsoft, jednocześnie Apple starał się opatentować wygląd i wrażenia swojego systemu. Doprowadziło to do długotrwałego procesu o własność intelektualną.

MacOS działał w systemach opartych początkowo na procesorach Motoroli z linii 680x0, a następnie linii PowerPC opracowanej wspólnie przez Apple, Motorola i IBM. W 1983 roku próbnie przeniesiono go także na platformę IA32. Próbne testy wykazały, że działał on szybciej niż na ówczesnych układach wykorzystywanych w komputerach Apple. Kierownictwo firmy wtedy nie zdecydowało się jednak na kontynuowanie tych doświadczeń w obawie o utratę zapotrzebowania na własną platformę sprzętową.

Mimo, że interfejs MacOS był bardzo intuicyjny i wygodny, sam system cierpiał na wiele poważnych błądów, jak np. brak wielozadaniowości z wywłaszczeniem czy w pełni bezpiecznej pamięci. Apple przez wiele lat starało się rozwiązać te kłopoty na bazie kodu MacOS. Starania te nie przyniosły zadowalającego rezultatu. Nigdy nie zostały ukończone projekty Copland oraz Rhapsody, która była próbą połączenia uniksowego jądra z macintoshowym interfejsem użytkownika.

Po powrocie do Apple jednego z założycieli firmy Steve'a Jobsa i wchłonięciu jego firmy NeXT rozpoczęto prace nad nowym systemem - OS X. Prowadzono je w oparciu o doświadczenia z Rhapsody oraz NextStep, rewolucyjnego systemu operacyjnego stworzonego w ramach projektu firmy NeXT utworzenia nowego komputera osobistego. W międzyczasie doskonalono MacOS, aby nie utracić pozycji rynkowej. Jego ostatnia wersja była oznaczona numerem 9.2.2.

MACOS X

W 2000 roku wprowadzono MacOS X (współcześnie OS X) - tj. MacOS numer 10. Mimo kolejnego numeru (po 9 wersji) OS X był zupełnie nowym produktem. Pierwotnie działał tylko i wyłącznie na Mac-ach z procesorem PowerPC. Jak wspomnieliśmy - system OS X jest oparty na systemie NextStep, który z kolei był oparty na systemie Unix. W efekcie OS X jest zgodny ze standardem POSIX i często jest traktowany jako kolejna mutacja Unix-a.



Wygląd przykładowego ekranu w pierwszym systemie systemie MacOS X.

Jądro systemu MacOS X jest oparte na projekcie Mach realizowanym na uniwersytecie Carnegie-Mellon. Jądro Mach opracowywane w ramach tego projektu było mikrojądrzem opracowywanym na potrzeby zastąpienia jądra monolitycznego wersji BSD systemu Unix. W praktyce pierwsza i druga wersja jądra Mach nie były w pełni mikrojądrami - dopiero finalna, 3 wersja była w pełni zgodna z paradygmatem mikrojądra. I właśnie na tych wcześniejszych wersjach oparto jądro XNU będące podstawą systemu OS X. Jest to jądro hybrydowe, oparte na jądrze Mach 2.5, dość nowoczesne i zapewniające wszystkie wymagane przez nowoczesny system funkcje, włącznie z wywłaszczeniem, obsługą pamięci wirtualnej czy zaawansowanym schedulerem.

Firma Apple wraz z wprowadzeniem nowego systemu zdecydowała się zmienić również graficzną powłokę na zupełnie nowe rozwiązanie, nazwane Aqua, która została wyjątkowo dobrze przyjęta przez rynek, głównie ze względu na jej estetyczne walory. Sam Steve Jobs podkreślał, że "*One of the design goals was when you saw it you wanted to lick it*", czyli w wolnym tłumaczeniu - jednym z celów projektu było doprowadzenia użytkownika do chęci polizania interfejsu kiedy go zobaczy. Powłoka Aqua jako pierwsze rozwiązanie masowe na szeroką skalę wykorzystywała efekty graficzne udostępnione przez nowoczesne karty - przeźroczystość, animacje, odbicia i tym podobne, przy czym pozostała wygodna dla użytkowników.

Aktualna wersja systemu nosi numer 10.10 Yosemite.

PODSUMOWANIE

Wybór systemu operacyjnego powinien być ściśle związanego z docelowym przeznaczeniem komputera, na którym ma być zainstalowany i wykorzystywany. Dobór powinien również uwzględniać administratora komputera oraz jego docelowego lub docelowych użytkowników w zakresie znajomości danego systemu, jak również ogólnej wiedzy i umiejętności posługiwania się oprogramowaniem. Inaczej powinien być dobrany system dla komputera domowego, serwera wydruków w małej firmie czy serwera przechowującego dane finansowe w banku. Decydując się na konkretny system operacyjny należy mieć świadomość o jego niezawodności i stabilności, możliwościach tworzenia kopii zapasowej itd.

Do użytku osobistego w znaczającej większości wykorzystuje się systemy rodziny MS Windows 7 oraz 8, oraz OS-X, przy czym ten drugi tylko przy wyborze komputerów Apple. Zdecydowanie najpopularniejszym rozwiązaniem pozostaje system MS Windows. W niektórych zastosowaniach nie ma dla niego sensownej alternatywy (np. gry, czy stacje przeznaczone do uruchamiania specjalizowanego oprogramowania takiego jak CAD). W wielu przypadkach jednak inne rozwiązania sprawdzą się tak samo dobrze albo i lepiej. Typowym przykładem jest tutaj przetwarzanie grafiki czy montaż wideo - tu króluje OS-X, ze względu na wysoką stabilność, wydajność oraz jakość oprogramowania.

Jednak systemy te, ze względu na swoje ograniczenia wydajnościowe i budowę, nie nadają się do zastosowań serwerowych. W przypadku Windows istnieją specjalizowane wersje serwerowe systemu, które są w stanie pracować nieprzerwanie przez miesiące i lata - lecz w tym przypadku na rynku nie uzyskały pozycji dominującej. Częściej jako systemy serwerowe wykorzystuje się różne wersje systemów z rodziny Unix, włączając w to rozwiązania płatne (RedHat, AIX) oraz darmowe (różne dystrybucje Linux-a, FreeBSD czy OpenBSD). Dobra przemyślana i zrealizowana architektura tych systemów, przydziału zasobów i kontroli dostępu pozwala na nieprzerwaną i wydajną pracę bez konieczności restartu.

W ostatnim czasie bardzo szybki i aktywny rozwój gałęzi Linux oraz graficznego interfejsu użytkownika dla tego systemu jak również wszelkiego rodzaju aplikacji z niego korzystającego powoduje wzrost zastosowań tego systemu wśród użytkowników komputerów osobistych i domowych.

W praktyce ciężko jest zbadać rzeczywisty udział poszczególnych systemów operacyjnych na rynku. Pewną wskazówką może być rodzaj systemu wykorzystywanego przez internautów - a tu, wg badań firmy gemius, wychodzi pozostająca dominacja systemów Windows - ogółem ok 85 %. Drugie miejsce zajmuje Android (czyli urządzenia mobilne) - ok. 10%, a bezpośredni konkurenci MS Windows, czyli Linux i MacOS X mają odpowiednio 1.3 % oraz 1.1 %. Tak więc zapowiadany od dawna zmierzch Windows w zastosowaniach do komputerów osobistych raczej nie występuje, a przynajmniej nie widać go w naszym kraju. Na świecie sytuacja wygląda podobnie - dominuje Windows, natomiast MacOS X ma zdecydowanie więcej użytkowników - ok 5 do 9 % (wg różnych źródeł), udział Linux-a jest podobny jak w Polsce i oscyluje w granicach 1 do 2 %

BEZPIECZEŃSTWO W SYSTEMACH INFORMATYCZNYCH

W tej lekcji chcielibyśmy się skupić na zagadnieniach ogólnie rozumianego bezpieczeństwa w systemach informatycznych. Ze względu na fakt, iż niniejszy podręcznik i przedmiot ma stanowić wprowadzenie w zagadnienia informatyki - nie znajdziecie tu jednak kompendium wiedzy o ochronie systemów komputerowych. Raczej postaramy się przedstawić Wam podstawowe pojęcia związane z ochroną i bezpieczeństwem systemów informatycznych, oraz omówić proste metody zapobiegania zagrożeniom. Tak więc - znajdziecie tu raczej przyczułek do dalszych, samodzielnych poszukiwań, niż szczegółowy, prowadzący krok po kroku, kurs pt. "Jak zabezpieczyć mojego PC".

BEZPIECZEŃSTWO DANYCH I SYSTEMÓW

Zagadnienia bezpieczeństwa systemów informatycznych obejmują całe, dość szerokie spektrum tematów. Wychodzimy tu z założenia, że system operacyjny wchodzi w skład większego organizmu jakim jest **system informatyczny**. Pod pojęciem **systemu informatycznego** rozumiemy zespół elementów sprzętowych i programowych służący do gromadzenia, przetwarzania i przesyłania danych.

To, co zazwyczaj chcemy zabezpieczyć - to wspomniane dane. Nie będziemy się tutaj zastanawiali co oznacza określenie "Kenigston lock" oraz jakim łańcuchem i do czego przymocować komputer ... ;)

Problem bezpieczeństwa i ochrony danych (informacji) przechowywanych przez systemy informatyczne nabiera coraz większego znaczenia. Liczba osób korzystających z systemów informatycznych ciągle rośnie, a ostatnie lata przynoszą ciągły, dynamiczny rozwój usług internetowych, usług teleinformatycznych, handlu elektronicznego i bankowości elektronicznej. Łatwo sobie wyobrazić jakie skutki dla funkcjonowania przedsiębiorstwa mogą wiązać się z utratą lub niekontrolowaną zmianą danych elektronicznych. Często też (niestety coraz częściej) słyszycie o udanych atakach na komputery / serwery / sieci komputerowe, o kradzieży danych, kont, czy też pieniędzy. Nie wynika to ze słabości systemów zabezpieczeń (te są coraz lepsze i bardziej wyrafinowane) - lecz ze skali stosowania systemów komputerowych i ilości przeprowadzanych na nie ataków. W zasadzie każdy komputer podłączony do internetu podlega nieustannym atakom różnych typów. Teraz chcieliśmy się skupić na zagrożeniu dla systemów komputerowych

ZAGROŻENIA

Zagrożenia dla danych przechowywanych w systemach teleinformatycznych są bardzo różnorodne. Należą do nich zarówno działania mogące spowodować zniszczenie, uszkodzenie lub zmianę danych w systemie jak również zmierzające do odczytania (skradzenia) ich przez osoby nieuprawnione.

Zbiory danych należące do użytkownika podłączonego do systemu otwartego (a więc udostępnionego w sieci publicznej) mogą zostać zniszczone np. w wyniku działania wirusów lub odczytane przez hakera. Mogą też zostać zniekształcone lub zniszczone w czasie transmisji np. w wyniku awarii lub też celowego działania innych osób. W ich wyniku może nastąpić zarówno utrata danych, przechwycenie przez osoby nieupoważnione jak również naruszenie integralności i spójności systemu (bazy danych), do którego dane te zostały przesłane. Dlatego też specjalisci dzielą istniejące zagrożenia na kilka grup:

- Włamania i kradzież danych i oprogramowania. Tego typu zagrożenia pojawiły się stosunkowo dawno, i zazwyczaj opierają się na aktywnym przejęciu kontroli nad atakowanym systemem. Po dokonaniu włamania hakerzy mogą wykorzystać zaatakowany system w zasadzie dowolnie - tzn

przejąć zawartość jego dysków twardych wraz z znajdującymi się tam danymi wrażliwymi i / lub oprogramowaniem, podszywać się pod prawowitego właściciela danego systemu, czy też wykorzystać go do dalszych ataków na inne systemy. Jednakże przeprowadzenie udanego włamania na współczesny system jest dość trudne, i stosunkowo niewielu cyberprzestępco&wacute;w jest w stanie taki atak skutecznie przeprowadzič.

- Oszustwa i fałszerstwa. Zagrożenie spotykane najczęściej w sieci rozległej - polega na próbie wyłudzenia informacji za pomocą spersonalizowanych maili, stron, itp. Jak zapewne się domyśliszc, poziom wiedzy technicznej wymaganej do przeprowadzenia takiego ataku jest zdecydowanie niższy niż w przypadku poprzedniego punktu. Można też powiedzieć iż skuteczność takich ataków jest niewielka.
- Szpiegostwo (podsłuch, kradzież danych). W odróżnieniu od oszustw - w przypadku szpiegostwa mówimy o działaniu typowo pasywnym, tzn. podsłuchiwaniu sieci albo na poziomie fizycznym, albo na poziomie wyższych warstw, lecz bez ingerencji ani w podsłuchiwanie systemy, ani w zawartość przesyłanej informacji
- Niszczenie danych. Jest to działanie destrukcyjne, zorientowane na doprowadzenie do strat u właściciela atakowanego systemu.
- Paraliżowanie pracy systemów komputerowych. Ostatnio dość popularne ataki, polegające na uniemożliwianiu normalnej pracy systemów, szczególnie serwerów w internecie, bez ingerencji w spójmoč przechowywanych na nich danych, oraz bez prób przejmowania nad nimi kontroli. Cała złośliwość ataku sprawdza się do uniemożliwienia skorzystania z określonej usługi przez podmioty uprawnione.
- Fałszowanie i uniemożliwianie transmisji danych. To zagrożenie jest raczej elementem wojny elektronicznej niż zagrożeniem spotykany na co dzień w sieci.

Jak widać, jest się czego bać ... skoro tak, to zanim przejdziemy do dokładniejszego omawiania zasadnienia, wprowadźmy kilka pojęć. Zazwyczaj ochronie podlegają dane - ale co to oznacza? w nomenklaturze specjalistycznej stosuje się różne poziomy zabezpieczenia i złamania zabezpieczenia, tzn:

- poufność (ang. confidentiality) - ochrona informacji przed nieautoryzowanym jej ujawnieniem. Tak zabezpieczacie zazwyczaj swoje prywatne dane na dysku - zależy Wam na tym by nikt niepowołany nie był w stanie ich odczytać.
- integralność danych (data integrity) - ochrona informacji przed nieautoryzowanym jej zmodyfikowaniem. Tak zabezpiecza się zawartość publicznych stron www - tam informacja jest dostępna dla każdego, lecz tylko uprawnieni mogą ją modyfikować.
- autentyczność (ang. authenticity) - pewność co do pochodzenia, autorstwa i treści danych. Temu służy podpis elektroniczny, lub system certyfikatów serwera
- niezaprzeczalność (ang. nonrepudiation) = ochrona przed fałszywym zaprzeczeniem przez nadawcę – faktu wysłania danych, przez odbiorcę - faktu otrzymania danych. Na dzień dzisiejszy najtrudniejsza do zapewnienia.

Skoro już jesteśmy przy nomenklaturze, to jeszcze pozwolicie, iż nazwiemy fachowo najczęściej spotykane zagrożenia:

- **hacking** włamywanie do systemów lub sieci komputerowych, naruszenie art. 267 § 1 Kodeksu karnego.
- **sniffing** podsłuchiwanie pakietów w sieci. Skanowanie sieci jest naruszeniem art. 267 § 2 Kodeksu karnego.
- **session hijacking** polega na uzyskiwaniu nieuprawnionego dostępu do systemów poprzez przechwycenie sesji legalnego użytkownika
- **browser hijacker** - oprogramowanie, które modyfikuje ustawienia przeglądarki internetowej użytkownika w celu przechwycenia sesji HTTP.
- **Spoofing** podszywanie się pod innego użytkownika w sieci.
 - **IP spoofing** - rodzaj ataku w którym adres IP napastnika rozpoznawany jest jako adres zaufany.
 - **TCP spoofing** - podszywanie bazujące na oszukaniu mechanizmu generowania numerów ISN.

- **UDP spoofing** – modyfikacja pakietów w celu atakowania usług i protokołów wykorzystujących protokół UDP.
- **ARP spoofing** - rodzaj ataku w którym adres MAC napastnika rozpoznawany jest jako adres zaufany (wysyła fałszywe odpowiedzi na pytania ARP).
- **DNS spoofing** – rodzaj ataku polegający na włamaniu się do serwera DNS (Domain Name System) i zmianie tablic mapowania nazw hostów na adresy IP. Kiedy klient wysyła żądanie podania adresu IP jakiegoś hosta, to w odpowiedzi otrzymuje adres podrobiony.
- **phishing** (password fishing) - metoda uzyskiwania haseł dostępu do zasobów internetowych (kont bankowych użytkownika) za pośrednictwem e-maili. Atakujący podszywa się pod przedstawiciela instytucji finansowej próbując nakłonić odbiorców e-maili do przesłania im haseł oraz innych danych osobowych (np. poprzez fałszowanie stron WWW).
- **cracking** - łamanie zabezpieczeń programów komputerowych, dostępu do systemów, usług sieciowych.
- **piractwo komputerowe** - kopiowanie, reprodukowanie, używanie i wytwarzanie bez zezwolenia produktu chronionego przez prawo autorskie.

Sporo, prawda? A to tylko subiektywny wybór autora tego podręcznika spośród mrowią innych nazw i terminów, zogniskowany na zagrożeniach prowadzących do włamania do systemu komputerowego. Przy czym, jak wspominałem - do włamywania się do systemów niezbędna jest wiedza - zarówno ta techniczna, z dziedziny informatyki, jak też trochę psychologii ... dlaczego psychologii? Bo ogromna większość (wg różnych źródeł ok 80 % do 95 % włamań do systemów opiera się na odgadnięciu hasła dostępu.

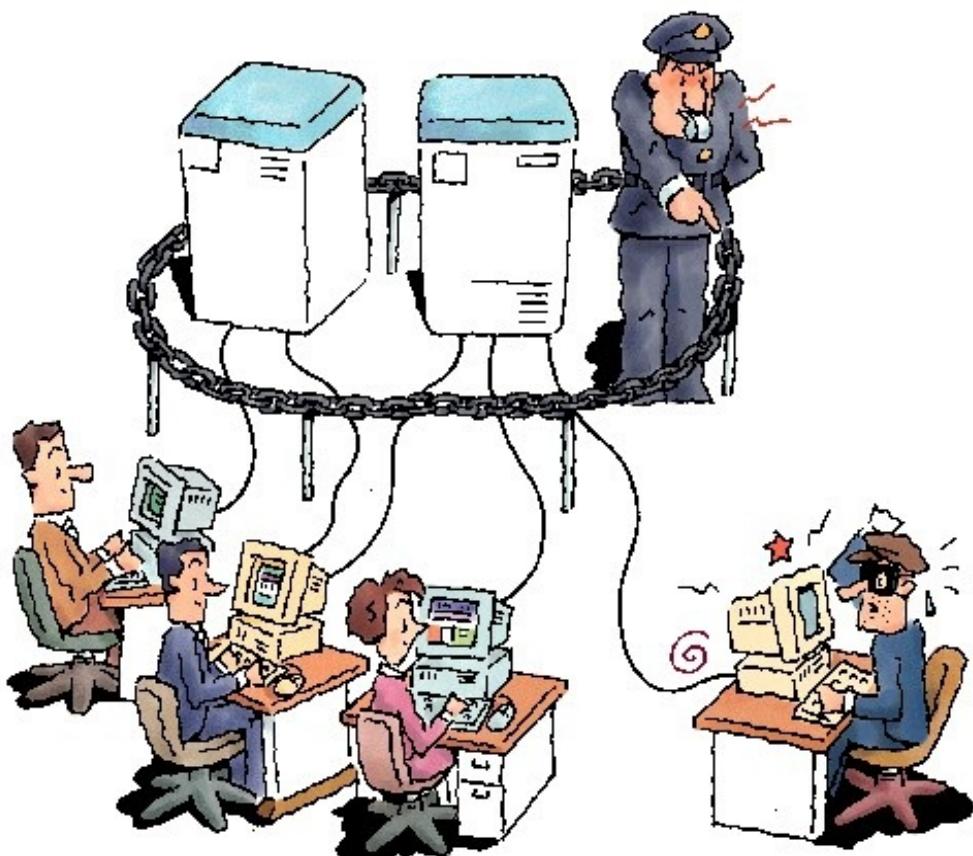
Ostatnimi czasy coraz bardziej "popularnym" zagrożeniem jest paraliżowanie pracy systemów komputerowych - w takich atakach specjalizuje się m.inn. słynna grupa Anonymous. W tym przypadku stosowane są mniej wyrafinowane techniki ataków, jak np:

- Atak **Denial of Service (DoS)**. Celem ataku DoS jest unieruchomienie całego systemu ofiary lub jego komponentów. Współcześnie wykorzystuje się DDOS – atak z wielu komputerów przejętych przez hackerów (tzw. Backnet).
- **Smurf attack.** Jest to atak DDOS (Direct Denial Of Service), polega na wygenerowaniu dużej liczby pakietów 'ICMP echo request' z adresem IP ofiary ataku jako źródłowym. Ofiara zostanie zalana odpowiedziami ping.
- Atak **SYN flood**. W przypadku ataku SYN flood atakujący wysyła na adres ofiary dużą liczbę segmentów SYN protokołu TCP adresowanych z dowolnych (nieistniejących) adresów IP. Ofiara odpowiada segmentami SYN/ACK i rozpoczyna bezwocne oczekiwanie na segmenty ACK. W trakcie oczekiwania wyczerpują się zasoby hosta ofiary. W 1997 r. atak SYN flood na WebCom wyłączył z użycia ponad 3000 stron WWW.
- **Ping of Death.** Atak ten przeprowadza się poprzez wygenerowanie pofragmentowanych pakietów ICMP przekraczających w sumie 64 kB, scalanie może powodować błędy prowadzące do zawieszenia się systemu hosta.
- **Land attack.** Atakujący wysyła segment SYN na adres ofiary podając jej własny adres jako źródłowy i nadając ten sam numer portu źródłowego i docelowego. Stacja TCP ofiary nigdy nie zestawi połączenia zapętlając się w nieskończoność.

Sporo tych zagrożeń ... jak się przed nimi chronić? No cóż, w 100% zabezpieczyć się nie da, ale zawsze coś można zrobić ;)

OCHRONA

Ochrona jest problemem wewnętrznym systemu informatycznego i dotyczy mechanizmów służących do kontrolowania dostępu programów, procesów lub użytkowników do zasobów zdefiniowanych w systemie komputerowym. Zauważcie - iż na poziomie systemu mówi się o dostępie nie tylko do danych, a właśnie zasobów, czyli pojęcia szerszego - zasobem jest też drukarka, pamięć operacyjna, itp ...



CELE OCHRONY

Pierwsze systemy komputerowe wyposażone były w bardzo słabe mechanizmy ochrony (w zasadzie można stwierdzić, iż nie było ich wcale). Wraz z pojawiением się systemów wieloprogramowych problem ochrony zaczął nabierać istotnego znaczenia i stał się bardziej intensywnym obiektem zainteresowań twórców systemów operacyjnych, i szerzej rozumianych systemów informatycznych. Pewne wstępne informacje o ochronie zasobów przez system już Wam przedstawiliśmy w trakcie omawiania systemów operacyjnych - tutaj spojrzymy na to zagadnienie z nieco innego punktu widzenia.

Po pierwsze - większość współczesnych systemów komputerowych pracuje w sieci. Tak więc - ich ochrona rozciąga się także na zasoby rozproszone po sieci. Fachowo mówimy, iż prawidłowa ochrona w systemie dostarcza odpowiednich mechanizmów, które mogą posłużyć do realizacji określonej polityki wykorzystania zasobów systemu, definiowanej zarówno na poziomie lokalnym, jak i zdalnym. Przy czym poprzez politykę wykorzystania zasobów w uproszczeniu możecie rozumieć możliwość pełnej kontroli nad dostępem do nich, odczytem, zapisem, oraz wykonaniem.

Do mechanizmów ochrony należą między innymi:

- bezpieczne dzielenie wspólnej przestrzeni nazw logicznych, np. katalogów plików;
- bezpieczne dzielenie wspólnej przestrzeni obiektów fizycznych, np. pamięci.

Podstawowym celem stosowania ochrony jest zapewnienie, aby każdy wykonywany proces korzystał tylko i wyłącznie z zasobów mu przyznanych i to w sposób określony przez politykę ochrony. Samo definiowanie polityki może być realizowane w różny sposób. W lekcji poświęconej plikom omawialiśmy zasady i prawa dostępu do nich - to właśnie, w połączeniu z tworzeniem grup użytkowników i przydzielaniem ich do nich - jest prostą realizacją zapisu polityki dostępu do plików na dysku. W praktyce, w skomplikowanych, rozproszonych systemach informatycznych stosuje się daleko bardziej skomplikowane mechanizmy ochrony.

DOMENOWY MODEL OCHRONY

Współcześnie najbardziej popularnym rozwiązaniem jest domenowy model ochrony zasobów. Ogólnie mówiąc.

Domena ochrony jest zbiorem obiektów i rodzajów operacji, które można wykonać dla danego obiektu.

Obiektem może być wszystko to co system operacyjny obsługuje i udostępnia: segment pamięci, procesor, drukarka, dysk, plik, program itd. W zależności od rodzaju z danym obiektem związane są określone operacje. Na przykład z plikiem związane są takie operacje jak: otwarcie, zamknięcie, odczyt, zapis (modyfikacja), usunięcie czy wykonanie, a z urządzeniem USB - możliwość zamontowania, odmontowania, odczytu jakichkolwiek danych, ich zapisu czy wykonania programów z nośnika.

W modelu domenowym korzysta się również z pojęcia prawa dostępu - nadawanego użytkownikowi systemu, zarówno jeśli tym użytkownikiem jest człowiek, jak i inny system informatyczny czy też uruchomiony program (proces) w systemie. Prawo dostępu definiuje nam zbiór operacji, które można wykonać na określonym obiekcie. W prawidłowo chronionym systemie trzymamy się założenia, iż nigdy nikt nie powinien mieć większych praw niż są one mu niezbędne do realizacji swoich zadań. Fachowo takie podejście nazywamy zasadą wiedzy koniecznej.

Zasada wiedzy koniecznej (ang. need-to-know) mówi, że każdy proces ma w systemie dostęp tylko do tych zasobów, które są mu niezbędne potrzebne do zakończenia zadania i do których otrzymał prawo dostępu.

Ścisłe trzymanie się tej zasady pozwala na minimalizację szkód - np. w wyniku włamania na konto użytkownik może wykonać jedynie te operacje, co do których ma prawo wykonania, w przypadku przejęcia kontroli nad procesem możliwe jest wykonanie jedynie tych operacji, co do których wykonania uprawniony jest dany proces, itp...

Zazwyczaj domena ochrony jest w systemach informatycznych powiązana z użytkownikiem danego systemu - i tak, uruchomiony program ma takie prawa jak użytkownik który go uruchamiał. Jeśli więc użytkownik nie ma prawa do zmiany zawartości katalogu systemowego, to żaden program uruchomiony przez tego użytkownika zmian w katalogach systemowych nie wykona.

Niestety, zasada wiedzy koniecznej w przeszłości nazbyt często była ignorowana. Przez długi czas (w zasadzie do dnia dzisiejszego) w systemach z rodziny Windows normalny użytkownik pracował z uprawnieniami administratora. W rozwiązaniach Unix-owych i pochodnych ta zasada jest dotrzymywana - tam wyraźnie zaznacza się, iż administrator, nazywany *root* - jest tylko jeden, i nie można korzystając z tego konta pracować jako użytkownik. Konto root jest wykorzystywane jedynie do zadań konfiguracyjnych.

Jeszcze dalej ochrona jest rozwinięta w systemach dla urządzeń przenośnych (Android, iOS) - tam,

ponieważ zazwyczaj użytkownik jest tylko jeden, prawa nadaje się każdej aplikacji z osobna.

BEZPIECZEŃSTWO

Bezpieczeństwo jest zagadniением szerszym niż ochrona systemu. Bezpieczeństwo systemu polega na zapewnieniu nienaruszalności systemu przez czynniki zewnętrzne. Uważamy, że system jest bezpieczny jeśli jest chroniony przed zagrożeniem ze strony środowiska, na przykład przed próbami naruszenia poufności danych.

POLITYKA BEZPIECZEŃSTWA

Bezpieczeństwo systemu zależy w dużej mierze od przyjętej polityki bezpieczeństwa. Nawet najlepsze zabezpieczenia nie uchronią przed kradzieżą lub zniszczeniem danych, jeśli polityka bezpieczeństwa pozwala na swobodny dostęp do konsoli administratora. Z tego punktu widzenia, polityka bezpieczeństwa, rozumiana jako zbiór wszystkich działań przyjętych w celu osiągnięcia wysokiego poziomu bezpieczeństwa systemu informatycznego, jest zagadniem kluczowym, i złożonym. Podstawowymi mechanizmami stosowanymi w celu zapewnienia bezpieczeństwa w systemie jest dokonywanie identyfikacji, uwierzytelniania oraz autoryzacji użytkowników systemu.

IDENTYFIKACJA

Identyfikacja polega na stwierdzeniu tożsamości użytkownika lub innego obiektu zamierzającego skorzystać z zasobów systemu. Najczęściej każdy użytkownik systemu ma przypisany unikalny w obrębie systemu identyfikator lub numer. Dzięki stosowaniu identyfikatorów system wie z jakim użytkownikiem ma do czynienia. Może również dokonywać rejestracji działań wykonanych przez tego użytkownika i na tej podstawie przedstawiać raporty lub dokonywać rozliczeń.

UWIERZYTELNIANIE

Sama identyfikacja nie może jednak zapewnić bezpieczeństwa systemu. Identyfikatory użytkowników są najczęściej tworzone na podstawie dostępnego algorytmu i są wobec tego znane lub łatwe do odgadnięcia. Z tego powodu poza procesem identyfikacji przeprowadzany jest proces uwierzytelniania. Uwierzytelnianie polega na sprawdzeniu, że obiekt, który zgłasza żądanie dostępu do systemu jest tym za kogo się podaje.

HASŁA

Najczęstszym sposobem uwierzytelniania użytkowników systemu jest wymaganie podania hasła. Długość i rodzaj hasła są oczywiście zależne od systemu (np. niektóre systemy przyjmują hasła o długości do 8 znaków lub nie rozróżniają wielkich i małych liter) i od przyjętej polityki bezpieczeństwa, ale warto jest przestrzegać następujących ogólnych zasad:

- hasło powinno być dostatecznie długie;
- hasło nie powinno być słowem słownikowym (np. imieniem członka rodziny, psa, datą urodzenia itd.);
- hasło powinno składać się ze znaków alfanumerycznych czyli zawierać zarówno litery jak i cyfry. Jeśli to możliwe należy stosować kombinacje dużych i małych liter;
- nie należy zapisywać hasła w łatwo dostępnym miejscu, a już na pewne nie należy umieszczać go

na obudowie komputera (!)

W praktyce dość silne, i jednocześnie łatwe do zapamiętania hasła zazwyczaj tworzymo korzystając ze znanego nam algorytmu, oraz zdań / stwierdzeń które są dla nas na tyle istotne, że je pamiętamy. Typowy przykład: Lom2yead. Na pierwszy rzut oka nie ma to sensu - ale to początek Pana Tadeusza: pierwsze litery z "Litwo ojczyzno moja", drugie z "ty jesteś jak zdrowie". Finalnie otrzymujemy hasło o statystycznej charakterystyce zbliżonej do rozkładu losowego, jednocześnie nie występujące w słownikach. Jeśli jeszcze w coś takiego wpłaciemy znaki specjalne, i cyfry - hasło będzie też trudne do złamania.

Ze względu na swoją naturalność i prostotę systemy identyfikacji opierające swoje działanie na hasłach są wciąż najczęściej stosowane w dzisiejszych systemach komputerowych. Stosowanie haseł w celu określenia tożsamości posiada jednak wiele wad. Najważniejsze z nich to:

- komputer musi przechowywać hasła użytkowników. Co prawda - w dobrze skonfigurowanych systemach nigdy nie przechowuje się ich w jawnej, lub możliwej do odkodowania postaci - lecz zawsze jakaś postać pozostaje;
- hasło może zostać przechwycone w czasie przesyłania go do komputera (dotyczy to szczególnie systemów uwierzytelniania w sieci). Dlatego też zostały opracowane systemy które nie wymagają przesyłania haseł - np omówiony w dalszej części Kerberos;
- użytkownicy zapominają swoje hasła;
- użytkownicy wybierają hasła, które łatwo odgadnąć. Tutaj elementem przeciwdziałającym może być wymuszanie na użytkowniku stosowania haseł, które nie są trywialne;
- użytkownicy ujawniają swoje hasła innym osobom.

PRZEDMIOTY

Oprócz lub obok uwierzytelniania za pomocą hasła stosuje się też uwierzytelnianie za pomocą specjalnych identyfikatorów (kart chipowych lub tokenów). Takie systemy również mają swoje wady. Przedmioty używane do identyfikacji w rzeczywistości nie potwierdzają tożsamości danej osoby, lecz przedmiotu. Każdy, kto zdobędzie taki przedmiot będzie mógł podszywać się pod jego właściciela. Z tego też powodu takie systemy są uzupełnieniem systemów opartych na hasłach. Do tej samej grupy zaliczamy też systemy oparte na jednorazowych hasłach przesyłanych na telefon komórkowy.

INDYWIDUALNE CECHY UŻYTKOWNIKA

Do uwierzytelniania stosowane są też metody biometryczne wykorzystujące fakt unikatowości pewnych cech fizycznych człowieka – takich jak linie papilarne, czy rysunek tęczówki oka lub też cech behawioralnych takich jak sposób mówienia, czy pisania.

Typowym przykładem uwierzytelniania na podstawie cech fizycznych użytkownika jest skanowanie linii papilarnych. Rozpoznawanie tej cechy dowiodło już swoją skuteczność, wiarygodność i wygodę. Skanowanie obrazu odcisku palca zabiera mało czasu i wysiłków użytkownika oraz jest jedną z najmniej inwazyjnych metod biometrycznych. Weryfikacja odcisku linii papilarnych jest również stosunkowo szybka, choć nie jest stuprocentowo bezpieczna.

W chwili obecnej weryfikacja użytkowników za pomocą ich cech fizycznych lub behawioralnych osiąga coraz większą popularność. Część popularnych laptopów (np. firmy Lenovo) jest standardowo wyposażana w czytniki linii papilarnych. Technika ta jest używana razem z weryfikacją opartą na hasle. Taki dwustopniowy proces uwierzytelniania zapewnia wyższy poziom bezpieczeństwa, niż każda z tych metod oddzielnie. Natomiast przestrzegamy przed bezwarunkową wiarą w odporność na oszukanie tanich skanerów linii papilarnych.

Bezpieczne są skanery aktywne, prześwietlające wierzchnią warstwę skóry - lecz tych nie znajdziecie standardowo wbudowywanych w PC.

AUTORYZACJA

Po przeprowadzeniu identyfikacji i uwierzytelnienia obiektu system przydziela danemu obiekowi określone praw dostępu do obiektów systemu informatycznego - ten etap jest zazwyczaj nazywany autoryzacją.

ZAGROŻENIA BEZPIECZEŃSTWA

Zagrożenia bezpieczeństwa systemu można podzielić na dwie zasadnicze grupy. Pierwsza grupa, to zagrożenia wynikające ze świadomego działania człowieka. Druga, to nieświadome działania użytkowników, awarie sprzętu, zaniki zasilania oraz oddziaływanie czynników zewnętrznych (pożary, powodzie, katastrofy).

UŻYTKOWNICY

Najczęstszym zagrożeniem dla systemu informatycznego jest działanie czynnika ludzkiego, a w tym użytkowników systemu. Z badań wynika, że większość naruszeń bezpieczeństwa systemu jest wynikiem działań wewnętrz tego systemu (organizacji). Naruszenia te są zarówno wynikiem działań świadomych (włamań), jak też zwykłej nieuwagi czy też niedbalstwa użytkowników. Do najistotniejszych zagrożeń wewnętrznych należy zaliczyć:

- sabotaż wewnętrzny;
- kradzież informacji;
- kradzież usług;
- błędy użytkowników;
- niedbalstwo;
- nieprawidłowe stosowanie procedur polityki bezpieczeństwa.

ATAKI NA SYSTEM

Atak na system przeprowadzany jest najczęściej poprzez przechwycenie lub odgadnięcie danych uprawnionego użytkownika.

Do najczęstszych metod ataku należą:

- Włamanie (ang. *hacking*) do systemów lub sieci komputerowych (stanowi naruszenie art. 267 §1 Kodeksu Karnego). Włamanie może następować za pomocą najróżniejszych dróg i technik, zarówno korzystając z luk w oprogramowaniu systemowym i jego zabezpieczeniach, jak i z naiwności użytkowników, przy czym to drugie działanie jest zdecydowanie częściej spotykanym przypadkiem. Finalnym efektem włamania często jest przejęcie kontroli nad atakowanym systemem. Odbywa się to najczęściej albo poprzez nieautoryzowane zainstalowanie na nim oprogramowania hackera, albo poprzez zdobycie informacji umożliwiającej prawidłowe uwierzytelnienie się w systemie (zdobycie haseł). W przypadku instalacji nieautoryzowanego

oprogramowania, zazwyczaj dzieli się je na 4 grupy:

Wirusy, czyli programy, które rozprzestrzeniają się poprzez infekowanie innych plików kopią samego siebie. Do plików, które mogą zostać zainfekowane należą wszystkie binarne pliki programowe, oraz pliki dokumentów dla aplikacji, które zawierają obsługę różnych makrojęzyków na tyle zaawansowanych, by umożliwić działanie wirusa. Niektóre wirusy oprócz zakażania plików infekują, zmieniają lub nadpisują sektory startowe dysku (MBR).

Robale - czyli programy komputerowe, które szerzą się przez wysyłanie swoich kopii do innych komputerów, które uruchamiają go i dalej rozprzestrzeniają do innych komputerów. Najczęściej robale rozprzestrzeniają się za pomocą programów pocztowych. Robal wysyła swoje kopie listownie do wszystkich adresatów książki adresowej użytkownika. Taka sama rzecz przydarza się dalej tym adresatom i w ten sposób robal potrafi skutecznie rozprzestrzenić się w Internecie w przeciągu kilku godzin.

Konie trojańskie, programy, które rzekomo ma wykonać dla nas coś pożytecznego lub interesującego, a w rzeczywistości przeprowadzają w tle złośliwe działania, podczas gdy użytkownik posługuje się głównym programem.

Bomby logiczne, złośliwe fragmenty kodu programującego wstawione w normalne pod pozostałymi względami programy. Często są tam umieszczone przez autora programu lub osobę, która go współtworzy. Bomby logiczne mogą być ustawione tak, aby zaczęły działać w określonej porze, lub dniu.

Dzięki zainstalowaniu takiego oprogramowania hacker ma możliwość zarówno kradzieży naszych danych, jak i zdalnego wykonywania operacji korzystając z naszego systemu. Typowym przykładem konia trojańskiego jest zmiana oryginalnego programu rejestrującego użytkowników do systemu. Program taki przechwytuje dane wprowadzane przez użytkownika i przesyła je do osoby atakującej. Następnie przekazuje sterowanie prawdziwemu programowi rejestrującemu.

Najlepszą ochroną przed włamaniem jest rozsądek - nie należy nigdy uruchamiać oprogramowania pochodzącego z nieznanych źródeł. Ponadto pomocne mogą być wszelkiego rodzaju programy antywirusowe.

- Podsłuch połączenia sieciowego (ang. *sniffing*) jest popularnym rodzajem ataku, który często prowadzi do utraty tajności danych. Skanowanie sieci jest naruszeniem art. 267 § 2 Kodeksu Karnego. W wyniku podsłuchu można uzyskać między innymi: hasła, identyfikatory kont oraz dane prywatne. Podsłuch jest atakiem biernym (nie czyni więc bezpośrednich szkód w systemie), ale bardzo niebezpiecznym, ponieważ najczęściej stanowi wstęp przed zastosowaniem innych metod. Najważniejszym narzędziem w walce z podsłuchem połączenia sieciowego w przypadku połączeń kablowych jest podział sieci komputerowej na osobne fragmenty (segmenty). W idealnej sytuacji każda maszyna powinna należeć do osobnego segmentu, a jej interfejs nie powinien mieć dostępu do danych dla niego nie przeznaczonych. W praktyce stosuje się też często koncepcję tak zwanych zaufanych systemów (segmentów). Pod tym pojęciem rozumie się system, w którym wszystkie maszyny w danym segmencie są bezpieczne. W celu osiągnięcia takiego stanu komputery oraz połączenia pomiędzy nimi muszą posiadać wystarczającą ochronę fizyczną (zamki w drzwiach, strażnicy, zabezpieczenia przed emisją ujawniającą), aby mieć pewność, że włamywacz nie będzie mógł zainstalować w tym segmencie urządzenia podsłuchującego. Przy transmisji danych należy stosować bezpieczne protokoły oraz szyfrowanie. Należy także unikać przesyłania haseł w formie jawniej. W przypadku sieci bezprzewodowej jedną formą zabezpieczenia jest szyfrowanie całej komunikacji, przy czym muszą być to szyfry silne. Stosowane domyślnie przez sieć WiFi czy GSM algorytmy szyfrujące nie mogą być zaliczone na dzień dzisiejszy do grupy silnych szyfrów, i jako takie - nie zabezpieczają sieci przed podsłuchem. Dlatego też w przypadku sieci bezprzewodowych bezwzględnie zaleca się stosowanie albo szyfrowanych protokołów (np. https) albo rozwiązań szyfrujących całą komunikację pomiędzy dwoma hostami, takich jak VPN.
- Łamanie haseł. Najczęściej stosowanymi metodami ataku w celu uzyskania nieautoryzowanego dostępu są tzw. metody korzystające z haseł. Polegają one na podjęciu próby przeniknięcia do systemu poprzez podanie identyfikatora użytkownika i hasła. Atakujący może próbować wielu haseł, aż do momentu podania właściwego. Szybko zorientowano się, że nie jest trudno napisać program generujący rozmaite haseła. Obecnie istnieje wiele tego typu programów, które skutecznie działają na różnych platformach systemowych. Używają one zbiorów słów (słowników), dlatego też ataki tego typu znane są jako metody słownikowe. Programy takie mogą

również próbować wszystkich możliwych kombinacji znaków, jakie mogą wystąpić w hasle – jest to wtedy tak zwany atak brutalny lub wyczerpujący. Skutecznym sposobem obrony przed atakami na hasła jest stosowanie podstawowych lub zaawansowanych sposobów tworzenia haseł, systemowe blokowanie konta użytkownika po określonej liczbie błędnych wprowadzeń hasła, stosowanie sprawdzonych mechanizmów powiadamiania o próbach nieautoryzowanego dostępu oraz szyfrowanie plików zawierających hasła.

- Podsywanie się (ang. *spoofing*) pod inną maszynę z sieci może wystąpić na każdej z warstw protokołu TCP/IP (spoofing adresu sprzętowego, spoofing ARP, spoofing routingu IP, spoofing nazw DNS, spoofing połączeń TCP). Komputery wymieniające między sobą informacje przekazują nawzajem dane o sobie. Osoba niepowołana może wysłać do komputera-odbiorcy fałszywe informacje o swoim komputerze, które świadczą, że jest on bezpiecznym komputerem głównym znajdującym się wewnątrz sieci lub poza nią. Podczas ataku tego typu pakiety wysyłane przez intruza mają dostęp do systemu, do którego się on włamuje i do usług tego systemu.
- Zablokowanie usługi – DoS (ang. *Denial of Service*) jest to przerwanie dostarczania usługi spowodowane zniszczeniem systemu lub jego chwilową niedostępnością. Zagrożenie to dotyczy konkretnych usług realizowanych przez system, takich jak WWW, FTP czy poczta elektroniczna, lub też całego serwera. Ataki DoS najczęściej są atakami zewnętrznyimi, odbywającymi się z zewnątrz sieci lokalnej na przykład z Internetu, uzyskując dostęp poprzez lukę w systemie zabezpieczeń. Często atakujący za pomocą techniki spoofingu ukrywa swój prawdziwy adres internetowy tak, że zlokalizowanie go często staje się niemożliwe. Współcześnie wykorzystuje się DDoS (ang. *Distributed Denial of Service*) – atak z wielu komputerów przejętych przez hackerów (tzw. Backnet). Ochrona przed takim atakiem jest trudna, w praktyce często wymaga interwencji administratora i ręcznego blokowania określonych grup adresów IP.

WIRUSY KOMPUTEROWE

Wirusy i konie trojańskie należą o grupy programów powodujących zakłócenie pracy systemu informatycznego. Pod pojęciem wirus komputerowy rozumie się program, który potrafi się rozmnażać i dopisywać w postaci ukrytej do innych programów lub w sektorach rozruchowych dysków. Najlepszym sposobem ochrony przed wirusami komputerowymi jest rozsądek - czyli korzystanie tylko z bezpiecznych źródeł danych i oprogramowania, w połączeniu ze stosowaniem programów antywirusowych. Powszechną grupę wśród wirusów stanowią makrowirusy, czyli wirusy tworzone za pomocą języków makropoleceń dostępnych na przykład w edytorech tekstów takich jak MS Word.

Zagrożenie wirusami i ich pochodnymi dotyczy głównie systemów z rodziny Windows. Liczba wirusów zdolnych zaatakować systemy Unixowe czy MacOS jest znikoma, i nie istnieje uzasadniona potrzeba stosowania oprogramowania antywirusowego dla tych systemów

AWARIE SPRZĘTU

Przyczyną utraty danych może być też awaria sprzętu. Wiele systemów nie może pozwolić sobie na utratę danych lub przestoje spowodowane awarią sprzętu. Aby osiągnąć odpowiednią niezawodność stosuje się następujące metody:

- wykonywanie stałych kopii zapasowych.
Kopie zapasowe powinno wykonywać się codziennie. Jest to jedna z prostszych i mniej kosztowych metod zapewnienia bezpieczeństwa danych. Powinna być stosowana nawet przez pojedynczego użytkownika – w tym również przez Ciebie drogi Czytelniku. Jednak wykonywanie kopii bezpieczeństwa nie rozwiązuje całkowicie problemu awaryjności sprzętu, gdyż pozwala na odzyskanie danych aktualnych w momencie wykonania ostatniej kopii.

- macierze RAID
Stanowią one zestaw kilku dysków magnetycznych traktowanych przez system operacyjny jak jeden dysk logiczny. Istnieje wiele rozwiązań tego typu różniących się szybkością działania, możliwością i szybkością odtwarzania danych oraz kosztami eksploatacji.
- zasilacze awaryjne
przed zanikiem napięcia zasilającego można uchronić stosując specjalne akumulatorowe zasilacze awaryjne (UPS) lub generatory prądotwórcze. Urządzenia takie pozwalają na normalną pracę systemu od kilku minut do kilku godzin. Praca zasilaczy awaryjnych może być nadzorowana przez system operacyjny.

KRYPTOGRAFIA

Kryptografia dostarcza narzędzi dzięki którym można, stosując metody matematyczne, zabezpieczać zarówno dane gromadzone w lokalnych magazynach informacji jak i dane przesypane przez sieć.

Szyfrowanie informacji ma szczególne znaczenie, jeśli korzystamy z systemów otwartych takich jak na przykład Internet. Rozwój bezpiecznych metod kryptograficznych w znaczący sposób przyczynił się do rozwoju handlu elektronicznego i bankowości elektronicznej.

FUNKCJE SKRÓTU, CZYLI SZYFROWANIE BEZ MOŻLIWOŚCI ODSZYFROWYWANIA

Zanim przejdziemy do opisu pełnoprawnych szyfrów, przyjrzymy się działaniu z pozoru bez sensu: czy da się zaszyfrować informację tak, aby nikt, nawet autor szyfru ani szyfrujący, nie mógł jej odczytać? Matematycy już dość dawno udowodnili, że jest to możliwe, dzięki stosowaniu tzw. **funkcji skrótu**. Opracowano ich wiele, najpopularniejszym przedstawicielem jest na dzień dzisiejszy algorytm **md5**. Ogólnie rzecz ujmując - funkcja skrótu przyjmuje dowolnie długi ciąg bajtów, i przekształca go w inny ciąg nazywany **skrótem**, zazwyczaj o stałej długości (np 32 bajtów), niezależnie od długości ciągu pierwotnego. Przekształcenie ponadto charakteryzuje się dwoma cechami:

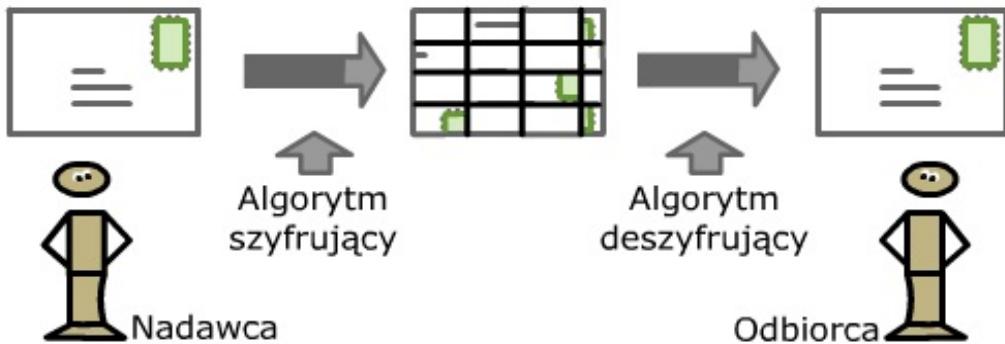
- jednostronność - funkcje skrótu nie mają funkcji odwrotnych,
- wysoka prędkość ucieczki - zmiana choćby jednego bitu w łańcuchu pierwotnym powoduje kompletną zmianę wyniku przekształcenia.

W oparciu o takie funkcje działa mechanizm identyfikacji użytkownika przez system lokalny. W systemach Unixowych hasła użytkowników przechowywane są na dysku w postaci zaszyfrowanej, właśnie w postaci skrótu oryginalnego hasła. Gdy wpiszemy hasło, system od razu zamienia je na postać zaszyfrowaną i porównuje z zaszyfrowanym hasłem użytkownika, za którego się podaliśmy. Jeśli zgadzają się postacie zaszyfrowane, to znaczy, że wpisaliśmy ten sam ciąg znaków, co właściciel konta przy ostatniej zmianie hasła. Jeśli natomiast ktoś uzyska dostęp do naszego dysku twardego - to ze skrótu i tak nigdy nie będzie w stanie odtworzyć oryginalnej postaci hasła. Jedyne co można to hasło „zgadnąć”. Jeśli zaszyfrujemy kilka „podejrzanych” wyrazów, jak imiona najbliższych i ulubione sporty użytkownika, to może się zdarzyć, że „trafimy” i otrzymamy identyczny ciąg znaków, jak zaszyfrowane hasło.

SZYFROWANIE

Szyfrowanie jest metodą powszechnie stosowaną do ochrony informacji. Choć metody szyfrowania zmieniały się na przestrzeni wieków, to ogólny mechanizm wygląda ciągle tak samo. Można go przedstawić w następujący sposób:

1. Tekst jawny zostaje poddany obróbce zgodnie z algorytmem szyfrowania, czyli podlega kodowaniu. W wyniku tej operacji otrzymuje się tekst, który można co prawda czytać (w sensie odczytu znaków), ale nie można poznać jego sensu.
2. Tekst zaszyfrowany przesyłany jest do odbiorcy normalnymi, niechronionymi kanałami.
3. Po odebraniu tekstu zaszyfrowanego, odbiorca stosując algorytm deszyfrowania dekoduje go do postaci czytelnej.



Proces szyfrowania i deszyfrowania

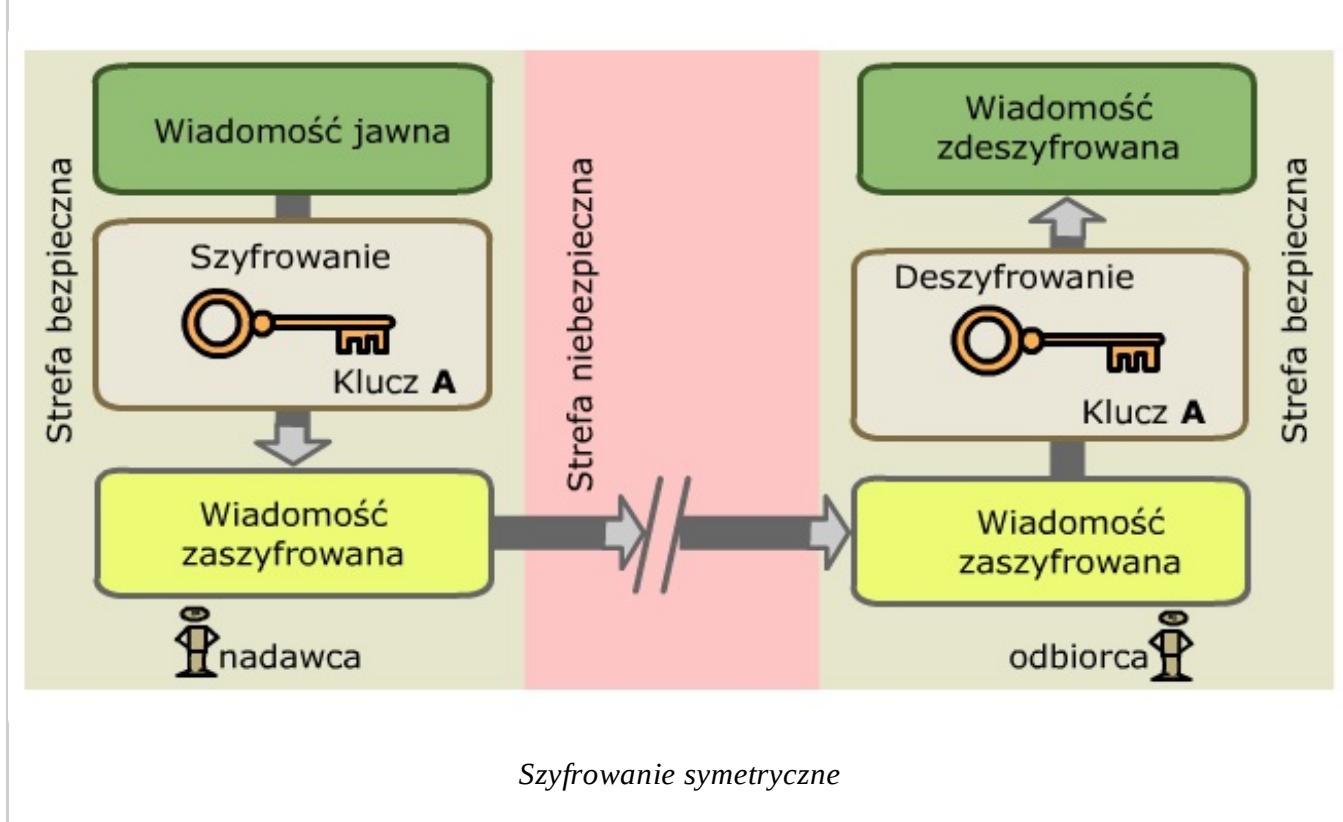
Aby można było stosować opisaną powyżej metodę do bezpiecznego przesyłania informacji, to odczytanie tekstu zaszyfrowanego bez znajomości algorytmu deszyfrującego powinno być bardzo trudne lub niemożliwe. Istnieje wiele metod spełniających te wymaganie. Ogólnie wyróżniamy dwie metody szyfrowania: symetryczne i asymetryczne.

SZYFROWANIE SYMETRYCZNE

W szyfrowaniu symetrycznym zarówno odbiorca jak i nadawca posługują się tajnym kluczem służącym do szyfrowania i deszyfrowania wiadomości (dlatego symetryczne). Przechwycenie tego klucza powoduje możliwość odczytania każdej zaszyfrowanej wiadomości.

Aby można było stosować tę metodę komunikacji muszą być dostępne następujące elementy:

- ogólny algorytm szyfrowania E ;
- ogólny algorytm deszyfrowania D ;
- tajny klucz (lub klucze) służący do szyfrowania i deszyfrowania informacji.



Klucze prywatne są powszechnie stosowane przez protokoły bezpieczeństwa, jako klucze sesji w poufnej komunikacji w trybie on-line. Na przykład protokół IPSec wykorzystuje symetryczne klucze sesji ze standardowymi algorytmami stosowanymi do szyfrowania i deszyfrowania poufnej komunikacji między stronami. Dla każdej poufnej sesji komunikacji używane są inne klucze.

Szyfrowanie symetryczne jest też powszechnie stosowane przez technologie zapewniające masowe szyfrowanie trwałych danych, takich jak wiadomości e-mail czy pliki typu dokument. Protokół S/MIME stosuje klucze symetryczne do szyfrowania wiadomości poufnej poczty, a system szyfrowania plików EFS w Windows 2000/XP używa symetrycznych kluczy do szyfrowania plików.

Powszechnie wykorzystywany, symetrycznym algorytmem szyfrującym jest algorytm DES (*ang. data-encryption standard*), który w ogólnych założeniach wygląda następująco:

Jeśli przez E_k oznaczamy algorytm szyfrowania z kluczem k , a przez D_k – algorytm deszyfrowania z tym kluczem, to wówczas dla każdej wiadomości m muszą być spełnione następujące warunki:

1. $D_k(E_k(m))=m$ – oznacza to, że po zaszyfrowaniu wiadomości ($E_k(m)$) przesłaniu jej, a następnie odszyfrowaniu ($D_k(E_k(m))$) nie zgubimy informacji, czyli że proces szyfrowania i deszyfrowania nie zniekształca wiadomości;
2. Obliczenie E_k i D_k jest efektywne, czyli daje się wykonać w rozsądny czasie przy zastosowaniu dostępnym zasobów obliczeniowych;
3. Bezpieczeństwo systemu zależy tylko od tajności klucza k , a nie od tajności algorytmów E i D – oznacza to, że algorytmy szyfrowania (E) i deszyfrowania (D) mogą być ujawnione, a chronić należy jedynie wartość klucza k .

Szczegółowy opis działania algorytmu DES wykracza poza zakres tego wykładu.

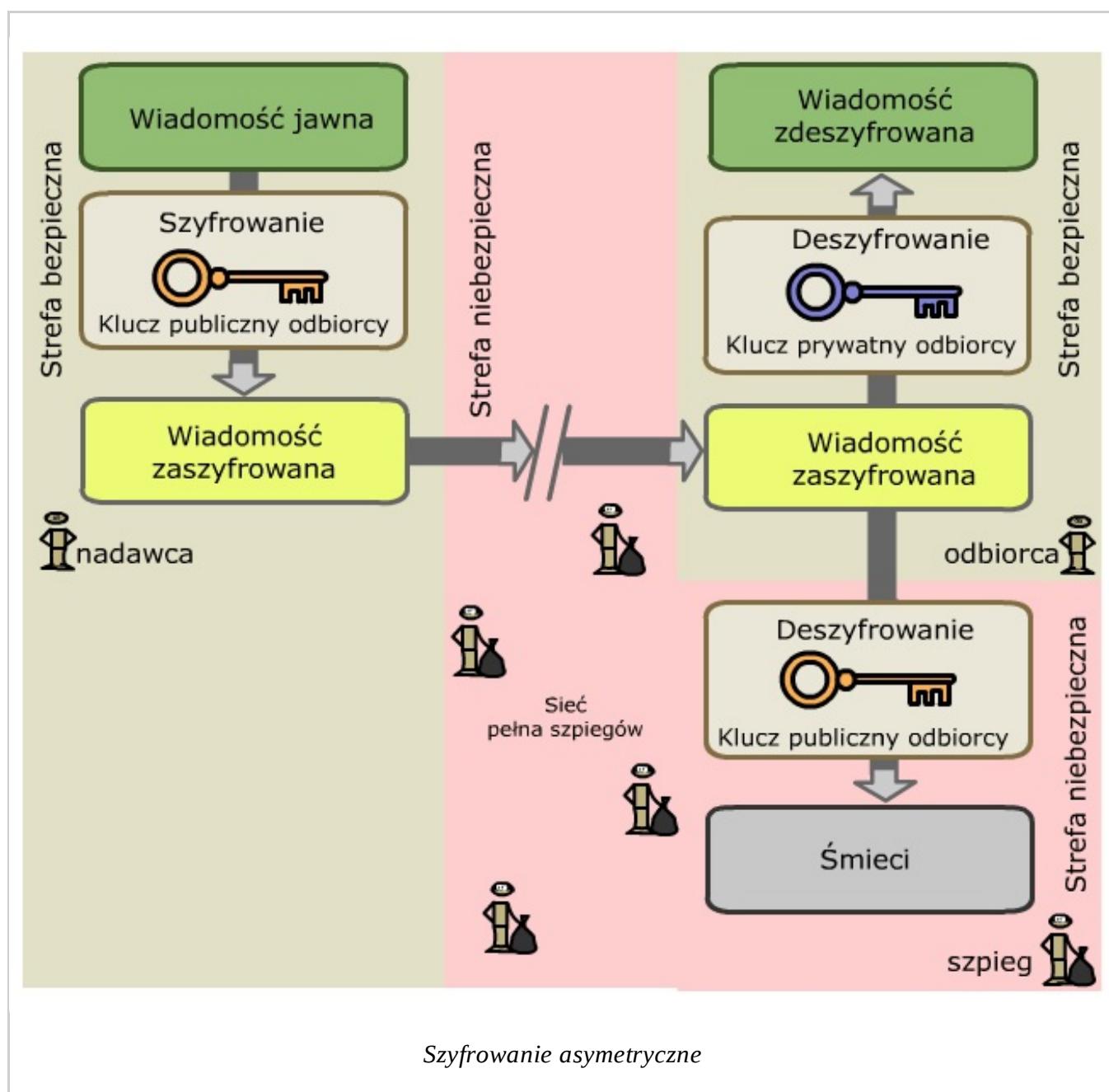
Gwałtowny wzrost mocy obliczeniowej komputerów doprowadził do tego, że obecnie algorytm DES można złamać w ciągu zaledwie kilku godzin. Dlatego opracowany został algorytm 3-DES (Triple DES), w którym zwiększone bezpieczeństwo uzyskano poprzez zastosowanie dwóch lub trzech kluczy. Dzięki zastosowaniu 3-DES czas potrzebny na złamanie szyfru znacząco się wydłuża – na dzień dzisiejszy liczony jest w milionach lat.

Problemem w szyfrowaniu symetrycznym jest przekazywanie kluczy. Aby zapewnić bezpieczeństwo informacji klucz taki musi pozostać tajny. Nie może więc być bezpośrednio przesłany przez sieć telekomunikacyjną, gdyż byłby narażony na łatwe przechwycenie.

Problem ten rozwiązało poprzez szyfrowanie z wykorzystaniem różnych kluczy do szyfrowania i deszyfrowania. Ten sposób szyfrowania nosi nazwę szyfrowania asymetrycznego (lub publicznego). Polega on na zastosowaniu dwu kluczy: jednego do szyfrowania wiadomości, a drugiego do deszyfrowania.

SZYFROWANIE ASYMETRYCZNE

W szyfrowaniu asymetrycznym każdy użytkownik dysponuje dwoma kluczami. Jeden z nich to klucz jawnny służący do szyfrowania wiadomości. Drugi to klucz tajny, znany tylko odbiorcy wiadomości. Służy on do odszyfrowania wiadomości.



Dwaj użytkownicy mogą się skutecznie porozumiewać znając swoje klucze jawnie.

Zasada działania algorytmu szyfrowania opartego na kluczach prywatnych bazuje na własnościach matematycznych dużych liczb. Polega ona na tym, że mnożenie dwóch dużych, specjalnie dobranych liczb jest łatwe. Trudne jest natomiast wykonanie rozkładu liczby na czynniki (o ile liczba jest odpowiednio duża i dobrze dobrana). Oznacza to, że odszyfrowanie wiadomości bez znajomości klucza tajnego wymaga ogromnych mocy obliczeniowych (a co za tym idzie czasu) lub jest niewykonalne.

Najbardziej popularnym algorytmem z kluczem publicznym jest algorytm RSA opracowany przez Ronaldą Rivesta, Adi Shamira i Leonarda Adlemana. Bazuje on na trudności rozłożenia dużej liczby na czynniki pierwsze. Wykorzystuje funkcję Eulera Totient

$$\phi(n)$$

zdefiniowaną jako ilość liczb naturalnych mniejszych od n i względnie pierwszych z n . Liczby m i n są względnie pierwszymi, jeśli nie mają wspólnych podzielników innych niż 1. Funkcja $\phi(n)$ zawsze przyjmuje wartość mniejszą niż n . Euler odkrył, że każda liczba k , względnie pierwsza z n podniesiona do potęgi $\phi(n)$ modulo n daje w wyniku 1:

$$k^{\phi(n)} \bmod n = 1$$

($a \bmod b$ oznacza resztę z dzielenia liczby a przez b . Na przykład $10 \bmod 3 = 1$; $4 \bmod 2 = 0$)

W RSA wykorzystuje się jeszcze jedną własność. Jeżeli k i l są losowymi liczbami naturalnymi będącymi odwrotnościami modulo $\phi(n)$ oraz A jest dowolną liczbą względnie pierwszą z n to:

$$(A^k)^l \bmod n = (A^l)^k \bmod n = A$$

Jeśli A jest częścią wiadomości wówczas szyfrowania dokonujemy za pomocą funkcji

$$S = A^k \bmod n$$

zaś deszyfrowania za pomocą funkcji

$$A = S^l \bmod n$$

Wyboru kluczy szyfrujących dokonuje się w następujący sposób:

- losuje się dwie duże liczby pierwsze p, q ;
- oblicza się $n = pq$, oraz wyznacza funkcję $\phi(n)$
- losowo wybiera się liczbę e z przedziału $(1, \phi(n))$, względnie pierwszą z $\phi(n)$;
- wyznacza się liczbę d odwrotną do $e \bmod \phi(n)$, czyli $d = e^{-1} \bmod \phi(n)$
- kluczami szyfrującymi są $k = (n, e)$ oraz $l = (n, d)$.

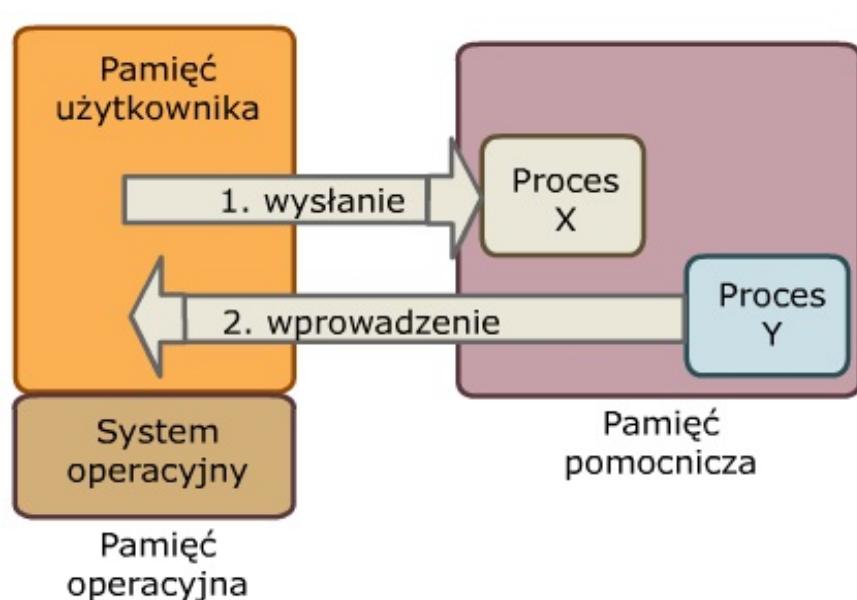
Dysponując kluczami szyfrujemy wiadomość A .

W algorytmie RSA liczby n oraz k lub l można wyjawiać bez poważnego narażenia bezpieczeństwa. Do złamania szyfru potrzebna jest funkcja $\phi(n)$, zaś do jej znalezienia musimy rozłożyć n na czynniki pierwsze, co jest zadaniem bardzo trudnym. W praktyce stosuje się klucze o długości 512–1024 bitów. W 1999 roku rozkład na czynniki pierwsze liczby 512-bitowej zajął trzystu szybkim stacjom około siedmiu miesięcy. Dlatego klucze o takiej długości nie stanowią obecnie dobrego zabezpieczenia.

W praktyce często stosuje się połączenie metody szyfrowania symetrycznego i asymetrycznego. Ponieważ szyfracja i deszyfracja wiadomości przy użyciu klucza prywatnego jest dużo szybsza niż przy stosowaniu klucza publicznego, więc przy komunikacji dwu stron stosuje się następującą metodę.

Klucze szyfru symetrycznego przekazuje się stosując metodę szyfrowania z kluczem publicznym. Klucz taki może więc być bezpiecznie przekazany przez publiczną sieć telekomunikacyjną. Następnie wiadomości mogą być już przekazywane przy zastosowaniu szyfrowania symetrycznego.

Każdy z rozmówców generuje fragment klucza prywatnego i przed przesłaniem do partnera szyfruje go stosując metodę szyfrowania z kluczem publicznym.



KERBEROS

Podstawowym, współcześnie wykorzystywanym w rozproszonych systemach informatycznych, mechanizmem zabezpieczeń jest Kerberos.

System Kerberos został opracowany w Massachusetts Institute of Technology w latach osiemdziesiątych XX wieku. Został on zaimplementowany w wielu systemach Uniksowych i internetowych, oraz w systemie Windows od wersji Windows 2000.

Kerberos jest dwukierunkowym mechanizmem uwierzytelniania z udziałem zaufanej trzeciej strony. Protokół ten przyjmuje założenie, że transakcje zachodzące między klientami i serwerami mają miejsce w sieci otwartej, czyli w środowisku, gdzie większość klientów i serwerów nie jest fizycznie zabezpieczona, a przesyłane pakiety sieciowe mogą być z łatwością monitorowane i modyfikowane. Kerberos zaprojektowany jest dla środowiska przypominającego dzisiejszą sieć Internet, w której niepowołane osoby z łatwością mogą podszywać się pod klienta lub pod serwer oraz mogą przechwycić lub przekształcić dane przesłane w ramach połączenia między legalnymi klientami i serwerami.

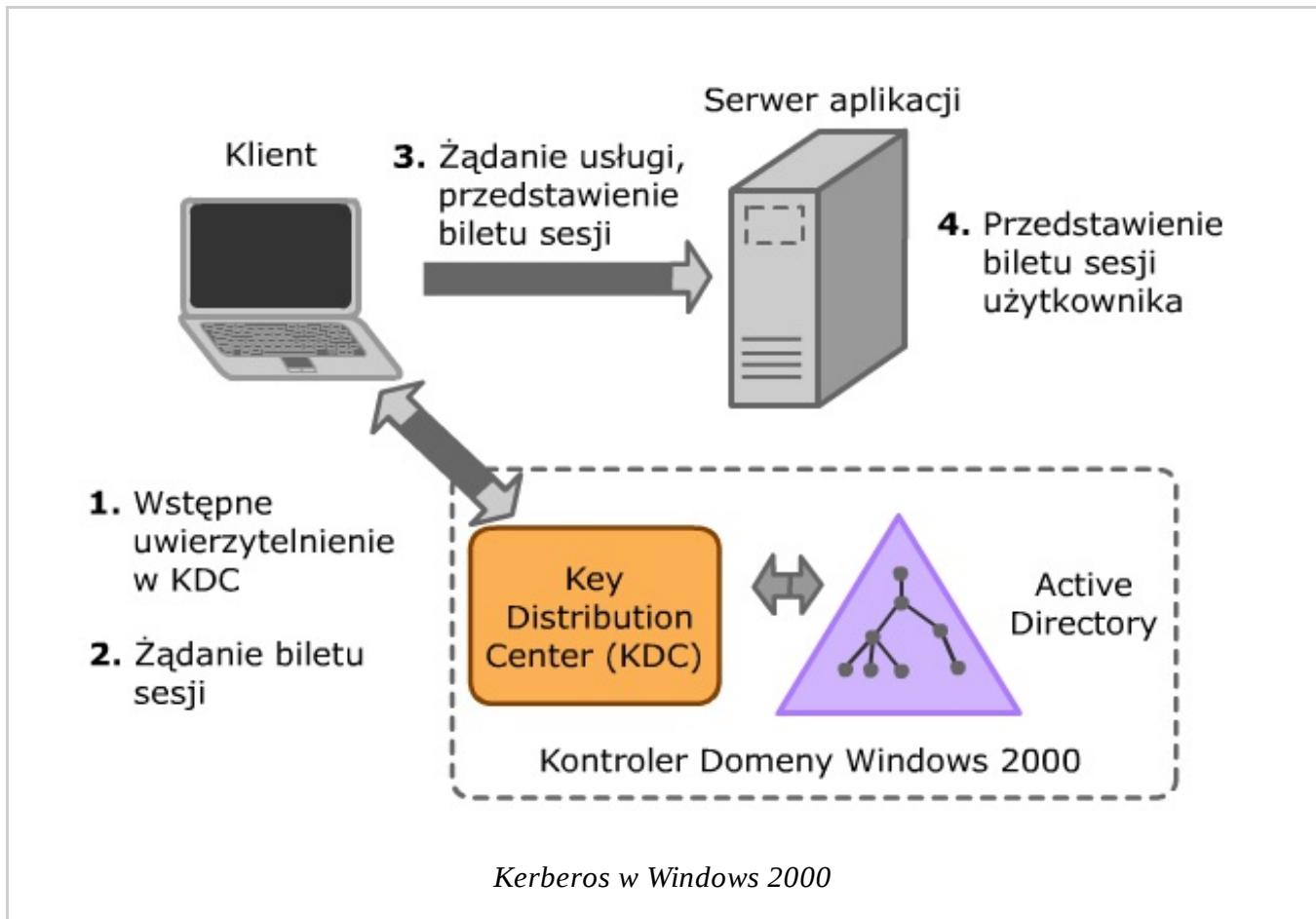
Kerberos stosuje do utajniania informacji algorytm szyfrowania symetrycznego DES (algorytm z kluczem tajnym), który jest najlepiej znany i najszerzej stosowanym w świecie algorytmem kryptograficznym. Podczas używania systemu DES, należy zwrócić uwagę na kilka czynników, które mają wpływ na bezpieczeństwo szyfrowanych danych. Należy często zmieniać klucze kryptograficzne, aby uniknąć ataku przez przedłużoną analizę danych oraz nadawca i odbiorca komunikacji muszą znaleźć bezpieczną metodę przekazywania sobie nawzajem kluczy DES. Kerberos zapewnia mechanizmy gwarantujące bezpieczną wymianę kluczy oraz pozwala na ograniczanie czasu ich ważności, przez co wymusza ich częste zmiany.

W systemie Kerberos wyróżnić można cztery podstawowe jednostki:

- klient – stacja robocza;
- serwer uwierzytelniający AS (ang. *Autentication Server*);
- serwer przyznawania biletów TGS (ang. *Ticket Granting Server*);
- serwer aplikacji AP (ang. *Application Server*).

Procedura uwierzytelniania wymaga wymiany informacji trzech typów (tzw. transakcji). Pierwsza wymiana zachodzi między klientem i serwerem uwierzytelniającym AS. Zadaniem tej wymiany jest sprawdzenie przez serwer AS autentyczności użytkownika. Uwierzytelniony użytkownik otrzymuje informację niezbędną do zainicjowania wymiany z serwerem przyznawania biletów TGS. Druga wymiana zachodzi między klientem i serwerem TGS. W jej wyniku użytkownik otrzymuje bilet uprawniający do dostępu do określonego serwera usług AP. Trzecia wymiana zachodzi między klientem i serwerem AP. Użytkownik za pomocą biletu otrzymanego od serwera TG może udowodnić serwerowi AP, że jest legalnym użytkownikiem.

W systemie Windows 2000 usługa uwierzytelniania (AS) oraz usługa przydzielania biletów (TGS) zaimplementowane są jako jedna usługa KDC (ang. *Key Distribution Center*). Usługa KDC znajduje się na każdym kontrolerze domeny (czyli komputerze, który zarządza grupą komputerów tworzących tak zwaną domenę) i korzysta z usługi Active Directory (centralna baza kontrolera domeny) jako bazy danych przechowujących konta podmiotów zabezpieczeń wraz z dodatkowymi informacjami.



Podstawowym elementem systemu Kerberos jest żeton. Jest to rodzaj certyfikatu, który jest wydawany przez usługę Kerberosa jako potwierdzenie, że wyrażono zgodę na ustanowienie sesji.

Kiedy komputer PC chce uzyskać dostęp do informacji przechowywanych na serwerze w obrębie tej samej domeny, musi przejść przez proces weryfikacji tożsamości w sieci. W praktyce ta weryfikacja tożsamości może zostać podzielona na sekwencję zdarzeń zachodzących pomiędzy klientem a KDC.

Kerberos stosuje komunikaty do dostarczenia każdemu z komponentów systemu Kerberos koniecznych informacji dotyczących tego co zaszło podczas procesu weryfikacji tożsamości. Jak już wspomniano, wiele z tych komunikatów jest szyfrowanych i zawiera znaczniki czasu, które zapobiegają możliwości przechwycenia pakietu za pomocą tradycyjnych urządzeń przechwytyjących i powtórнемu wykorzystaniu go w późniejszym czasie, bez zauważenia czegokolwiek podejrzanego przez mechanizm zabezpieczeń.

Poniżej opisana została typowa sekwencja czynności realizowanych podczas weryfikacji tożsamości klienta na określonym serwerze (lub usłudze):

1. Klient przesyła do KDC zwyczajny komunikat tekstowy, prosząc o żeton umożliwiający wymianę informacji z KDC. Komunikat przesyłany przez klienta zawiera nazwę użytkownika, nazwę serwera KDC lub usługi oraz znacznik czasu.
2. KDC przesyła klientowi zaszyfrowaną odpowiedź. Ten komunikat jest szyfrowany za pomocą hasła klienta i zawiera klucz sesji opatrzony znacznikiem czasu, który będzie wykorzystywany przy wymianie informacji z KDC, oraz żeton TGT, który klient może wykorzystać do uzyskania kolejnych żetonów umożliwiających korzystanie z wybranych usług w obszarze dziedziny KOC.
3. Klient przesyła KDC zaszyfrowany komunikat, w którym prosi o prawo do komunikowania się z wybranym serwerem lub usługą. Klient szyfruje ten komunikat za pomocą klucza sesji otrzymanego od KDC. Komunikat zawiera nazwę serwera lub usługi, z której klient chce korzystać, znacznik czasu oraz żeton TGT. Gdy KDC otrzymuje komunikat, może mieć pewność, że pochodzi on od właściwego klienta, gdyż komunikat jest deszyfrowany za pomocą jego

klucza sesji. Następnie KDC tworzy wspólny klucz sesji, który jest wykorzystywany zarówno przez klienta, jak i przez serwer. KDC tworzy także specjalny żeton dla serwera, który zawiera klucz sesji, nazwę klienta, adres jego karty sieciowej, okres ważności żetona oraz znacznik czasu.

4. KDC przesyła klientowi komunikat zawierający zaszyfrowany wspólny klucz sesji oraz zaszyfrowany żeton. Wspólny klucz sesji jest szyfrowany za pomocą klucza sesji posiadanego przez klienta, natomiast żeton - za pomocą klucza sesji posiadanego przez serwer.
5. Klient przesyła do serwera komunikat informujący, że ma prawo do komunikowania się z nim. (To, czy kolejne prośby serwera zostaną wykonane, zależy oczywiście od systemu bezpieczeństwa serwera). Komunikat zawiera zaszyfrowany żeton, który klient otrzymał od KDC, oraz czasowy identyfikacyjny znacznik czasu zaszyfrowany za pomocą wspólnego klucza sesji. Serwer deszyfruje nadesłany żeton za pomocą swojego własnego hasła. Żeton ten zawiera kopię wspólnego klucza sesji oraz kilka innych bardzo istotnych informacji dotyczących klienta. Serwer używa wspólnego klucza sesji do odszyfrowania identyfikacyjnego znacznika czasu, aby określić, kiedy klient wysłał komunikat. jeśli komunikat został wysłany w okresie ważności żetona oraz jeśli wszystkie inne warunki zostaną poprawnie spełnione, to serwer zaakceptuje prośbę klienta.
6. Po zatwierdzeniu klienta serwer przesyła zaszyfrowany komunikat, informując klienta o udzieleniu zezwolenia na komunikację. Ten komunikat zawiera identyfikacyjny znacznik czasu, który klient przesłał na serwer w kroku 5. Identyfikacyjny znacznik czasu zaszyfrowany jest za pomocą wspólnego klucza sesji.

PODSUMOWANIE

Współczesne systemy informatyczne podlegają wielu zagrożeniom zarówno wewnętrzny jak i zewnętrzny. Konieczne stało się więc zaimplementowanie mechanizmów ochronnych, pozwalających w skuteczny sposób zabezpieczyć system i gromadzone w nim informacje przed celowym lub przypadkowych uszkodzeniem, utratą bądź kradzieżą danych. Większość systemów operacyjnych dostarcza odpowiednich mechanizmów do realizacji polityki ochrony. Polityka ta jest równie ważna jak stosowane zabezpieczenia systemów. Należy więc ścisłe przestrzegać ustalonych zasad. Podstawowe reguły bezpieczeństwa dotyczą wszystkich systemów od pojedynczego domowego stanowiska aż po wielkie systemy korporacyjne. Oczywiście poziom zabezpieczeń jest w każdym wypadku inny , ale nawet w swoim domowym komputerze warto jest dbać o wykonywanie kopii zapasowych i ochronę antywirusową.

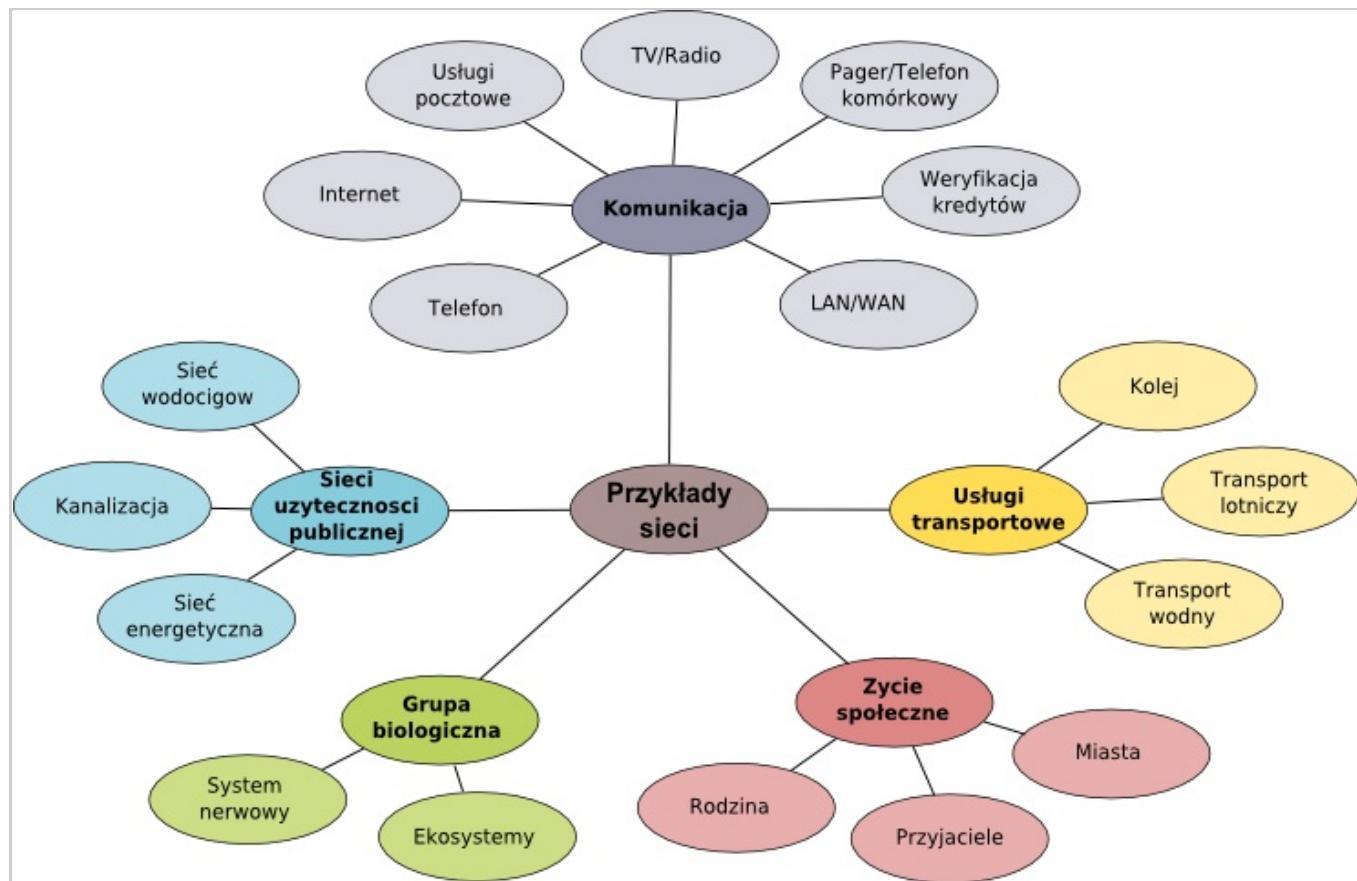
Wraz z rozwojem Internetu i usług sieciowych wzrasta również konieczność bezpiecznego przesyłania danych. Dzięki rozwojowi kryptografii i technik matematycznych możliwe jest dziś prawie bezpieczne komunikowanie się w sieci publicznej. Najpopularniejszymi metodami stosowanymi w tego rodzaju wymianie danych jest szyfrowanie symetryczne lub asymetryczne oraz wykorzystanie certyfikatów i podpisów elektronicznych.

Niemniej jednak nie ma (i raczej nie będzie) metody 100% zabezpieczenia systemów informatycznych. Dlatego też te naprawdę kluczowe systemy (jak np. układy sterowania instalacji przemysłowych) zabezpiecza się poprzez odłączenie ich od internetu, i budowanie wewnętrznych sieci zakładowych typu intranet. Zawsze powinniście zakładać, że włamanie może nastąpić - poleganie na zabezpieczeniach informatycznych (a szczególnie na jednym z nich) to zbyt wysokie ryzyko.

SIECI KOMPUTEROWE

Sieć to zawiłe połączony system obiektów lub ludzi.

Nie zdajemy sobie często sprawy, ile sieci otaczają nas w życiu codziennym. Jadąc do pracy korzystamy z sieci komunikacyjnej, gdy do kogoś dzwonimy, jesteśmy łączeni z rozmówcą za pomocą sieci telefonicznej a płacąc za zakupy kartą kredytową przesyłamy informacje za pomocą sieci teleinformatycznej. Ludzki system nerwowy oraz układ krwionośny to także sieci. Przykłady sieci można mnożyć, poniższy rysunek przedstawia kilka z nich, pogrupowanych logicznie.



Przykłady sieci występujących w naszym otoczeniu.

Sieci jako takie doczekały się też wielu prac naukowych - zajmuje się nimi zarówno matematyka, jak i nauki przyrodnicze czy społeczne. My w tym podręczniku skupimy się na sieciach komputerowych.

Ich historia zaczyna się niewiele później niż historia samych komputerów. Pierwszym wyzwaniem było przesłanie informacji cyfrowej na odległość. Pierwsze terminale dostępowe do ówczesnych komputerów korzystały z różnego rodzaju łącz szeregowych, które charakteryzowały się dwoma cechami:

- przesyłany sygnał nie był kodowany - więc na nośniku (kablach) pojawiała się fala prostokątna, wyjątkowo mało odporna na zakłócenia,
- łącze szeregowe zazwyczaj służy do komunikacji typu punkt - punkt, jest to więc linia a nie sieć.

W latach 50 XX-go opracowywane były teoretyczne podstawy działania sieci pakietowych (opartych na transmisji małych porcji danych pomiędzy poszczególnymi węzłami sieci - dzięki czemu uzyskiwano możliwość współdzielenia pasma przenoszenia pomiędzy wielu użytkowników z jednej strony, a z drugiej możliwość automatycznej komunikacji między dwoma urządzeniami cyfrowymi połączonymi siecią składającą się z wielu segmentów).

W latach 60 eksperymentowano z zapewnianiem transmisji danych cyfrowych na duże odległości - wtedy też powstały pierwsze cyfrowe połączenia na odległości liczone w kilometrach. Równolegle powstawały teoretyczne opracowania dotyczące budowy rozproszonych sieci komunikacyjnych, także w wariantach cyfrowych, zdolnych na przetrwanie przewidywanej wtedy wojny atomowej. Badania te były głównie prowadzone w USA w ramach agencji ARPA. Ta sama agencja przyczyniła się do utworzenia pierwszych działających sieci pakietowych, opartych o transmisję danych w formie ramek / pakietów. W ten sposób w 1969 roku uruchomiono w Kalifornii pierwsze działające połączenie sieciowe pomiędzy czterema komputerami - uważane dziś jako pierwsze działające połączenie internetowe.

Motywacja tych prac była dwójaka - z jednej strony chodziło o zapewnienie przetrwania infrastruktury USA w wypadku wojny atomowej, z drugiej jednak - chodziło również o ekonomiczne aspekty wykorzystania coraz popularniejszych komputerów. Zamiast wyposażyć każdy ośrodek naukowy we własne komputery - taniej było połączyć ośrodki siecią i umożliwić im korzystanie nawzajem ze swoich zasobów.

Z drugiej strony - pojawienie się tanich układów elektronicznych i zbudowanych z nich komputerów osobistych pociągnęło za sobą rozwój sieci lokalnych (protoplądów dzisiejszych sieci typu Ethernet). Szybko okazało się, że niektóre urządzenia związane, takie jak chociażby drukarki, z powodzeniem mogą być współdzielone przez kilku lub nawet kilkudziesięciu użytkowników. Wspólne dane bardziej efektywnie i bezpiecznie jest przechowywać na serwerze, do którego dostęp mają wszystkie komputery, które z tych danych muszą skorzystać. Z początku komputery osobiste łączone były w małe sieci obejmujące swoim zasięgiem jeden pokój, później całe biuro a następnie budynek.

Naturalną konsekwencją rozwoju sieci było połączenie się tych dwóch podejść dzięki czemu mamy stan dzisiejszy - globalną sieć internet, w której powoli zaczyna zacierać się różnica pomiędzy tym co lokalne a tym co dostępne zdalnie z dużej odległości. Lecz zanim do tego przejdziemy - pora przyjrzeć się, jak cyfrowa informacja jest przesyłana na odległość.

TRANSMISJA, PRZEPUSTOWOŚĆ, MEDIA TRANSMISYJNE

TRANSMISJA ANALOGOWA I CYFROWA

Transmisja sygnału w sieciach komputerowych odbywa się po łączach, nazywanych również mediumi fizycznymi. Pod względem rodzaju przenoszonego sygnału, łącza można podzielić na łącza cyfrowe i analogowe.

- **Łącza analogowe** - przenoszą sygnały ciągłe, modulowane, najczęściej o częstotliwościach z zakresu 300-3300 Hz. Ponieważ komputery operują jedynie na informacji cyfrowej, transmisja analogowa tak naprawdę odbywa się jedynie pomiędzy urządzeniami sieciowymi. Urządzenia te po każdej ze stron tłumaczą sygnał cyfrowy na analogowy i odwrotnie. Przykładem łącza analogowego jest połączenie modemowe.
- **Łącza cyfrowe** - przenoszą sygnały dyskretne, co oznacza, że mają dwa rozpoznawane przez urządzenia nadawcze i odbiorcze poziomy napięcia sygnału odpowiadające odpowiednio wartościom 0 i 1 lub -1 i 1 w zależności od standardu. Wszystkie sygnały, które nie będą odpowiadały tym dwóm sygnałom z odpowiednim marginesem tolerancji, uznane zostaną za zakłócenia i odrzucone. Przykładem połączenia cyfrowego jest transmisja karty sieciowej komputera.

PRZEPUSTOWOŚĆ ŁĄCZA

Z naszego punktu widzenia – jako użytkowników, najbardziej interesuje nas kilka parametrów łącza głównym z nich jest jego przepustowość. Przepustowość łącza (ang. *bandwidth*) określana również jako pasmo oraz bardzo często potocznie nazywana „szybkością łącza” definiuje jego możliwości. Jest to miara tego, jak wiele informacji może przepłynąć między dwoma punktami łącza w danym okresie czasu. Elementarną jednostką informacji jest bit (przyjmująca wartość „jest” lub „nie ma”), jednostkową jednostką czasu jest sekunda a więc przepustowość łącza zwykle określana jest w „bitach na sekundę”. Ponieważ dzisiejsze łącza są miliony razy szybsze, niż 1 bit na sekundę, przyjęto poniższe oznaczenia w celu określenia przepustowości łącza.

Jednostka pasma	Skrót	Równowartość	Przykład
Bit na sekundę	b/s *	1 b/s = podstawowa jednostka pasma	Nie istnieje w rzeczywistych łączach
Kilobity na sekundę	Kb/s	1 Kb/s = 1 000 = 10^3 b/s	Łącza modemowe starej generacji miały przepustowość rzędu 5 kb/s
Megabity na sekundę	Mb/s	1 Mb/s = 1 000 000 = 10^6 b/s	Standardowa sieć lokalna w naszych czasach ma przepustowość 100 Mb/s
Gigabity na	Gb/s	1 Gb/s = 1 000 000 000 b/s =	Sieci lokalne najnowszej generacji zwane Gigabit Ethernet przesyłają

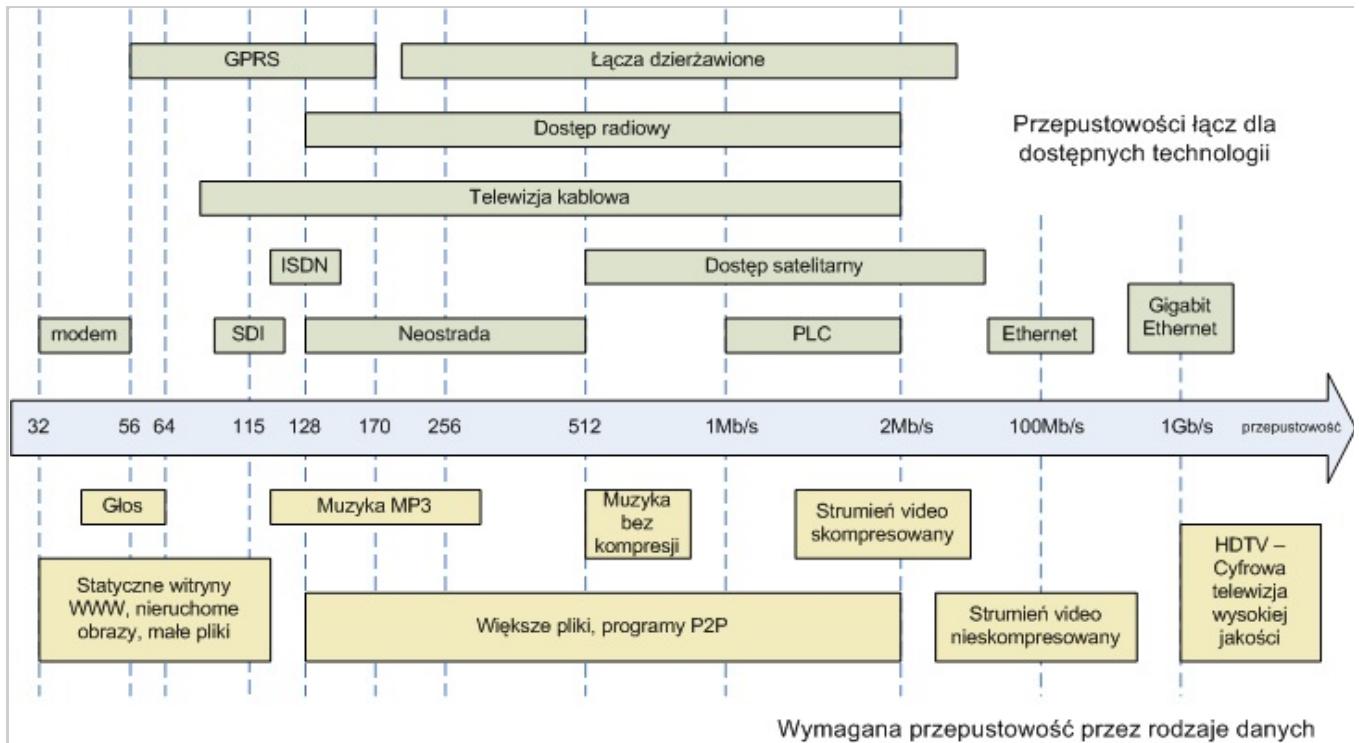
sekundę		10^9 b/s	dane z prędkością 1 do 10 Gb/s
Terabit na sekundę	Tb/s	$1 \text{ Tb/s} = 1\ 000\ 000\ 000 \text{ b/s} = 10^{12} \text{ b/s}$	Przepustowości rzędu terabitów na sekundę to standard w sieci szkieletowej. Za pomocą najnowszych technologii światłowodowych naukowcom udało się przesłać sygnał pasmem rzędu kilkuset terabitów na sekundę.

* bity zwykle oznaczane są za pomocą małej litery „b”, bajty za pomocą wielkiej litery „B”. Oznaczenie b/s zastępowane jest czasem symbolem bps (ang. *bits-per-second*).

Jako użytkownicy Internetu czy innej sieci najlepiej dla nas, aby nasze łącze miało jak największą przepustowość. Jeżeli chodzi o dostęp do Internetu to oczywiście przepustowość prawie bezpośrednio przekłada się na koszt opłaty abonamentowej za łącze. Podpisując umowę z dostawcą Internetu należy ustalić najodpowiedniejszy dla nas produkt biorąc pod uwagę oferowane pasmo za określoną cenę. Pozostałymi parametrami łącza, które powinny nas interesować to przepustowość gwarantowana, oraz cecha łącza określana jako symetryczność. Przepustowość gwarantowana w przeciwnieństwie do przepustowości maksymalnej, która podawana jest zawsze w momencie definicji danego łącza określa minimalny gwarantowany w każdej chwili czasowej transfer danych. Maksymalna przepustowość łącza określa jedynie ograniczenia sprzętowe lub konfiguracyjne naszego dostawcy sygnału, nigdy nie mamy pewności, że taką szybkość transferu osiągniemy w danej chwili - niestety w naszym kraju zazwyczaj dostawcy internetu nie podają wartości tego parametru. Bardzo ważnym czynnikiem, na który nie ma właściwie wpływu przepustowość łącza ani na który nie możemy wpływać jako użytkownicy jest ogólny ruch panujący w sieci w danej chwili czasowej. Przepustowość gwarantowana daje nam pewnego rodzaju pewność, że globalny ruch w sieci nie wpłynie całkowicie na zablokowanie naszego łącza. Łącze posiada jeszcze jedną istotną np. w przypadku udostępniania danych na własnym serwerze. Łącze symetryczne charakteryzuje się jednakową przepustowością w stronę „od” i „do” naszego komputera. Oznacza to, że z jednakową szybkością możemy odbierać i wysyłać informacje. Łącze asymetryczne charakteryzuje się różną przepustowością w przypadku pobierania danych a inną w przypadku wysyłania. Z reguły maksymalna prędkość wysyłania informacji w łączach asymetrycznych jest o 2, 4 lub 8 razy mniejsza niż maksymalna prędkość pobierania danych.

Przykładem są łącza udostępniane przez dostawców telewizji kablowej, czy też popularne łącza ADSL.

Aby bardziej obrazowo przedstawić omówione zagadnienie przepustowości, poniżej znajduje się porównanie dostępnych kilka lat temu w Polsce łącz do Internetu w Polsce oraz wymagań, jakie niesie ze sobą przesyłanie pewnych określonych w swojej formie danych.



Schemat porównawczy przepustowości łącz dostępnych w Polsce i wymaganej przepustowości przez pewne formy danych.

Główna oś schematu stanowi przepustowość łącz, od najwolniejszego połączenia modemowego o prędkości 32 kb/s do zaawansowanych technologii lokalnych sieci zwanych „Gigabit Ethernet” o przepustowości 1 Gb/s. Wymagana przepustowość na dane jest w niektórych przypadkach przedstawiona orientacyjnie, to znaczy, że gdy mamy do czynienia z dźwiękiem czy strumieniem wideo, określona przepustowość na sekundę jest wymagana, aby przekaz był pełny i zrozumiały. Inaczej jest w przypadku plików, witryn WWW i pozostałych danych, które właściwie nie stanowią pewnego rodzaju transmisji „na żywo”. W tym przypadku otworzyć witrynę czyściągnąć plik można zawsze, liczy się natomiast czas trwania tej operacji.

Aby bardziej uzmysłowić różnice opisanych łącz zastanówmy się, jak długo ściągałby się pewien teoretyczny plik. Rozważmy średniej wielkości plik zawierający utwór muzyczny skompresowany za pomocą algorytmu mp3. Kompresja ta pozwala orientacyjnie zwiększyć jedną minutę utworu w sensownej jakości w objętości 1 MB (jednego megabajta). Zważywszy, że średni czasu trwania utworu muzycznego możemy przyjąć na jakieś 4 minuty, nasz plik będzie miał rozmiar około 4 MB ($4 \text{ MB} * 1024 * 8 = 32\,768\,000 \text{ b}$). W poniższej tabeli znajdziemy czasy transmisji rozważanego pliku dla różnych dostępnych w tej chwili dla zwykłego użytkownika łącz.

Rodzaj połączenia	Przybliżony czas ściągania pliku
Modem (56 kb/s)	9,5 min (571 s)
SDI (115 kb/s)	4,6 min (278 s)
ISDN (128 kb/s)	4,2 min (250 s)
GPRS (170 kb/s)	3,1 min (188 s)
Neostrada (512 kb/s)	1 min (62 s)
Łącza telewizji kablowych (2 Mb i	mniej niż 16 s

więcej)

CZAS OPÓZNIEŃ

Jednak przepustowość w przypadku sieci to nie wszystko - jest jeszcze jeden istotny parametr sieci komputerowych, szczególnie w przypadku sieci wykorzystywanych do sterowania urządzeń, czy też w grach komputerowych.

Jego znaczenie może Wam uzmysłosić następujący przykład: Jak myślicie, jakie technicznie dostępne łącze zapewnia nam największą przepustowość w komunikacji transatlantyckiej? Wbrew pozorom, nie jest to ani łącze satelitarne, ani światłowody, tylko ... ogromny kontenerowiec wyładowany nagranymi płytami BluRay. Płynie co prawda kilka dni, ale za to jakie ilości danych przewozi w tym czasie ... ;)

NOŚNIKI DANYCH

Nośniki danych są częścią najniższej warstwy używanej do transmisji danych. Wszelkie dane są reprezentowane odpowiednio, na miedzianych przewodach za pomocą impulsów elektrycznych, nazywanych napięciem lub na światłowodach za pomocą impulsów świetlnych.

Do głównych typów kabli stosowanych głównie w sieciach lokalnych należą:

- Skrętka
- Kabel koncentryczny
- Światłowód

Kabel typu **skrętka** (ang. *twisted-pair*) składa się z czterech par skręconych ze sobą cienkich izolowanych przewodów miedzianych. Dzięki skręceniu odpowiednich kabli w pary wykorzystuje się efekt redukowania zakłóceń elektromagnetycznych (EMI) oraz częstotliwości radiowych (RFI) wywoływanych przez przepływ impulsów elektrycznych w każdym kablu pary. Pozwala to na ograniczenie degradacji sygnału. Aby efekt ten zaistniał, kabel musi być wykonany precyzyjnie według specyfikacji, która mówi o tym, jak wiele skrętów lub splotów jest dozwolone na określonym odcinku kabla.

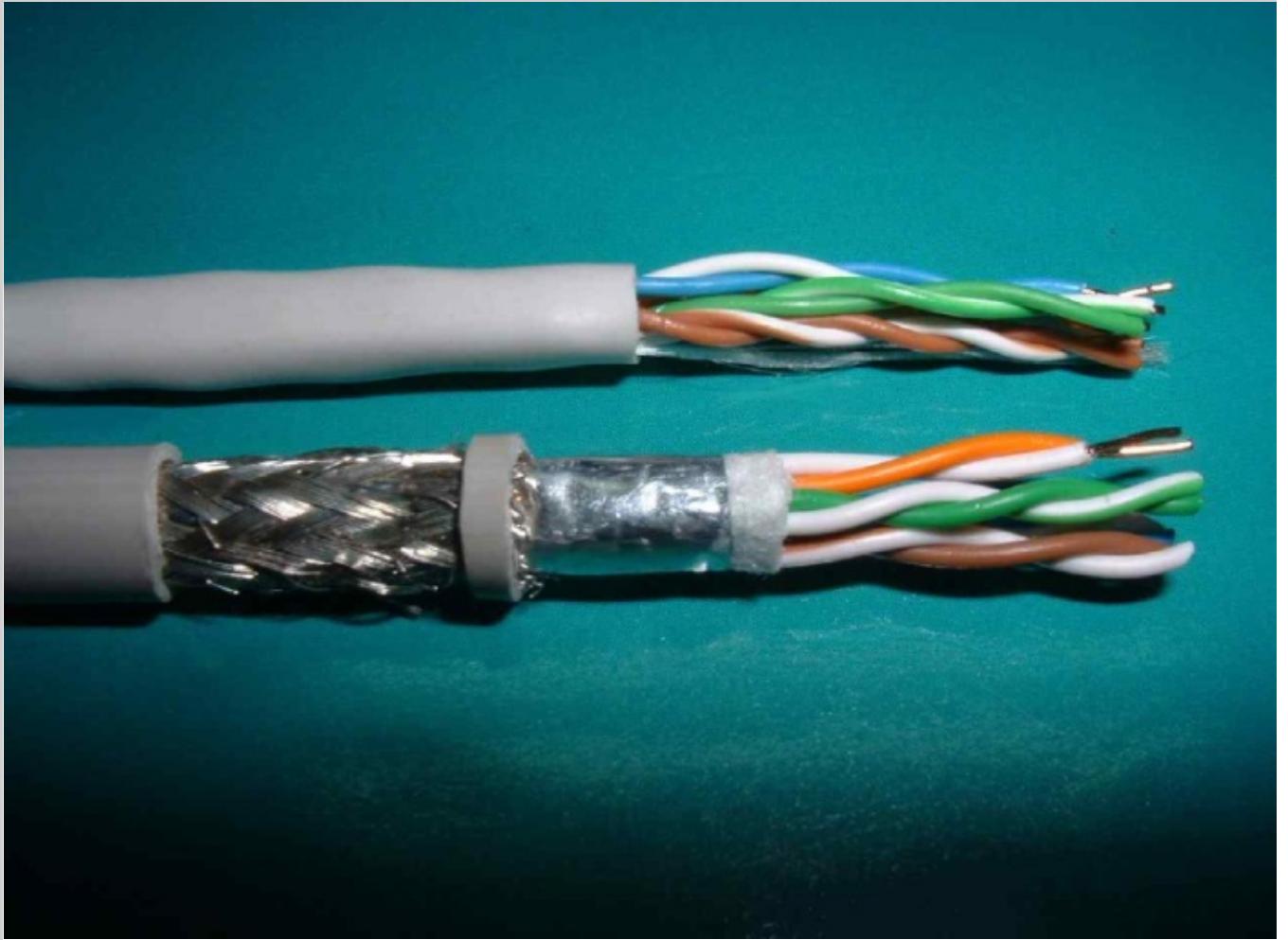
W różnych architekturach sieci i dla różnych wymagań dotyczących transmisji danych wykorzystuje się kilka odmian tego typu kabli.

Skrętka nieekranowana UTP (ang. *unshielded twisted-pair*) jest najprostszą wersją tego typu kabla. Składa się jedynie z 4 par przewodów, każdy z nich ma cienką plastikową izolację, całość otoczona jest nieco grubszą wspólną izolacją. Kabel ten ma wiele zalet, między innymi ze względu na prostotę jest tańszy od bardziej wyrafinowanych rozwiązań tego typu, z uwagi na brak dodatkowych warstw ekranujących jest cieńszy, lżejszy i bardziej giętki co ma niebagatelne znaczenie podczas instalacji. Kabel UTP może być wykorzystywany w większości architektur sieciowych i z tego powodu jest dominującym nośnikiem w sieciach LAN. Wadą skrętek UTP jest ich większa podatność na szum i zakłócenia w porównaniu do innych mediów sieciowych a dopuszczalna maksymalna odległość między źródłami sygnału mniejsza niż w porównaniu do kabla koncentrycznego czy światłowodu.

Skrętka ekranowana STP (ang. *shielded twisted-pair*) łączy w sobie technikę redukowania zakłóceń oraz ekranowania. Każda para skręconych przewodów dodatkowo otoczona jest metalową folią, wszystkie 4 pary otoczone są wspólną metalową elastyczną osłoną lub folią. Skrętka typu STP redukuje szum elektryczny powstający wewnątrz kabla (przesłuch) przez zastosowanie ekranów par oraz zakłócenia poza kablem wywoływanie przez inne kable i urządzenia elektryczne dzięki zastosowaniu zewnętrznego ekranu. STP ma wiele zalet jak również wad. Dużo lepiej ekranuje zakłócenia, dzięki

czemu może być wykorzystywana na dłuższych odległościach niż standardowa skrętka UTP. Z powodu dodatkowych warstw ekranujących jest jednak droższa oraz mniej elastyczna i grubsza, przez co trudniejsza w instalacji.

Pewnym kompromisem pomiędzy skrętką UTP i STP jest tak zwana **skrętka foliowana FTP** (ang. *foil twisted-pair*). W tym rodzaju kabla występuje jedynie zewnętrzny ekran wspólny dla wszystkich par przewodów w postaci metalowej elastycznej folii.



Skrętka nieekranowana UTP (na górze) oraz ze wspólnym ekranem dla wszystkich par u dołu.



Złączka RJ45 do kabli typu skrętka.

Kabel koncentryczny składa się z zewnętrznej, cylindrycznej osłony przewodzącej zwanej masą, która otacza pojedynczy, wewnętrzny przewód. Te dwa przewody oddzielone są warstwą izolacji tak samo jak cały kabel. W centrum kabla znajduje się przewód miedziany, natomiast zewnętrzna odizolowana warstwa to zwykle siatka druciana lub metalowa folia, działająca jako drugi przewód w obwodzie. Zewnętrzna warstwa działa jako ekran pomagający zredukować zakłócenia zewnętrzne. Bardzo popularny swojego czasu nośnik informacji w sieciach lokalnych ma tę przewagę nad skrętką, że można go wykorzystywać na kilkakrotnie większych odległościach. Wadą jest tylko jeden obwód, który możemy wykorzystać do transmisji danych (w przypadku skrętki mamy kilka par przewodów). Z czasem zaczęto odchodzić od tego rodzaju kabla na rzecz skrętki przy projektowaniu sieci. Powrót do łask przyniosła mu technologia wykorzystania istniejącego okablowania telewizji kablowych do podłączenia sieciowego. Transmisyja danych cyfrowych odbywa się na tym samym kablu, przez który dociera do nas analogowy sygnał telewizyjny. Możliwe jest to dzięki zastosowaniu różnych pasm częstotliwości oraz specjalnych urządzeń.

Kable koncentryczne różnią się grubością centralnego przewodu miedzianego oraz dokładnością wykonania masy. Parametry te przekładają się prawie bezpośrednio na maksymalną możliwą długość połączenia kablem bez utraty danych.



Dwa rodzaje kabla koncentrycznego. Końcówki kabla wykorzystywane w sieciach komputerowych niewiele różnią się od końcówek antenowych.

Kabel światłowodowy to medium sieciowe, które przenosi modulowane transmisje świetlne zamiast elektrycznych jak w przypadku poprzednich nośników. Nośnik ten jest droższy od innych typów mediów sieciowych, ma jednak kilka bardzo znaczących przewag. Cechują go dużo lepsze współczynniki przekazy danych w porównaniu z innymi mediami, zarówno jeżeli chodzi o odległości jak i przepustowość. Dodatkowo rodzaj przenoszonych impulsów (światło) jest nieczuły na wszelkie zakłócenia, jak również nie możliwe jest podsłuchanie w prosty sposób transmisji świetlnej. Sygnały elektryczne wysyłane z komputera zamieniane są przez specjalny konwerter na impulsy świetlne, które następnie przesyłane są przez kabel światłowodowy. Z drugiej strony przewodu, impulsy świetlne zamieniane są z powrotem na sygnał elektryczny rozpoznawany przez system komputerowy lub urządzenie sieciowe.

Komunikacja światłowodowa opiera się na kilku wynalazkach XX wieku. W latach sześćdziesiątych ubiegłego wieku wprowadzono w życie źródła światła laserowego oraz wysokiej jakości szkło wolne od zanieczyszczeń, dopiero wtedy komunikacja światłowodowa stała się możliwa do zastosowania. Wykorzystanie tej technologii na ogólnoświatową skalę zapoczątkowały firmy telekomunikacyjne, które zdały sobie sprawę z korzyści, jakie zapewnia w dziedzinie komunikacji na duże dystanse.

Kabel światłowodowy używany w przemyśle sieciowym składa się z dwóch włókien zamkniętych w oddzielnych osłonach. Oglądając ten kabel w przekroju można zauważyć, że warstwy ochronnego materiału zabezpieczającego, zazwyczaj plastiku (np. włókno Kevlara) oraz powłoka zewnętrzna, otaczają każde włókno optyczne. Powłoka zewnętrzna zabezpiecza cały kabel. Zazwyczaj wykonana jest z plastiku i spełnia normy przeciwpożarowe i architektoniczne. Zadaniem włókna Kevlara jest zapewnienie dodatkowej ochrony i zabezpieczeń dla szklanego włókna grubości włosa.

Części przewodzące światło w światłowodzie nazywane są rdzeniem i powłoką. Rdzeń to zazwyczaj czyste szkło o wysokim współczynniku załamania. Jeśli rdzeń jest otoczony powłoką wykonaną ze szkła lub plastiku i niskim współczynniku załamania, światło nie może wydostać się poza rdzeń włókna. Proces ten nosi nazwę całkowitego odbicia wewnętrznego. Dzięki niemu włókno optyczne może działać jak kanał świetlny, którym światło może wędrować na ogromne odległości, także pokonując zakręty.

Z uwagi na budowę światłowodu możemy wyróżnić światłowody jednomodowe i wielomodowe. Światłowód jednomodowy charakteryzuje się mniejszą średnicą rdzenia, sygnał wytworzony przez laser półprzewodnikowy ulega niewielkim zmianom a fala rozchodzi się prawie równolegle do osi światłowodu. Dzięki temu światło jednego impulsu dociera do celu w prawie tym samy czasie. Światłowody te przeznaczone są do dalekosiąznej komunikacji ponieważ nie ma tu efektu rozmycia impulsu, które wzrasta wraz z odległością, jaką musi przebyć światło.

Światłowód wielomodowy ma większą średnicę rdzenia w porównaniu do jednomodowego, w jego przypadku następuje rozbicie fali wejściowej na wiele promieni, które przez to, że odbijają się pod

innym kontem od powłoki przebywają różną odległość podczas drogi do celu. Następuje zatem efekt rozmycia impulsu wraz z odlegością przebytą przez światło a zatem ograniczenie maksymalnej odległości transmisji sygnału.



Światłowód z założonymi końcówkami.

Poniższa tabela przedstawia główne cechy najpopularniejszych mediów sieciowych – maksymalną przepustowość oraz odległość.

Nośnik	Maksymalna teoretyczna przepustowość	Maksymalna fizyczna odległość
Kabel koncentryczny 50-omowy	10-100 Mb/s	185 m
Kabel koncentryczny 75-omowy	10-100 Mb/s	500 m
Kabel UTP kategorii 5	100 Mb/s – 1 Gb/s	100 m
Kabel UTP kategorii 6	10 GB/s	100 m

6a/7		
Kabel STP kategorii 5	10-100 Mb/s	250 m
Światłowód wielomodowy	1-10 Gb/s	do kilku kilometrów
Światłowód jednomodowy	1-100 Gb/s	od kilku do kilkuset kilometrów

SIECI LAN, MAN, WAN

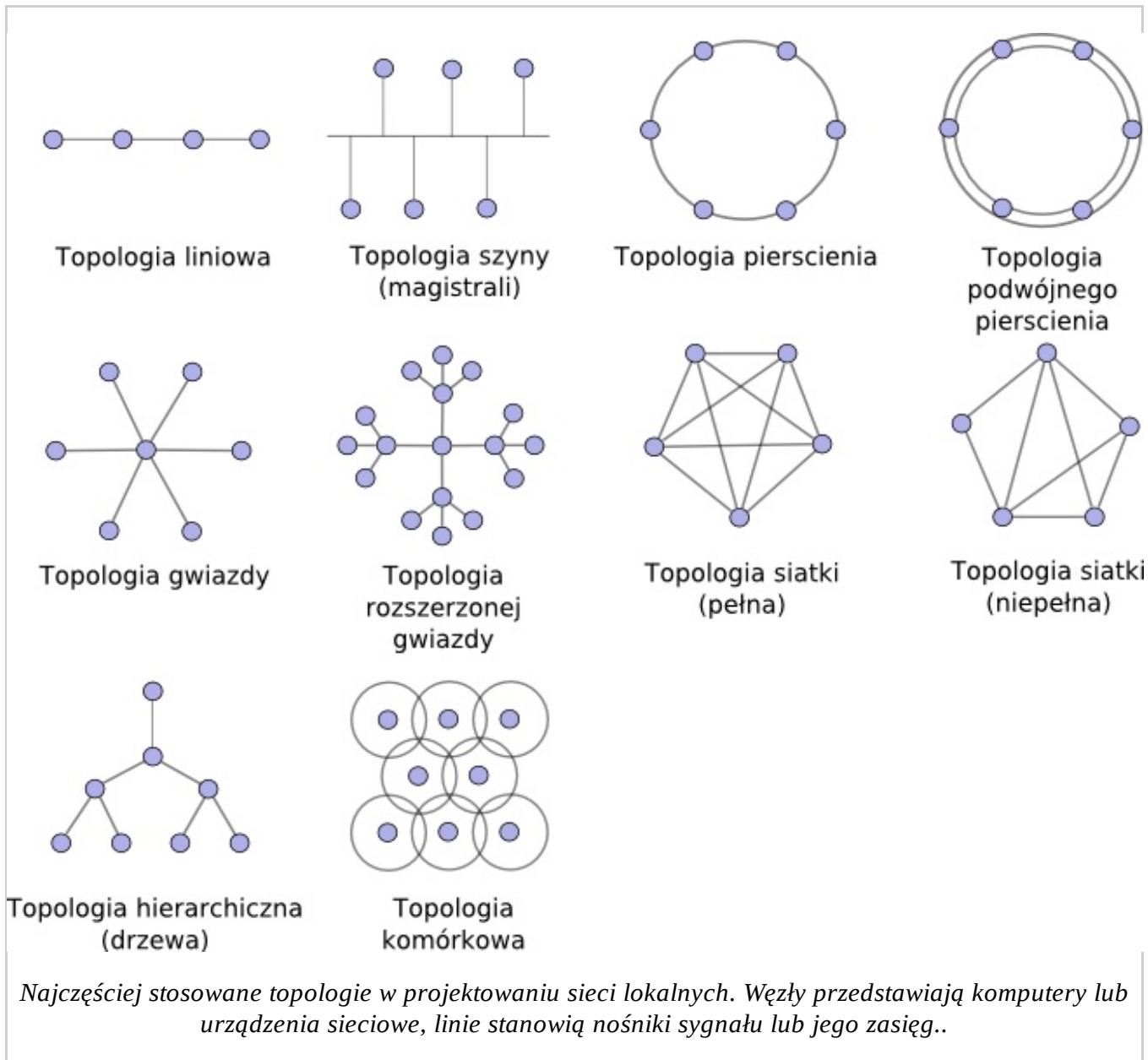
PODZIAŁ SIECI ZE WZGLĘDU NA OBEJMOWANY OBSZAR

W miarę rozwoju technologii sieciowych i coraz większe wymagania i oczekiwania odbiorców oraz narastającego wykorzystania komputerów, szybko okazało się, że sieci lokalne oznaczone skrótem **LAN** (ang. *Local Area Network*) nie są już wystarczające. W systemie LAN, każdy dział firmy czy sieć osiedlowa to swego rodzaju elektroniczna wyspa. Rosło zapotrzebowanie na metodę szybkiego i efektywnego przenoszenia informacji nie tylko w obrębie firmy, ale także między różnymi oddziałami zlokalizowanymi w znacznej odległości czy przedsiębiorstwami. Rozwiążaniem było łączenie sieci LAN z sieciami miejskimi **MAN** (ang. *Metropolitan Area Network*) a następnie sieciami rozległymi **WAN** (ang. *Wide Area Network*). Sieci WAN były sieciami obejmującymi bardzo duże obszary geograficzne, łączyły miasta i kraje.

LAN

Sieć LAN to najmniej rozległa postać sieci komputerowej, składa się z komputerów, nośników sieciowych, urządzeń sterujących ruchem oraz urządzeń peryferyjnych. Umożliwiają one efektywne współdzielenie danych i zasobów na ograniczonym obszarze geograficznym (przeważnie o średnicy maksymalnie do kilkuset metrów). Do głównych zadań sieci lokalnych należy umożliwienie korzystania z nośników o szybkim paśmie równolegle wielu użytkownikom, zapewnienie ciągkiej łączności z lokalnymi usługami (np. serwerem plików, wydruków, pocztowym) oraz łączenie fizycznie sąsiadujących ze sobą urządzeń.

Technologie stosowane w sieciach lokalnych można podzielić na rozwiązanie oparte na przewodach (kable miedziane, światłowody) lub komunikacji radiowej (bezprzewodowe). W sieciach lokalnych przewodowych najczęściej używaną technologią jest Ethernet (za pośrednictwem kart sieciowych). Czasem są to takie urządzenia, jak np. port szeregowy, port równoległy czy port podczerwieni. Sieci lokalne mogą być budowane w oparciu o różne topologie, najczęściej stosowane topologie przedstawione są na poniższym rysunku.



Topologia liniowa jest najprostszą z możliwych. Cechuje ją najmniejsze zużycie nośnika (kabla), oraz spore ograniczenia i awaryjność. W topologii tej każdy węzeł widzi może komunikować się jedynie z najbliższymi sąsiadami a aby skontaktować się z dalszymi jednostkami potrzebuje pośrednika. W przypadku awarii węzła lub zerwania połączenia w dowolnym punkcie sieć zostaje podzielona na 2 osobne segmenty.

Topologia szyny jest ulepszoną wersją topologii liniowej. Zużycie nośnika sygnału jest tu dołożo większe ponieważ każdy z węzłów musi zostać podpięty do punktu centralnego lub centralnej magistrali stanowiącej szkielet sieci. Zaletą jest to, że wszystkie węzły są ze sobą bezpośrednio połączone i mogą komunikować się ze sobą bez pośredników. W przypadku awarii węzła lub połączenia węzeł-szkielet, pozostała część sieci może nadal pracować w sposób niezakłócony. W przypadku przerwania magistrali sytuacja jest porównywalna z awarią w topologii liniowej. Wadą przy dużej ilości węzłów jest duży ruch typu rozgłoszeniowego (o którym w dalszej części lekcji).

Topologia pierścienia tworzy jeden zamknięty pierścień składający się z węzłów i łącz. Tak naprawdę można ją przyrównać do topologii liniowej z dodatkowym przewodem „zamykającym obwód”. Tak samo jak w przypadku topologii liniowej każdy węzeł komunikuje się tylko ze swoimi najbliższymi sąsiadami. Zaletą jest to, że w przypadku awarii dowolnego jednego węzła lub łączka w jednym dowolnym miejscu nie powoduje rozdzielenia sieci. Jest to zatem topologia dosyć odporna na awarie. Zużycie nośnika też jest tutaj niezbyt duże.

Topologia podwójnego pierścienia składa się z dwóch pierścieni o wspólnym środku. Pierścienie nie są połączone ze sobą w żadnym punkcie, każdy węzeł jest częścią dwóch niezależnych topologii pierścienia. W tej topologii przerwanie łącza w kilku miejscach lub awaria kilku węzłów nie musi spowodować rozdzielenia sieci na mniejsze segmenty. Jest to topologia o bardzo dużej odporności na awarie a przy okazji oszczędna w użyciu nośnika, co jest zdecydowaną zaletą w momencie utrudnionego kładzenia kabli lub ograniczonych środków.

Topologia gwiazdy to centralnie położony węzeł, do którego promieniście połączone są wszystkie pozostałe węzły sieci, nie ma innych połączeń. Topologia ta pozwala na wygodną komunikację wszystkich węzłów sieci – zawsze poprzez jednego pośrednika – węzeł centralny. Zaletą sieci jest mała awaryjność jeżeli chodzi o zrywanie poszczególnych połączeń, zawsze odłączony zostanie tylko jeden węzeł. Wadą jest całkowity brak odporności na awarię węzła centralnego, która dzieli sieć na pojedyncze komputer i urządzenie. Zużycie nośnika będzie w tym przypadku znacznie większe niż w przypadku wcześniej opisywanych technologii, chociaż zależy to oczywiście od fizycznego rozlokowania węzłów sieci i punktu centralnego. Cały przepływ informacji pomiędzy wszystkimi węzłami przechodzi przez jedno urządzenie, co daje ciekawe możliwości pod względem bezpieczeństwa czy ograniczeń dostępu.

Topologia rozszerzonej gwiazdy pochodzi od topologii gwiazdy, gdzie niektóre z pośrednich węzłów topologii działają jako centra dla własnych topologii gwiazdy. Zaletą tej topologii nad jej prostszą wersją jest to, że przewody są krótsze oraz odciążony jest węzeł centralny topologii, na rzecz którego obsługa ruchu w ramach „lokalnej gwiazdy” przejmują centra pośrednie. Topologia rozszerzonej gwiazdy ma strukturę hierarchiczną i może być konfigurowana w takie sposób, aby ruch pozostawał lokalny. Taką strukturę ma obecny system telefoniczny.

Topologia siatki w swojej pełnej formie stanowi połączenie każdego węzła z każdym. Takie graniczne podejście do połączenia węzłów w sieć ma swoje oczywiste wady i zalety i prawdopodobnie bardzo rzadko wykorzystywane jest w rzeczywistych przypadkach. Połączenie „każdy z każdym” pozwala na transmisję danych alternatywnymi ścieżkami, co może być przydatne nawet w przypadku przeciążeń na poszczególnych łączach. Każdy z węzłów ma bezpośrednie połączenie z każdym innym. Sieć ta jest całkowicie odporna na awarie, zarówno węzłów jak i połączeń między nimi. Dopiero duża liczna awaria może doprowadzić do rozdzielenia sieci na oddzielne segmenty. Wadą tej topologii jest ogrom połączeń, które trzeba wykonać przy większej liczbie węzłów. Z tego powodu stosuje się niepełną topologię siatki, gdzie każdy z węzłów ma połączenia do kilku innych węzłów. Z połączeń mniej krytycznych można zrezygnować, należy jednak tak wybrać połączenia, aby segmenty sieci nigdy nie były połączone tylko jedną nicią.

Topologia hierarchiczna, nazywana również topologią drzewa przypomina topologię rozszerzonej gwiazdy. Główną różnicą jest brak centralnego węzła, używany jest węzeł podstawowy, z którego rozchodzą się kolejne węzły. Istnieją dwa rodzaje tej topologii: drzewo binarne, gdzie każdy z węzłów ma dokładnie dwa węzły podrzędne oraz drzewo szkieletowe, gdzie węzły rozchodzą się dowolnie od pnia szkieletu. Przepływ informacji jest hierarchiczny, odporność na awarię i skala jej wpływu na sieć jest zależna od punktu awarii. Nie ma tu znaczenia, czy awarii ulegnie węzeł czy połączenie, skutki są takie same.

Topologia komórkowa możliwa jest do zbudowania jedynie za pomocą przekaźników sygnałów bezprzewodowych. W topologii tej zasięg sygnału poszczególnych węzłów musi pokrywać cały teren, który ma zostać objęty siecią. Węzły w takim przypadku połączone mogą być za pomocą kabli lub anten kierunkowych. Jeżeli węzły nie posiadają własnego okablowania, muszą komunikować się między sobą również za pomocą transmisji bezprzewodowej. W tym przypadku muszą one być rozlokowane w taki sposób, aby znajdowały się w zasięgu węzłów sąsiednich. W zależności od konfiguracji sieci, awaria jednego przekaźnika nie powinna wpłynąć na ogólne działanie sieci, może jedynie spowodować zanik sygnału w punkcie i okolicy uszkodzonego węzła. Nie istnieje tu problem z przerwaniem nośnika sygnału, wpływ na wydajność mogą mieć jednak czynniki zewnętrzne, np. atmosferyczne. Jest to jedyna topologia, która dopuszcza utrzymywanie połączenia urządzenia znajdującego się w ruchu, m.in. poprzez przełączanie obsługi połączenia pomiędzy poszczególnymi przekaźnikami. Taką topologię wykorzystują współczesne sieci telefonii komórkowej.

MAN

Sieć miejska to sieć komputerowa, której zasięg z reguły obejmuje aglomerację lub miasto. W sieciach MAN, do komunikacji pomiędzy wchodzącymi w jej skład rozrzuconymi sieciami LAN lub węzłami sieci, jako nośnika używa się najczęściej przewodów światłowodowych. Wymuszone jest to już dosyć znacznymi odległościami pomiędzy poszczególnymi węzłami sieci (od kilku do kilkudziesięciu kilometrów).

Sieci miejskie budowane są zwykle przez wszelkiego rodzaju organizacje. Przykładem wykorzystania takich sieci miejskich są ośrodki akademickie Uczelni wyższych, które oprócz okablowania lokalnego dla każdego z budynków, połączone są szybkimi łączami stanowiącymi pewnego rodzaju szkielet sieci oraz najczęściej w centralnym miejscu połączone są do Internetu lub innych podobnych sieci. Do technologii używanych przy budowaniu takich sieci należą ATM, FDDI, SMDS oraz ostatnio Gigabit Ethernet. Tam gdzie niemożliwe jest użycie połączeń światłowodowych często stosuje się bezprzewodowe połączenia radiowe, laserowe lub podczerwone.

WAN

Sieć rozległa obejmuje swoim zasięgiem duży obszar geograficzny, często powierzchnię całego kraju lub nawet kontynentu. WAN łączy sieci miejskie i lokalne oraz rzadziej pojedyncze komputery. W celu zestawienia połączenia pomiędzy odległymi sieciami wykorzystywane są usługi operatorów telekomunikacyjnych. Sieci WAN używają różnego rodzaju łącz szeregowych. Przykładem sieci WAN jest Internet.

MODEL ODNIESIENIA OSI

A SPEKT HISTORYCZNY

W późnych latach osiemdziesiątych i wczesnych dziewięćdziesiątych XX wieku zaobserwowano znaczący wzrost liczby oraz rozmiarów sieci komputerowych. Wiele sieci zbudowano w oparciu o różne implementacje sprzętowe i programowe. W wyniku tego sieci były ze sobą niezgodne, a komunikacja między sieciami wykorzystującymi różne specyfikacje bardzo utrudniona. W odpowiedzi na ten problem Międzynarodowa Organizacja Normalizacyjna (ISO - ang. *International Organization for Standardization*) zbadała wiele schematów organizacji sieci. ISO zauważała potrzebę utworzenia modelu sieci, który pomógłby twórcom implementować takie sieci, które mogłyby współpracować i komunikować się ze sobą. W roku 1984 powstał model odniesienia OSI (ang. Open System Interconnection).

M ODEL ŁĄCZNOŚCI, PAKIETOWANIE DANYCH

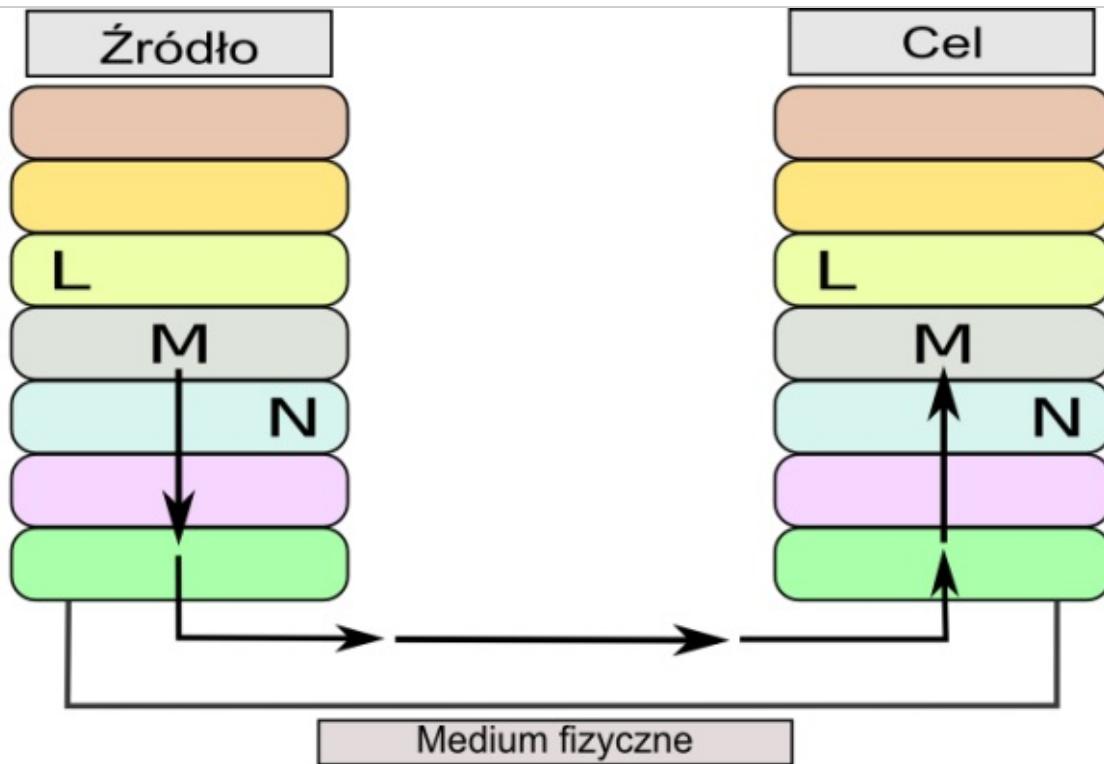
Proces łączności sieciowej jest złożony. Dane, w formacie sygnałów elektronicznych, muszą wędrować nośnikiem do właściwego komputera docelowego, po czym muszą zostać skonwertowane z powrotem do początkowej postaci, aby stały się czytelne dla odbiorcy. Proces ten składa się z kilku etapów, dlatego więc najbardziej efektywnym sposobem jego realizacji jest proces podzielony na warstwy. W takim procesie każda z warstw wykonuje konkretne zadanie.

Jak się dowiedzieliśmy w poprzednich rozdziałach, najbardziej podstawowy poziom informacji komputerowej składa się z cyfr binarnych, czyli bitów (zera i jedynki). Komputery wysyłające jeden lub dwa bity informacji nie są jednak użyteczne, dlatego konieczne są inne metody grupowania informacji (na przykład bajty, kilobajty, itd.). Aby komputery mogły wysyłać informacje w sieci, każdy proces przesyłania rozpoczyna się w źródle, a kończy w punkcie docelowym, czyli miejscu przeznaczenia. Zanim będzie można wysłać dane w postaci impulsów elektrycznych, najpierw trzeba podzielić je na możliwe do zarządzania części. Dane same w sobie nie są informacjami; to zakodowana forma informacji będąca serią elektrycznych impulsów. Informacje są tłumaczone do takiej postaci, aby mogły zostać wysłane.

Informacje wędrujące w sieci nazywane są zatem danymi, pakietami lub pakietami danych. Pakiet danych to logicznie pogrupowana jednostka informacji przemieszczająca się między systemami komputerowymi. Zawiera on informacje na temat źródła i miejsca przeznaczenia, jak również inne elementy konieczne do nawiązania łączności.

P ROTOKÓŁ

Dla pakietów wędrujących ze źródła do celu w sieci ważne jest, aby wszystkie urządzenia mówili tym samym językiem, czyli protokołem. Protokół sieciowy to zestaw zasad, który sprawia, że łączność sieciowa jest możliwa i bardziej efektywna. Techniczna definicja protokołu łączności brzmi następująco: zestaw zasad lub uzgodnień określający format i transmisję danych. Warstwa n na jednym komputerze komunikuje się z warstwą n na innym komputerze. Zasady i konwencje wykorzystywane w tym procesie komunikowania się nazywane są zbiorowo protokołem warstwy n.



- L, M, N - warstwy w modelu komunikacji komputerowej
- M-Źródło, M-Cel - warstwy równorzędne
- - komunikacja między warstwami równorzędnymi
- Protokół warstwy M - zasady, według których M-Źródło komunikuje się z M-Celem

Warstwowy model komunikacji sieciowej.

ZADANIE MODELU ODNIESIENIA OSI

Model odniesienia OSI to podstawowy model komunikacji sieciowej. Pozwala on obserwować funkcje sieci pełnione przez każdą z warstw. Co ważniejsze, model OSI to szkielet, którego można użyć, aby zrozumieć wędrówkę informacji w sieci. Ponadto model OSI można wykorzystać do wizualizacji procesu wędrówki informacji lub pakietów danych z programów aplikacji (np. arkuszy kalkulacyjnych, dokumentów, itd.) przez medium sieci, do innych programów umieszczonych na innym komputerze w sieci, nawet jeśli nadawca i odbiorca mają różne typy mediów sieciowych.

W modelu odniesienia OSI mamy do czynienia z siedmioma ponumerowanymi warstwami. Każda warstwa ilustruje konkretne funkcje sieciowe. Podział funkcji sieciowych nosi nazwę podziału na warstwy. Dzielenie sieci na siedem warstw daje następujące zalety:

- Dzieli komunikację sieciową na mniejsze, prostsze części, z którymi łatwiej pracować;
- Ułatwia standaryzację elementów sieciowych, co pozwala na korzystanie i obsługę z poziomu produktów wielu producentów;
- Umożliwia nawiązanie komunikacji pomiędzy różnymi rodzajami sprzętu i oprogramowania sieciowego;
- Zapobiega wpływom zmian dokonanych w jednej warstwie na inne warstwy, co przyspiesza ich dokonywanie;
- Dzieli komunikację sieciową na mniejsze części, dzięki czemu poznawanie ich jest znacznie łatwiejsze.

WARSTWY MODELU OSI

Proces przemieszczania się informacji między komputerami jest podzielony na siedem mniejszych, łatwiejszych w zarządzaniu etapów. Każda z siedmiu części reprezentowana jest za pomocą własnej warstwy w modelu. Oto siedem warstw modelu odniesienia OSI:

- Warstwa 7: Warstwa aplikacji (ang. *Application layer*);
- Warstwa 6: Warstwa prezentacji (ang. *Presentation layer*);
- Warstwa 5: Warstwa sesji (ang. *Session layer*);
- Warstwa 4: Warstwa transportu (ang. *Transport layer*);
- Warstwa 3: Warstwa sieci (ang. *Network layer*);
- Warstwa 2: Warstwa łącza danych (ang. *Data link layer*);
- Warstwa 1: Warstwa fizyczna (ang. *Physical layer*).

Trzy wyższe warstwy modelu (a więc warstwy 7, 6 i 5) nazywane są warstwami aplikacji. Cztery niższe warstwy modelu (a więc warstwy 4, 3, 2 i 1) definiują sposób przesyłania danych fizycznym kablem przez urządzenia sieci do końcowych stacji, i ostatecznie do warstw aplikacji.

WARSTWA 7: WARSTWA APLIKACJI

Warstwa aplikacji to najbliższa użytkownikowi warstwa modelu OSI. Dostarcza usługi sieciowe, na przykład dostęp do plików lub drukowanie, aplikacjom użytkownika. Różni się od innych warstw tym, że nie dostarcza usług pozostałym warstwom modelu OSI, ale tylko aplikacjom poza modelem.

Przykładami takich aplikacji są przeglądarki internetowe, programy peer-to-peer czy terminale bankowe. Warstwa aplikacji zapewnia dostępność do planowanych współużytkowników sieci. Synchronizuje i ustanawia porozumienie według procedur wykrywania błędów i integralności danych. Aby ułatwić sobie zapamiętanie warstwy 7, w największym skrócie można traktować ją jako przeglądarkę internetową.

Przykłady: Telnet, HTTP.

WARSTWA 6: WARSTWA PREZENTACJI

Warstwa prezentacji sprawia, że informacje wysyłane przez warstwę aplikacji jednego systemu są czytelne dla warstwy aplikacji innego systemu. Jeśli to konieczne, warstwa prezentacji dokonuje tłumaczenia wielu formatów danych wykorzystując wspólny format. Warstwa ta jest również odpowiedzialna za kompresję i szyfrowanie. Słowa, które najkrócej opisują warstwę 6 to wspólny format danych.

Przykłady: ASCII, EBCDIS, JPEG.

WARSTWA 5: WARSTWA SESJI

Jak wskazuje nazwa, warstwa sesji ustanawia, zarządza i zamyka sesje między dwoma komunikującymi się komputerami. Warstwa sesji dostarcza swoje usługi warstwie prezentacji. Synchronizuje dialog między warstwami prezentacji dwóch hostów i zarządza ich wymianą danych. Poza regulacją sesji, warstwa ta oferuje zasoby dla efektywnego transferu danych, klasę usług oraz zgłaszanie wyjątków problemów w warstwie sesji, prezentacji i aplikacji. Aby móc szybko zapamiętać warstwę sesji, można skojarzyć ją z dialogami i konwersacjami.

Przykłady: Planowanie dostępu do systemu operacyjnego/aplikacji.

WARSTWA 4: WARSTWA TRANSPORTU

Warstwa transportu dzieli na segmenty dane pochodzące z wysyłającego systemu hosta, i składa je w strumień danych w odbierającym systemie hosta. Granice między warstwą transportu a warstwą sesji można porównać od granic między protokołami aplikacji a protokołami przepływu danych. Podczas gdy warstwy aplikacji, prezentacji i sesji skoncentrowane są na zagadnieniach związanych z aplikacjami, niższe warstwy zajmują się aspektami transportu danych.

Warstwa transportu próbuje dostarczyć usługę transportu danych, przez co chroni wyższe warstwy przed zajmowaniem się szczegółami związanymi z realizacją transportu. Warstwa transportu przede wszystkim ocenia, w jakim stopniu niezawodny jest transport między dwoma hostami. Dostarczając usługę komunikacyjną, warstwa transportu ustanawia, utrzymuje i prawidłowo zamkna obwody oparte na połączeniach. Dzięki mechanizmowi niezawodności możliwe są wykrywanie i naprawa błędów transportowych oraz kontrola przepływu danych. Aby łatwo zapamiętać warstwę 4, można skojarzyć ją z kontrolą przepływu i niezawodnością.

Przykłady: TCP, UDP, SPX.

WARSTWA 3: WARSTWA SIECI

Warstwa sieci to złożona warstwa odpowiedzialna za łączność oraz wybór ścieżek między dwoma systemami hostów, które mogą rezydować w geograficznie oddzielonych sieciach. Aby zapamiętać warstwę 3, kojarzymy ją z wyborem ścieżki, routingu oraz logicznym adresowaniem.

Przykłady: IP, IPX.

WARSTWA 2: WARSTWA ŁĄCZA DANYCH

Warstwa łącząca danych odpowiedzialna jest za transport danych na fizycznym łączu. Zajmuje się fizycznym (przeciwieństwem logicznego) adresowaniem, topologią sieci (czasem nazywaną logiczną), dostępem do mediów sieciowych oraz wykrywaniem błędów. Aby zapamiętać warstwę 2, można ją skojarzyć z ramkami i kontrolą dostępu.

Przykłady: 802.3/802.2, HDLC.

WARSTWA 1: WARSTWA FIZYCZNA

Warstwa fizyczna definiuje elektryczne, mechaniczne, proceduralne i funkcjonalne specyfikacje aktywowania, utrzymywania i wyłączania fizycznego łączu między końcowymi systemami. Specyfikacje warstwy fizycznej definiują takie właściwości jak poziomy napięcia, okresy zmian napięcia, współczynniki fizycznych danych, maksymalne odległości transmisyjne, fizyczne łączniki oraz inne, podobne atrybuty. Aby zapamiętać warstwę 1, można skojarzyć ją z sygnałami i medium.

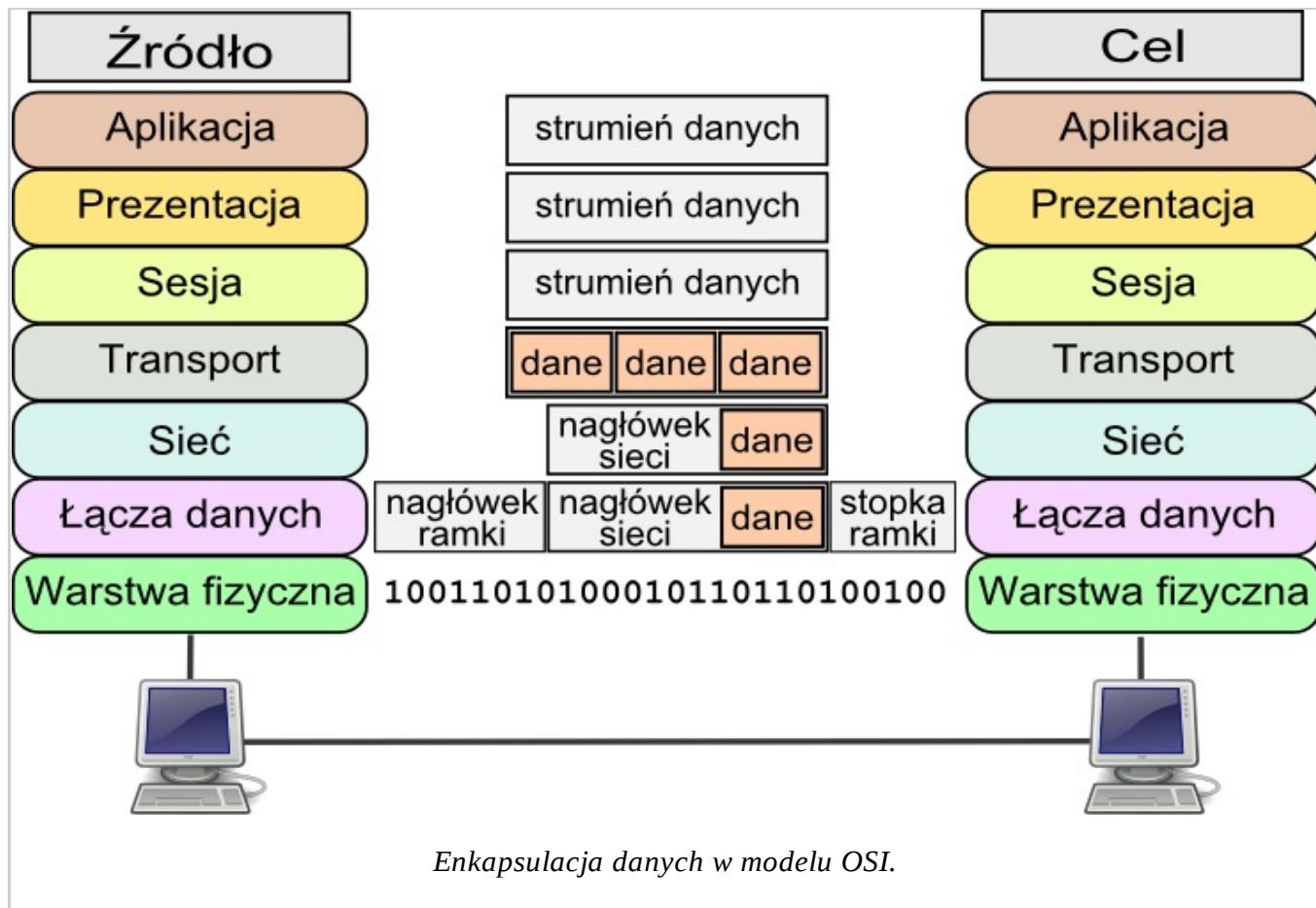
Przykłady: EIA/TIA-232, V.35.

ENKAPSULACJA

Cała komunikacja w sieci zaczyna się w źródle, a kończy w miejscu przeznaczenia. Informacje wysyłane w sieci nazywane są danymi lub pakietami danych. Jeśli jeden komputer (Host A) chce wysłać dane do innego komputera (Host B), dane najpierw muszą być spakowane w procesie nazywanym enkapsulacją. Enkapsulacja przed wysłaniem do sieci otacza dane koniecznymi informacjami na temat protokołu.

Dlatego w miarę przemieszczania się danych między kolejnymi warstwami modelu OSI, zbiera on

nagłówki, stopki i inne informacje. Aby zobaczyć jak działa enkapsulacja, zapoznajmy się ze sposobem, w jaki dane wędrują przez kolejne warstwy modelu OSI, co widać na poniżej ilustracji.



Po wysłaniu danych ze źródła, wędrują one przez warstwę aplikacji do niższych warstw. Jak widać, pakowanie i przepływ wymienianych danych zmienia się w następstwie usług wykonywanych przez warstwy.

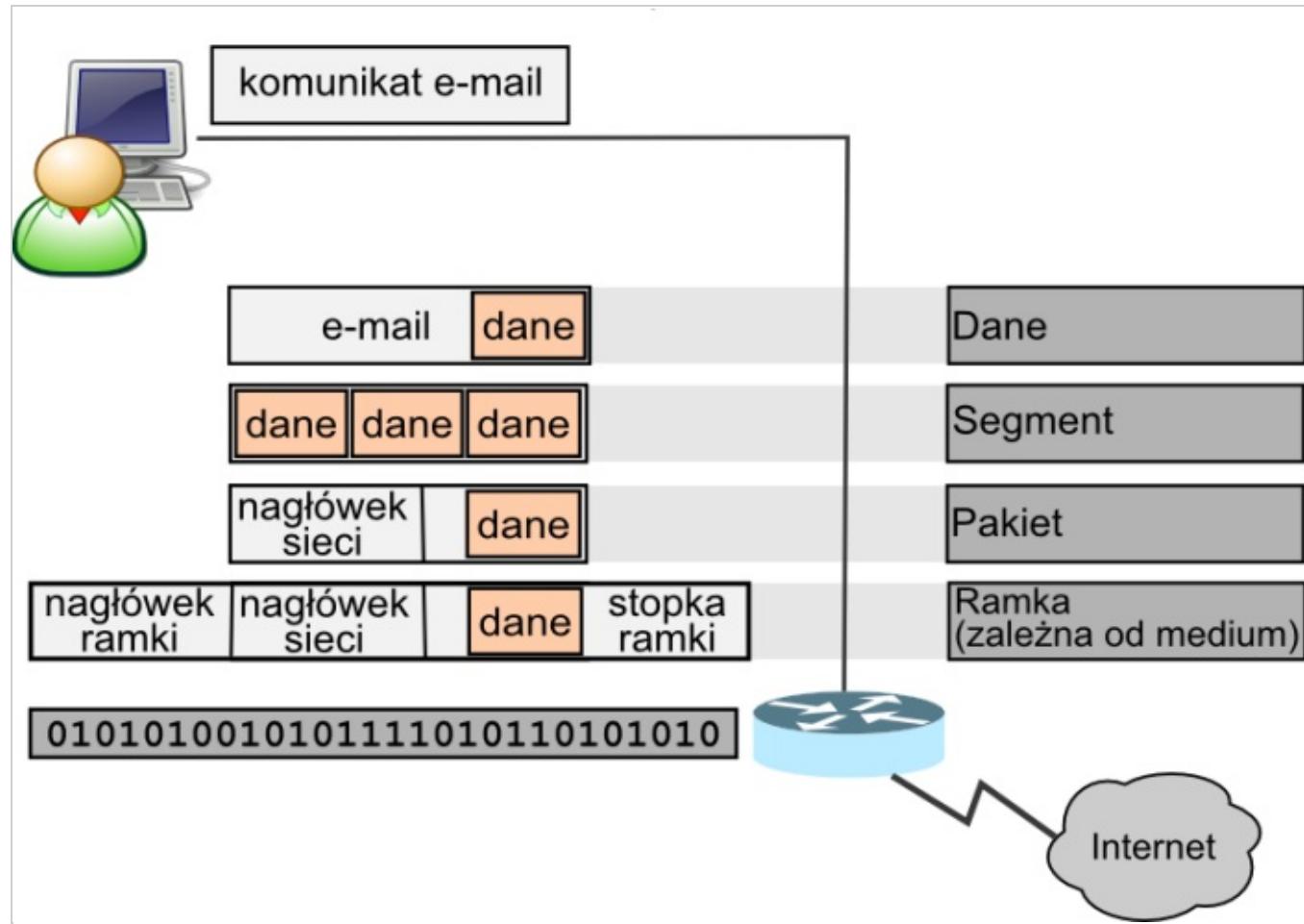
Dane, w postaci elektronicznych sygnałów, muszą wędrować kablem do właściwego komputera docelowego, a następnie są konwertowane do swojej początkowej postaci, aby były czytelne dla odbiorcy. Łatwo sobie wyobrazić, że proces ten składa się z kilku etapów. Z tego powodu twórcy sprzętu, oprogramowania i protokołów zauważyl, że najbardziej efektywnym sposobem komunikacji jest proces warstwowy.

Komputer przed wysłaniem danych za pomocą medium sieciowego musi zatem przeprowadzić 5 poniższych etapów konwersji aby dokonać enkapsulacji:

- **Budowa danych** – kiedy użytkownik np. wysyła komunikat pocztą elektroniczną, jego alfanumeryczne znaki konwertowane są do formatu danych, który może wędrować w sieci;
- **Pakowanie danych dla transportu między dwoma punktami łącza** – dane są pakowane w celu transportu w sieci rozległej. Dzięki użyciu segmentów, funkcje transportu zapewniają, że hosty po obu końcach systemu poczty elektronicznej mogą nawiązać niezawodną komunikację;
- **Dodanie adresu sieci do nagłówka** – dane są umieszczane w pakiecie, czyli datagramie zawierającym nagłówki sieci z logicznymi adresami IP źródła i celu. Adresy te pomagają urządzeniom sieciowym wysyłać pakiety w sieci dynamicznie wybieraną ścieżką;
- **Dodawanie lokalnego adresu (MAC) do nagłówka łączą danych** – każde urządzenie sieciowe musi umieścić pakiet w ramce. Ramka obejmuje nagłówek z fizycznym adresem następnego, bezpośrednio połączonego urządzenia na ścieżce;
- **Konwersja danych dla potrzeb transmisji** – ramka musi być przekształcona na ciąg jedynek i zer (bitów) dla transmisji za pośrednictwem medium (zazwyczaj kabla). Funkcja taktowania pozwala urządzeniom odróżnić te bity w miarę ich wędrówki. Medium w fizycznej sieci rozległej

może być różne na kolejnych etapach wybranej ścieżki. Na przykład, komunikat poczty elektronicznej może pochodzić z sieci LAN, przechodzić przez sieć szkieletową campusu, a następnie dojść do łącza WAN, aż osiągnie miejsce przeznaczenia w innej sieci LAN.

Powyższe działanie przedstawione zostało na poniższej ilustracji.



Dodawanie nagłówków w miarę przechodzenia po warstwach OSI.

DE-ENKAPSULACJA

W chwili gdy oddalone urządzenie odbiera sekwencję bitów, przekazuje je warstwie łączącej danych, która manipuluje ramkami. Po odebraniu ramki przez warstwę łączącej danych, wykonuje ona następujące czynności:

- Odczytuje fizyczny adres i inne informacje kontrolne dostarczone przez bezpośrednio połączoną równorzędną warstwę łączącej danych;
- Przeprowadza rozbiór informacji kontrolnych ramki, tym samym tworząc datagram;
- Przekazuje datagram w górę, do następnej warstwy, postępując zgodnie z instrukcjami znajdującymi się w kontrolnej części ramki;

Proces ten nazywany jest de-enkapsulacją. Każda kolejna warstwa przeprowadza podobny proces.

MODEL ODNIESIENIA TCP/IP

Chociaż model odniesienia OSI jest uniwersalny, historycznym, technicznym i otwartym standardem internetowym jest protokół TCP/IP. Model odniesienia TCP/IP, jak również rodzina protokołów TCP/IP, umożliwiają komunikację między dowolnymi dwoma komputerami znajdującymi się w dowolnym miejscu świata.

Model TCP/IP opracowany został przez Ministerstwo Obrony Stanów Zjednoczonych. Jest to standard, na którym „wyrósł” Internet. Model ten składa się z czterech warstw: warstwy aplikacji, warstwy transportu, warstwy internetowej i warstwy dostępu.

Warstwa aplikacji – projektanci TCP/IP wiedzieli, że protokoły wyższego poziomu powinny obejmować funkcje warstwy sesji i prezentacji. Utworzyli warstwę aplikacji, która obsługuje protokoły wyższego poziomu, aspekty reprezentacji, kodowanie oraz kontrolę dialogu. TCP/IP łączy wszystkie aspekty związane z aplikacją w jednej warstwie oraz zapewnia, że dane te są prawidłowo pakowane dla następnej warstwy. Warstwa ta nazywana jest także warstwą przetwarzania.

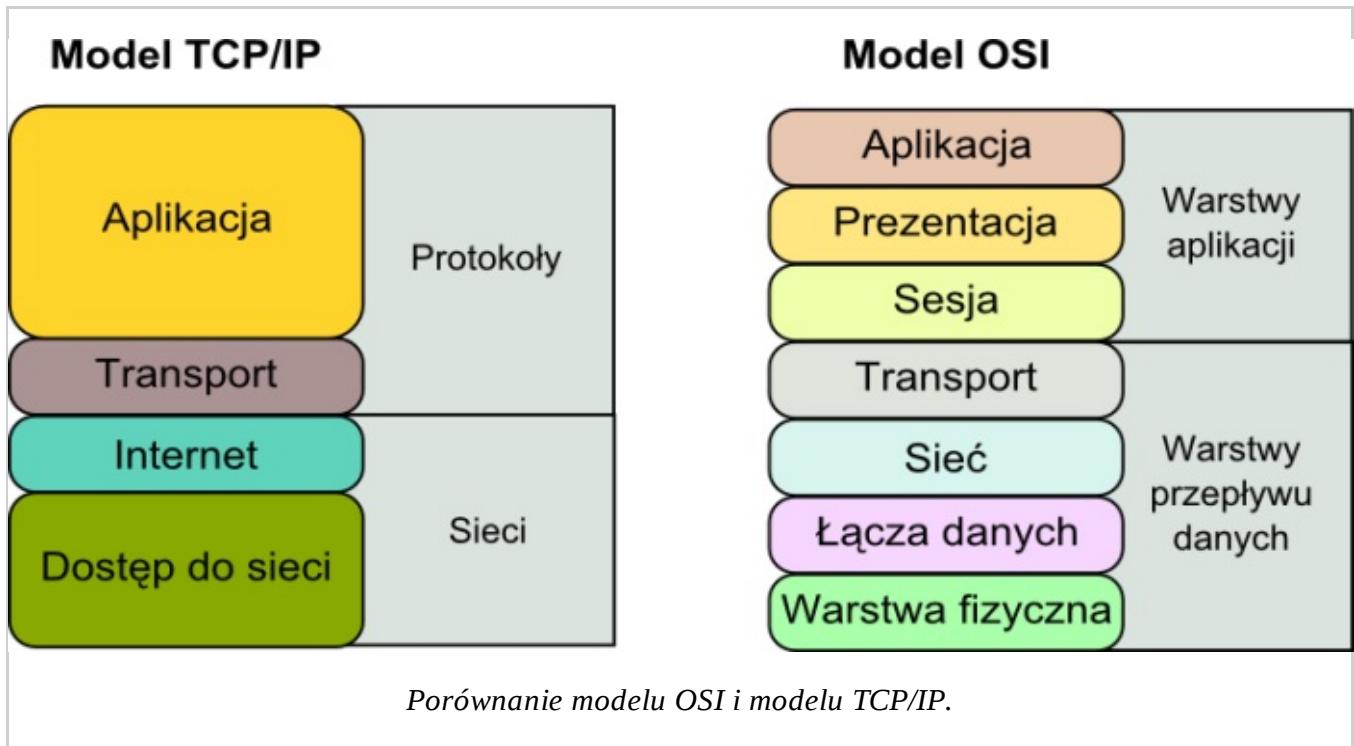
Warstwa transportu – zazwyczaj zajmuje się aspektami związanymi z niezawodnością, kontrolą przepływu i retransmisją. Protokół TCP dostarcza wspaniałe, elastyczne metody tworzenia niezawodnej komunikacji sieciowej cechującej się dobrym przepływem informacji. TCP jest połączeniowy. Obsługuje dialog między źródłem a miejscem przeznaczenia, pakując jednocześnie informacje warstwy aplikacji w jednostki nazywane segmentami. To, że jest protokołem połączeniowym, oznacza że między komunikującymi się komputerami istnieje fizyczny obwód. Oznacza to, że segmenty warstwy 4 muszą wędrować w tą i z powrotem między dwoma hostami, aby zachować logiczne połączenie przed wysłaniem danych. Warstwa ta nazywana jest czasami warstwą host-do-hosta.

Warstwa internetowa – jej zadaniem jest wysyłanie pakietów źródłowych z dowolnej sieci i dostarczanie ich do miejsca przeznaczenia, niezależnie od ścieżek i sieci napotkanych po drodze. Protokołem zarządzającym tą warstwą jest protokół IP. Wyznaczenie najlepszej ścieżki i komutacja pakietów następuje w tej warstwie. Można to porównać do systemu pocztowego. Po wysłaniu listu nie wiemy w jaki sposób dotrze do celu (istnieje wiele możliwych tras), ale zależy nam na tym, aby dotarł.

Warstwa dostępu do sieci – nazwanej warstwy ma szerokie znaczenie i może być myląca. Nazywana jest także warstwą host-do-sieci. Czasami przedstawiana jest jako dwie warstwy, tak jak w modelu OSI. Warstwa dostępu do sieci zajmuje się aspektami wymaganymi przez pakiet protokołu IP do przejścia fizycznym łączem z jednego urządzenia do drugiego, bezpośrednio połączonego. Obejmuje szczegóły związane z technologiami LAN i WAN, a także wszystkie zadania warstw fizycznej i łącza danych w modelu OSI.

MODEL OSI A MODEL TCP/IP

Porównując modele OSI i TCP/IP można zauważać, że łączą je podobieństwa, lecz także dzielą różnice.



Podobieństwa:

- Oba podzielone są na warstwy;
- Oba mają warstwy aplikacji, choć obejmują one różne usługi;
- Oba mają porównywalne warstwy transportu i sieci;
- Zakładana jest technologia komutacji pakietów (a nie komutacji obwodów);
- Profesjonalisi z dziedziny sieci muszą znać oba modele.

Różnice:

- TCP/IP łączy funkcje warstw prezentacji i sesji w warstwie aplikacji;
- TCP/IP łączy warstwy łączą danych i fizyczną modelu OSI w jednej warstwie;
- TCP/IP wydaje się prostszy, ponieważ ma mniej warstw; to jednak nieporozumienie. Model odniesienia OSI jest mniej skomplikowany; ma więcej warstw, a to pozwala na szybszą współpracę i rozwiązywanie problemów;
- Protokoły TCP/IP to standardy, na których oparty jest Internet, dlatego jest on bardziej wiarygodny. Sieci zazwyczaj nie są budowane w oparciu o protokoły modelu OSI, choć wykorzystuje się go jako przewodnika.

URZĄDZENIA SIECIOWE

Pojęcie "HOST"

Urządzenia użytkownika podłączone w sposób bezpośredni z segmentem sieci często nazywane są hostami. Hosty obejmują komputery (zarówno klientów, jak i serwery), drukarki sieciowe, skanery sieciowe oraz wiele innych urządzeń. Urządzenia hostów mogą istnieć i działać bez podłączenia do sieci, ale ich możliwości są wtedy ograniczone.

Urządzenia hostów nie są częścią żadnej warstwy modelu OSI czy TCP/IP. Mają fizyczne połączenie z medianami sieci za pomocą karty interfejsu sieciowego (lub w skrócie karty sieciowej). Funkcje innych warstw modelu OSI są wykonywane przez oprogramowanie hosta. Oznacza to, że działają na wszystkich siedmiu warstwach modelu OSI.

Karta sieciowa - NIC (ang. Network Interface Card) jest uważana za urządzenie warstwy 2 modelu OSI, ponieważ każda karta sieciowa posiada i operuje na unikalnym kodzie, nazywanym adresem MAC (ang. Media Access Control). Adres ten jest używany do komunikacji w warstwie łączności danych. Tak naprawdę karta sieciowa operuje również częściowo na warstwie 1 modelu OSI, ponieważ przetwarza fizyczne sygnały transmisji danych. W zależności od wykorzystanego medium, karta sieciowa będzie miała inne wejścia i wyjścia. Karta do komunikacji radiowej będzie miała dodatkową antenę, a karta do transmisji bezprzewodowej nie będzie miała żadnych wejść czy wyjść. Istnieją karty posiadające wejścia dla 2 typów kabli – skrętki i kabla koncentrycznego, takie karty oznaczane są symbolem *combo*.



Karta sieciowa typu *combo*, przeznaczona do podłączenia do kabla koncentrycznego lub „skrętki”.

Jeżeli chodzi o budowę karty sieciowej, to standardowe karty sieciowe są układami drukowanymi na płytach krzemowych, które umieszcza się w gniazdach rozszerzeń na płycie głównej komputera (np. PCI). W komputerach przenośnych zazwyczaj stosuje się dużo mniejsze karty w standardzie PCMCIA

TYPOWE URZĄDZENIA SIECIOWE

Oprócz hostów i mediów sieciowych, typowa sieć składa się jeszcze z urządzeń sieciowych łączących pozostałe elementy ze sobą.

WZMACNIAK (ANG. REAPETER)

Istnieje wiele typów mediów fizycznych, wadą każdego z nich jest ograniczona maksymalna dopuszczalna długość jednego nieprzerwanego odcina kabla. Wzmacniaki stosuje się w celu rozszerzenia sieci poza promień maksymalnej długości wykorzystywanego medium.

Tak jak media sieciowe, wzmacniaki to urządzenia sieciowe, które istnieją w warstwie 1, czyli warstwie fizycznej modelu odniesienia OSI. Zadaniem wzmacniaka jest zregenerowanie i przedłużenie sygnałów sieciowych na poziomie bitu, co pozwala na ich wędrówkę na dłuższych dystansach. Wzmacniak nie potrzebuje żadnych informacji pochodzących z wyższych warstw modelu OSI.

KONCENTRATOR (ANG. HUB)

Mówiąc ogólnie, termin „koncentrator” jest używany w miejsce wzmacniaka, jeśli odwołujemy się do urządzenia służącego jako centrum sieci. Chociaż koncentrator działa w fizycznej topologii gwiazdy, tworzy takie samo środowisko jak magistrala. Dzieje się tak dlatego, że wszystkie porty koncentratora połączone są za pomocą tych samych obwodów fizycznych. W chwili, gdy jedno urządzenie nadaje, wszystkie inne urządzenia to „słyszą”, z próba nadawania przez inne urządzenia powoduje kolizję.

Zadaniem koncentratora jest regeneracja i przedłużanie działania sygnałów sieciowych. Zadanie to wykonywane jest na poziomie bitu dla dużej ilości hostów. Zjawisko to zwane jest koncentracją. Oto najważniejsze właściwości koncentratorów:

- Regenerują i naprawiają sygnały;
- Propagują sygnały w sieci;
- Nie mogą filtrować ruchu sieciowego (wszystko co przyjdzie na jeden z portów jest rozsypane na wszystkie pozostałe);
- Nie mogą określać najlepszych ścieżek;
- Są używane jako sieciowe punkty koncentracji.



Przykład małego (5-portowego) huba do kabli typu skrętka.

Posiadając koncentrator można połączyć w sieć kilka hostów. Jest to najprostszy z możliwych schematów sieci. Sieć nie jest bardzo wydajna (z reguły koncentratory mają przepustowość rzędu 10 Mb/s) ale można ją zbudować bardzo niskim kosztem. Z uwagi na obniżenie kosztów produkcji sprzętu elektronicznego w ostatnich latach, rzadko stosuje się już tego typu urządzenia na rzecz bardziej wyrafinowanych rozwiązań, których koszty też nie są zbyt wysokie.

MOST (ANG. BRIGDE)

Mosty to urządzenia warstwy 2 modelu OSI. Ich zadaniem jest utworzenie dwóch lub więcej segmentów sieci LAN, z których każdy jest odrębną domeną kolizji. Oznacza to, że dzięki mostom można lepiej wykorzystać dostępne pasmo przepustowości urządzeń sieciowych. Zadaniem mostu jest filtrowanie ruchu w sieci na poziomie warstwy łączącej danych, zachowanie ruchu lokalnego umożliwiając zarazem łączność z innymi częściami sieci LAN. Mosty można przyrównać do prostych routerów warstwy 2 (o routerach w dalszej części lekcji).

Mosty filtryują ruch sieciowy opierając się tylko na adresach MAC, dlatego mogą bardzo szybko przesyłać ruch reprezentujący dowolny protokół warstwy sieci. Most śledzi, które adresy MAC znajdują się po każdej stronie mostu, zapisując je na swojej liście. Na ich podstawie most podejmuje decyzję o przepuszczeniu lub nie ruchu pomiędzy dwoma urządzeniami.

Mosty zajmują się tylko przekazywaniem ramek w oparciu o docelowe adresy MAC. Oto ważne właściwości mostów:

- Są bardziej „inteligentne” niż koncentratory – oznacza to, że mogą analizować przychodzące ramki i przekazywać (lub odrzucać) je w oparciu o informacje adresowe;
- Gromadzą i przekazują pakiety między dwoma lub większą ilością segmentów sieci LAN;
- Tworzą większą liczbę domen kolizji, umożliwiając więcej niż jednemu urządzeniu sieciowemu

- nadawanie bez powodowania kolizji (w różnych segmentach);
- Przechowują tablice adresów MAC.

Tak jak w przypadku kombinacji wiele połączonych wzmacniaków = koncentrator, połączenie wielu mostów tworzy następne urządzenie – przełącznik.

PRZEŁĄCZNIK (ANG. SWITCH)

Przełącznik to podobnie jak most urządzenie warstwy 2 modelu OSI. Przełącznik właściwie stanowi wieloportowy most. Ze względu na podejmowane przez przełączniki decyzje co do ruchu sieciowego, sprawiają one, że sieci LAN są bardziej efektywne. Dzieje się tak, ponieważ przełącznik dynamicznie zestawia połączenia tylko pomiędzy nadawcą i odbiorcą ramek na podstawie adresów MAC. Powoduje to kilkukrotne zmniejszenie generowanego ruchu w porównaniu do koncentratorów, co daje w rezultacie kilkukrotne zwiększenie przepustowości tego typu urządzeń. Z reguły mamy do czynienia z przełącznikami o przepustowości 100 Mb/s lub 1 Gb/s.

Przełączniki na pierwszy rzut oka wyglądają jak koncentratory. Zarówno koncentratory jak i przełączniki mają wiele portów połaczeniowych, ponieważ do ich funkcji należy koncentracja łączności (umożliwiająca połączenie wielu urządzeń z jednym punktem w sieci). Różnica pomiędzy koncentratorem a przełącznikiem tkwi w tym, co się dzieje wewnątrz urządzenia.

Zadaniem przełącznika jest koncentracja łączności, co powoduje, że transmisja danych jest bardziej efektywna. Przełącznik można traktować jako urządzenie łączące właściwe dla koncentratora połączenie z regulacją ruchu na każdym porcie właściwą dla mostu.

ROUTER

Routery nie mają oficjalnego polskiego odpowiednika. W dosłownym tłumaczeniu powinniśmy je nazywać „trasownikami”, lecz ta nazwa nie przyjęła się w środowisku informatycznym.

Routery są urządzeniami sieciowymi należącymi do 3 warstwy a więc warstwy sieciowej modelu OSI. Działania w warstwie 3 pozwala routerowi podejmować decyzje w oparciu o adresy sieciowe zamiast o adresy MAC warstwy 2. Routery mogą łączyć różne technologie warstwy 2, na przykład Ethernet, Token Ring, FDDI. Jednak ze względu na ich zdolności routingu pakietów w oparciu o informacje warstwy 3, routery stały się podstawą Internetu korzystającego z protokołu IP.

Zadaniem routera jest sprawdzenie przechodzących pakietów (danych warstwy 3), wyznaczenie najlepszej ścieżki w sieci i przesłanie ich do właściwego portu wyjścia. Routery to najważniejsze urządzenia regulujące ruch w dużych sieciach.

Routery różnią się od przełączników na wiele sposobów. Po pierwsze, działanie przełączników zachodzi w warstwie 2, warstwie łączna danych, podczas gdy routing następuje w warstwie sieci modelu odniesienia OSI. Po drugie, przełączniki używają fizycznych adresów MAC do podejmowania decyzji odnośnie przekazywania danych. Routery używają innego schematu adresowania, właściwego dla warstwy 3. Opierają się na adresach warstwy sieci, nazywanych adresami IP lub adresami logicznymi (więcej o adresach IP znajduje się w następnej lekcji podręcznika).

Rozróżnić można routery sprzętowe i programowe. Routery sprzętowe są urządzeniami, takimi jak przełączniki, zawierającymi dużą ilość portów (np. 8, 16, 32). W zależności od ilości portów, przepustowości routera jak i dodatkowych właściwości urządzenia te mają różną wielkość poziom skomplikowania i cenę. Router należy dobrą do stawianych przed nim wymagań sieci. Routerem programowym może być standardowy komputer (PC lub serwer) z zainstalowanymi co najmniej dwoma kartami sieciowymi. Oprogramowanie TCP/IP komputera zajmuje się zadaniami routingu ruchu pomiędzy interfejsami sieciowymi.



Przykład niewielkiego routera.

ACCESS POINT

Są to urządzenia wykorzystywane w łączności bezprzewodowej WiFi (ang. *Wireless Fidelity*). Urządzenia te spełniają kilka podstawowych zadań. Zazwyczaj posiadają wbudowane routery, które pozwalają na routing sieci bezprzewodowej do sieci globalnej, do której podłączone jest urządzenie. Zazwyczaj urządzenia te podłączone są do sieci ogólnej za pomocą łącza typu Ethernet. Bardzo ważną funkcją access pointa oprócz konwersji sygnałów elektrycznych na częstotliwości radiowe i na odwrót jest szyfrowanie danych przekazywanych w postaci radiowej. Jak wiadomo sygnały radiowe są bardzo łatwe do odebrania przez osoby trzecie. Łączność z access pointami zapewniają bezprzewodowe karty sieciowe WiFi.

Istnieją 3 standardy wykorzystania technologii WiFi 802.11: a, b oraz g. Przykładowe zasięgi wraz z szybkością przesyłania danych dostępnych na rynku access pointów nie wyposażonych w dodatkowe anteny w najpopularniejszych standardach rodzinie 802.11 zestawiono w poniższej tabeli:

Standard	Przepustowość	Częstotliwość sygnału	Zasięg
802.11a	54 Mb/s	5 GHz	18 m
802.11b	10 Mb/s	2,4 GHz	45 m
802.11g	54 Mb/s	2,4 GHz	40 m



Wygląd przykładowego access pointa.

SIECI TCP/IP

Powoli zbliżamy się do końca naszego podręcznika. Ta lekcja skupia się na sieciach, a właściwie sieci TCP/IP - czyli podstawie działania dzisiejszego Internetu. Przedstawimy w niej stos protokołów TCP/IP, omówimy adresację IP wykorzystywaną w Internecie. Pokażemy jak tworzy się sieci i podsieci. Następnie przedstawimy protokoły i usługi aplikacji TCP/IP, oraz opiszemy podstawowe narzędzia do diagnostyki sieci.

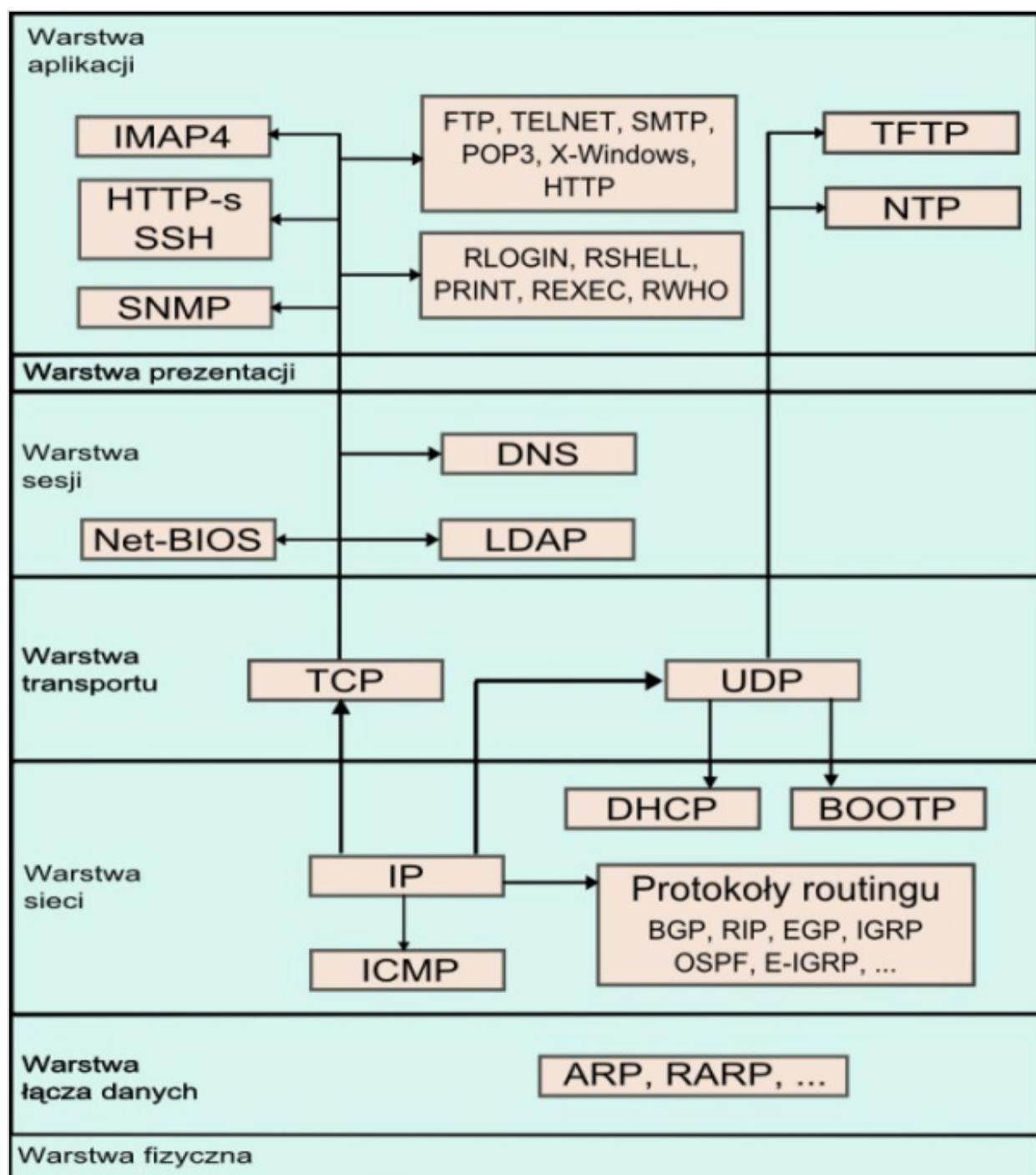
Na koniec zaprezentujemy przykład konfiguracji domowego komputera PC z zainstalowanym systemem Windows 10 do pracy w sieci oraz konfigurację przykładowego routera do zarządzania domową siecią lokalną.

RODZINA PROTOKÓŁÓW TCP/IP

Rodzina protokołów TCP/IP (ang. *Transfer Control Protocol/Internet Protocol*) nazywana często w skrócie protokołem TCP/IP powstała w wyniku prac badawczych przeprowadzonych przez agencję rządową Stanów Zjednoczonych – DARPA. Początkowo zadaniem protokołów było zapewnienie komunikacji w obrębie agencji DARPA, później TCP/IP został dołączony do wersji Berkeley Software Distribution systemu UNIX. Obecnie TCP/IP jest niezaprzecjalnym standardem w łączności sieciowej i służy jako protokół transportowy dla Internetu, dzięki czemu miliony komputerów na całym świecie komunikują się ze sobą.

Zadaniem rodziny protokołów TCP/IP jest przesyłanie informacji z jednego urządzenia sieciowego do drugiego. Poprzez takie działanie, protokół ten ściśle odwzorowuje model odniesienia OSI w niższych warstwach i obsługuje wszystkie standardowe protokoły warstwy fizycznej i warstwy danych.

Mapa wybranych, ważniejszych protokołów z rodziny TCP/IP, operujących na warstwach modelu OSI znajduje się na poniższej ilustracji.



WARSTWA ŁĄCZA DANYCH

ARP (ang. *Address Resolution Protocol*) – jest używany do przekształcania lub odwzorowywania adresów IP do adresu używanego w warstwie łączącej danych – adresu MAC urządzenia sieciowego. Umożliwia to komunikację na niższych warstwach, ponieważ sprzęt warstwy łączącej danych nie akceptuje ramek dopuków adresu MAC zawarty w ramce nie będzie pasował do sprzętowego adresu MAC (lub jego rozgłoszeniowego adresu MAC). Jeśli adres ten nie znajduje się w tablicy, ARP wysyła rozgłoszenie odbierane przez każdą stację w sieci.

RARP (ang. *Reverse Address Resolution Protocol*) – jego działanie jest odwrotne do ARP, oznacza to, że zajmuje się tłumaczeniem adresów MAC na odpowiednie adresy IP.

WARSTWA SIECI

IP (ang. *Internet Protocol*) – umożliwia bezpołączeniowy, najskuteczniejszy routing datagramów; nie zajmuje się zawartością datagramów; szuka drogi, którą może dostarczyć datagramy do miejsca ich przeznaczenia. Bezstanowość protokołu IP oznacza między innymi to, że kolejne datagramy transmisji z jednego komputera do drugiego mogą przebyć inną drogę. Datagram IP zawiera nagłówek IP oraz dane i jest otoczony nagłówkiem i stopką warstwy MAC. Adresacja IP jest adresacją wykorzystywaną w Internecie, więcej na ten temat znajduje się w dalszej części rozdziału.

ICMP (ang. *Internet Control Message Protocol*) – zapewnia kontrolę i przesyłanie komunikatów. Wszystkie hosty TCP/IP implementują protokół ICMP. Komunikaty ICMP są przenoszone w datagramach IP i są wykorzystywane do wysyłania wiadomości o błędach i komunikatów kontrolnych. Stanowią podstawowe narzędzie do diagnostyki sieci TCP/IP.

DHCP (ang. *Dynamic Host Configuration Protocol*) - to protokół komunikacyjny umożliwiający komputerom uzyskanie od serwera danych konfiguracyjnych sieci, np. adresu IP hosta, adresu IP bramy sieciowej, adresu serwera DNS, maski sieci. Serwer DHCP zajmuje się zarządzaniem i przydzielaniem wcześniej skonfigurowanej puli (zakresu) adresów IP.

BOOTP (ang. *Bootstrap Protocol*) – wcześniejsza wersja protokołu DHCP.

PROTOKOŁY ROUTINGU

Podobne terminy protokół routingu oraz protokół routowalny są często mylone ze sobą. Przedstawmy zatem różnicę między nimi:

- **protokół routowalny** - to dowolny protokół, który dostarcza wystarczającą ilość informacji w swoim adresie warstwy sieci, aby umożliwić przekazanie pakietu hosta do hosta w oparciu o schemat adresowania. Protokoły routowalne definiują format i przeznaczenie pól w pakiecie. Pakiety są przekazywane między systemami końcowymi. Protokół IP to przykład protokołu routowalnego;
- **protokół routingu** - to protokół, który obsługuje protokół routowalny, dostarczając mechanizmy dzielenia się informacjami routingu. Komunikaty protokołu routingu przepływają między routerami. Protokół routingu pozwala routerom komunikować się z innymi routerami w celu aktualnienia i utrzymywania tablic. Przykładem protokołu routingu jest RIP (ang. *Routing Information Protocol*).

Protokoły routingu określają ścieżki, którymi podążają protokoły routowalne do ich miejsca przeznaczenia. Przykładami protokołów routingu są wspomniany RIP, IGRA (ang. *Interior Gateway Routing Protocol*), E-IGRP (ang. *Enhanced Interior Gateway Routing Protocol*) czy OSPF (ang. *Open Shortest Path First*). Protokoły routingu pozwalają połączonym routerom tworzyć mapę wewnętrznych tras innych routerów w sieci lub Internecie. Dzięki temu możliwy jest routing (wybór najlepszej ścieżki) i komutacja (zestawienie obwodu). Każdy z protokołów routingu ma swoje właściwości i algorytmy doboru najlepszego w danej chwili następnego skoku dla pakietu. Niektóre z protokołów przeznaczone są dla małych lub autonomicznych sieci, niektóre stosuje się na granicy bardzo dużych sieci (np. krajowych). Szczegółowy opis poszczególnych protokołów routingu oraz ich działania wybiega poza zakres tego materiału. Należy jednak zdawać sobie sprawę o istnieniu zestawu protokołów routingu zajmujących się dynamicznym wybieraniem ścieżek dla poruszających się w sieci pakietów.

WARSTWA TRANSPORTU

TCP (ang. *Transfer Control Protocol*) jest połączeniowym, niezawodnym protokołem umożliwiającym kontrolę przepływu danych za pomocą okien przesyłowych. Połączeniowość oznacza utrzymanie kolejności wysyłanych danych po stronie hosta docelowego oraz potwierdzanie otrzymania każdego datagramu. Niezawodność zapewniana jest przez sekwencyjne numerowanie datagramów i potwierdzeń. TCP ponawia transfer wszystkiego, czego odbiór nie został potwierdzony przez system docelowy. TCP tworzy wirtualny obwód między aplikacjami użytkowników końcowych. Zaletą TCP jest to, że gwarantuje dostarczanie segmentów.

UDP (ang. *User Datagram Protocol*) jest bezpołączeniowym, „niepewnym” protokołem odpowiedzialnym za transmisję komunikatów, ale nie dysponującym oprogramowaniem sprawdzającym dostarczanie segmentów. Zaletą UDP jest szybkość. Ponieważ UDP nie oferuje potwierdzania dostarczenia wiadomości, ruch w sieci jest mniejszy, dzięki czemu transfer przebiega szybciej.

POJĘCIE PORTU

Protokoły TCP i UDP oprócz adresów źródłowego i docelowego hosta operują na dodatkowych parametrach połączenia - portach. Numery portów wykorzystywane są do przekazywania informacji wyższym warstwom stosu TCP/IP i służą do śledzenia różnych konwersacji mających miejsce w sieci w tym momencie czasie. Każda usługa wyższej warstwy TCP używa wybranego portu, zwykle nadaje się standardowe numery portów powszechnym usługom i aplikacjom.

Porty oznacza się liczbami całkowitymi od 0 do $2^{16} - 1 = 65535$. Niektóre porty są zarezerwowane zarówno w TCP jak i UDP, chociaż aplikacje, które mogłyby je obsługiwać mogą nie istnieć. Numerom portów przypisano następujące zakresy:

- **0 - 255** - numery przyznane dla aplikacji publicznych;
- **256 - 1023** - numery przypisane firmom sprzedającym oprogramowanie, porty poniżej 1024 nazywane są portami ogólnie znanyymi (ang. *well-known port numbers*);
- **od 1024** - numery nie są kontrolowane.

Każdy pakiet TCP i UDP zawiera numer portu źródłowego i docelowego pakietu, tak samo jak adresy IP. Przyjęto zasadę, że usługi uruchomione na komputerze działają na portach poniżej 1024, natomiast wychodzące połączenia z hostów klienckich mają port źródłowy zawsze powyżej liczby 1024. Pozwala to łatwą filtrację połączeń. Z reguły na komputerze dostęp do portów poniżej 1023 powinien być ścisłe ustalony i strzeżony.

Przykładami szeroko znanych numerów portów są: 21 - FTP, 22 - SSH, 23 - Telnet, 53 - DNS, itd. Więcej informacji o wymienionych usługach znajduje się w dalszej części lekcji.

PROTOKOŁY WYŻSZYCH WARSTW

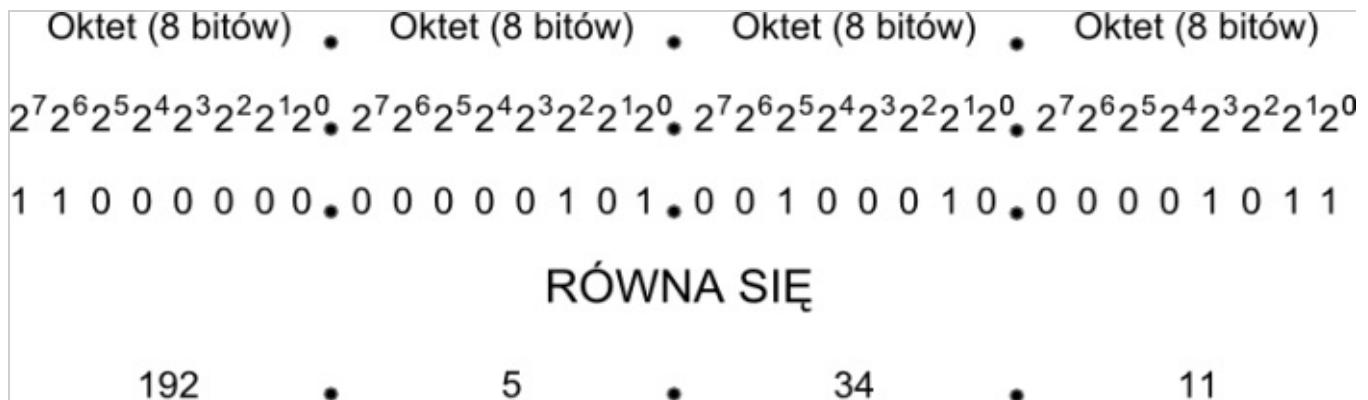
Protokoły wyższych warstw są już wyspecjalizowanymi protokołami służącymi do obsługi konkretnych usług i aplikacji. Przykładowo protokoły POP3, SMTP i IMAP4 służą do przesyłania poczty elektronicznej, protokół DNS zajmuje się tłumaczeniem nazw symbolicznych serwerów na adresy IP i odwrotnie, FTP i TFTP służą do przesyłania plików, SSH i TELNET to protokoły umożliwiające połączenia zdalne do konsoli systemu operacyjnego itd.

O protokołach i usługach wyższych warstw opowiemy w dalszej części lekcji.

ADRESY IP

Protokół IP to najpopularniejsza implementacja hierarchicznego schematu adresowania sieciowego. IP to protokół sieciowy, którego używa Internet.

Adres IP wersji 4 (aktualnie wykorzystywanej) jest liczbą 32 bitową. Adres IP zapisuje się zazwyczaj w postaci czterech oktetów reprezentowanych za pomocą liczb dziesiętnych. Można zatem powiedzieć, że adres IP składa się z 4 bajtów. Zasadę zapisu ilustruje poniższy rysunek.



Zasada przedstawiania adresu IP.

Minimalna wartość każdego z oktetów to 0 (binarnie 00000000), maksymalna 255 (binarnie 11111111). IP składa się z numeru sieci oraz numeru hosta w sieci. Numer sieci identyfikuje sieć, z którą połączone jest urządzenie o podanym IP. Część hosta w numerze IP identyfikuje konkretne urządzenie w sieci. Ponieważ adres IP składa się z 4 oktetów, jeden, dwa lub trzy z nich mogą być użyte do identyfikacji sieci. Podobnie, trzy, dwa lub jeden oktet mogą określać identyfikator hosta.

KLASY ADRESÓW IP

Numer IP jest samo opisujący się, oznacza to, że w nim zawarta jest informacja o klasie adresów, do której należy. Zarządzaniem adresami IP zajmuje się organizacja ARIN. Zdefiniowanych jest 5 klas adresów IP.

	Oktet 1	Oktet 2	Oktet 3	Oktet 4
Klasa A	N	H	H	H
Klasa B	N	N	H	H
Klasa C	N	N	N	H

Klasa D: rozgłoszanie do grup (multicasting)

Klasa E: do celów badawczych

N = Numer sieci przypisany przez ARIN lub dostawcę usług

H = Numer hosta przypisany przez administratora sieci

Podział adresacji IP na klasy.

KLASA A

Adres klasy A został zaprojektowany tak, aby obsługiwać bardzo duże sieci. Ponieważ potrzeba tworzenia sieci o takich rozmiarach została uznana za minimalną, architektura ta została skonstruowana w taki sposób, że maksymalizowała możliwą liczbę adresów hostów, jednocześnie poważnie ograniczając możliwe do zdefiniowania sieci klasy A. Zapisany w formacie binarnym, pierwszy z lewej bit adresu IP klasy A to zawsze 0. Oznacza to, że do dyspozycji mamy wartości od 00000000 (0 dziesiętnie) do 01111111 (127 dziesiętnie) dla pierwszego oktetu a więc tylko 128 możliwych sieci klasy A. Przykładowym adresem klasy A jest 124.14.93.103. Pierwszy oktet – wartość 124, identyfikuje numer sieci przypisany przez ARIN. Administratorzy sieci przypisują pozostałe 24 bity. Łatwą metodą sprawdzenia, czy urządzenie jest częścią klasy A jest ocena pierwszego oktetu jego adresu IP, który jest zawsze wartością od 1 do 126 (oktet 127 także zaczyna się od bitu 0 ale został zarezerwowany do specjalnych celów – o tym w kolejnej części lekcji).

Wszystkie adresy IP klasy A używają tylko pierwszych 8 bitów do identyfikowania części sieci w adresie. Pozostałe trzy oktety (24 bity) mogą być wykorzystane dla części hosta w adresie. Każda sieć wykorzystująca adres klasy A może przypisać 2 do potęgi 24 minus 2 a więc 16 777 214 możliwych adresów IP urządzeniom połączonym z siecią.

W rzeczywistości nigdy nie tworzy się tak wielkich sieci nie podzielonych na podsieci (informacje na temat podsieci znajdują się w dalszej części lekcji). Ruch rozgłoszeniowy chociażby zapytań protokołu ARP przy tak wielkiej ilości hostów całkowicie uniemożliwiłby wszelką komunikację. Sieci klasy A dzieli się logicznie na podsieci według potrzeb i możliwości.

KLASA B

Adresy klasy B zostały zaprojektowane tak, by obsługiwać potrzeby średnich i dużych sieci. Pierwsze 2 oktety adresu klasy B to zawsze 10 (jeden i zero). Przykładem adresu klasy B może być 152.234.65.33. Pierwsze dwa oktety identyfikują numer sieci przypisany przez ARIN. Administratorzy sieci przypisują pozostały 16 bitów. Łatwa metodą rozpoznania, czy urządzenie jest częścią sieci klasy B jest ocena pierwszego oktetu jego adresu. Adresy IP sieci klasy B zawsze mają wartości z zakresu 128.0.0.0 do 191.255.0.0.

Wszystkie adresy IP klasy B wykorzystują pierwszych 16 bitów do identyfikacji części sieci w adresie. Dwa pozostałe oktety adresu IP mogą być użyte dla części hosta. Każda sieć wykorzystująca IP klasy B może przypisać do 2 do potęgi 16 minus 2 a więc 65 534 możliwych adresów IP urządzeniom połączonym z siecią. Tyle samo bitów zarezerwowanych jest na adresację sieci a więc istnieje 65 534 różnych sieci klasy B.

KLASA C

Przestrzeń adresowa klasy C jest zdecydowanie najpowszechniej używaną wśród wszystkich klas adresów IP w wersji 4. Jej zadaniem jest obsługa małych sieci. Pierwsze 3 bity adresu klasy C to zawsze 110 (jeden, jeden i zero). Przykładem adresu IP klasy C może być 203.12.34.221. Pierwsze trzy oktety identyfikują numer sieci przypisany przez ARIN. Administratorzy sieci przypisują pozostałe 8 bitów. Łatwą metodą rozpoznania, czy urządzenie jest częścią sieci klasy C jest oceną pierwszego oktetu jego adresu IP. Adresy sieci klasy C mają zawsze wartości z zakresu 192.0.0.0 do 223.255.255.0.

Wszystkie adresy IP klasy C używają pierwszych 24 bitów do identyfikacji części sieci w adresie. Tylko ostatni oktet adresu IP klasy C może być użyty dla części hosta w adresie. Każda sieć wykorzystująca adres IP klasy C może przypisać 2 do potęgi 8 minus 2 a więc 254 możliwe adresy IP urządzeniom połączonym z siecią. Ponieważ na numerację sieci zarezerwowane są 24 bity, możliwych sieci klasy C jest 16 777 214.

KLASA D

Jest to specjalna klasa adresów wykorzystywana do przesyłania pakietów rozgłoszeniowych do grup (multicasting), czyli rozsyłaniu pakietu nie do pojedynczego hosta ani do wszystkich hostów w sieci czy podsieci a do pewnej predefiniowanej grupy hostów. Adresy klasy D mają zawsze wartości z zakresu 224.0.0.0 do 239.255.255.254. Pierwsze cztery bity adresu klasy D to 1110, pozostałych 20 bitów reprezentuje adres multicast.

KLASA E

Jest to klasa rezerwowa, przeznaczona do testów. Żaden adres z klasy E nie został dopuszczony do ogólnej sieci Internet. Adresy klasy E mają wartości z zakresu 240.0.0.0 do 255.255.255.254. Pierwsze cztery bity adresu klasy E to 1111.

	1	7	24
Klasa A	0	Sieć	Host
	1 1	14	16
Klasa B	1 0	Sieć	Host
	1 1 1	21	8
Klasa C	1 1 0	Sieć	Host
	1 1 1 1	28	
Klasa D	1 1 1 0	Adres multicast	
	1 1 1 1	28	
Klasa E	1 1 1 1	Rezerwa	

Wzory klas adresacji IP.

ADRESY SPECJALNEGO PRZEZNACZENIA

Oprócz ogólnego schematu adresacji IP oraz podziału na klasy adresów zdefiniowane adresy i grupy adresów specjalnego przeznaczenia. Są to: adres pętli zwrotnej (ang. *loopback*), adresy rozgłoszeniowe (ang. *broadcast*) oraz pula adresów prywatnych.

ADRES PĘTLI ZWROTNEJ

Ostatnia sieć klasy A – 127.0.0.0 została zarezerwowana do celów diagnostyki stosu TCP/IP systemu operacyjnego. Ta zawierająca ponad 16 milionów możliwych adresów hostów sieć została zarezerwowana dla jednego adresu: 127.0.0.1. Adres ten używany jest jako pętla zwrotna (*loopback*). Dane wysłane pod ten adres wędrują zwrotną pętlą do nadawcy z pominięciem sieci, a nawet nie przechodząc przez kartę sieciową. Wysłanie testowego pakietu ping pod adres pętli zwrotnej i odpowiedź z niej informuje nas o prawidłowym działaniu oprogramowania IP w naszym systemie operacyjnym. Wszystkie hosty TCP/IP używają tego adresu, aby odwoływać się do siebie. Z uwagi na wymuszenie wirtualnego połączenia każdego hosta do sieci 127.0.0.0, routing do tej sieci jest problematyczny. Dlatego zdecydowano na rezerwację całej puli adresów tej sieci. Należy zauważyć, że trzy ostatnie oktety adresu, do którego wysyłamy pakiet testujący połączenie może mieć dowolną wartość oprócz 0.0.0 i 255.255.255. Oznacza to, że wysłanie pakietu na adres 127.0.0.1 i 127.2.221.34 będzie miało taki sam rezultat. Odpowiednikiem adresu IP pętli zwrotnej jest nazwa symboliczna lokalnego hosta „localhost”. Nazwa ta tłumaczona jest przez wewnętrzny mechanizm DNS na adres właśnie pętli zwrotnej 127.0.0.1.

ADRES BROADCASTOWY

Oprócz adresu sieci (adres zawierający same zera w części hosta reprezentacji bitowej adresu), istnieje odpowiadający jej adres rozgłoszeniowy (ang. *broadcast*). Obu adresów nigdy nie można przydzielić hostowi, mają one specjalne znaczenie i zadanie w każdej sieci lub podsieci. Adres broadcast dla danej sieci zawiera same jedynki w części przeznaczonej na adresowanie hostów. Adres ten służy do wysyłania pakietów, które przeznaczone są i trafiają do wszystkich hostów z danej sieci. Za pomocą pakietów wysyłanych na adres broadcastowy następuje np. konfiguracja IP hosta za pomocą protokołu DHCP.

Adres sieci klasy B	145	87	0	0
	1 0 0 1 0 0 0 1	1 0 1 0 1 1 1 1	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0
Kolejne adresy hostów	145	87	0	1
	1 0 0 1 0 0 0 1	1 0 1 0 1 1 1 1	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 1
	145	87	0	2
	1 0 0 1 0 0 0 1	1 0 1 0 1 1 1 1	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 1 0
...				
	145	87	255	253
	1 0 0 1 0 0 0 1	1 0 1 0 1 1 1 1	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 0 1
	145	87	255	254
	1 0 0 1 0 0 0 1	1 0 1 0 1 1 1 1	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1 0
Adres broadcast	145	87	255	255
	1 0 0 1 0 0 0 1	1 0 1 0 1 1 1 1	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1 1

Schemat wykorzystania adresów IP w przykładowej sieci klasy B.

ADRESY PRYWATNE

Istnieje pewna pula adresów nie przypisanych w sieci publicznej. Są to tak zwane adresy prywatne. Adresy te mogą być przydzielane dowolnie przez administratorów sieci lokalnych o ile pakiet IP z takim adresem nie wydostanie się do sieci publicznej. Adresy te stosuje się do wewnętrznej adresacji komputerów w sieci, gdy mamy np. jeden punkt dostępowy do sieci globalnej i posiadamy wykupiony tylko jeden adres IP z puli publicznej. Jak zatem można łączyć się z komputerów znajdujących się wewnętrz sieci z Internetem jeżeli nie wolno nam „wypuścić na świat” pakietu z adresem prywatnym? Otóż zajmuje się tym specjalny mechanizm tłumaczenia adresów sieciowych NAT (ang. Network Adress Translation). Mechanizm NAT podmienia wewnętrzny adres IP w nagłówku pakietu wychodzącego z sieci na zewnętrzny (publiczny) adres IP i zapamiętuje tę translację, po czym gdy nadejdzie pakiet z odpowiedzią, z powrotem podmienia adres na ten z sieci prywatnej i przekazuje pakiet do hosta docelowego.

Do adresowania prywatnego dostępne są następujące zakresy adresów IP:

- **10.0.0.0 – 10.255.255.255** - cała sieć 10.0.0.0 klasy A udostępniona jest do wykorzystania do adresacji prywatnej.
- **172.16.0.0 – 172.31.255.255** - 16 sieci klasy B.
- **192.168.0.0 – 192.168.255.255** - 256 sieci klasy C.

Adres IP z podanych zakresów nigdy nie powinien pojawić się w sieci publicznej. Wybór konkretnego zakresu zależy od administratora sieci, ma on tutaj pełną dowolność. Wybór może być podykowany warunkom, jakie musi spełniać sieć – znacząca będzie z pewnością liczba hostów do zaadresowania ale również podział logiczny sieci (np. na budynek, piętra i pokoje – do tego celu najlepiej nadać adresację z pierwszego zakresu – drugi oktet może reprezentować budynek, trzeci piętro a czwarty konkretny komputer).

IP W WERSJI V6

W przypadku wersji 4 protokołu IP, wykorzystywane adresy miały rozmiar 32 bitów. Pozwala to na wykorzystanie 2^{32} a więc około czterech miliardów trzystu milionów unikalnych adresów IP. Odpowiada to 8,42 adresów na kilometr kwadratowy powierzchni Ziemi. W okresie bumu Internetu i nowych technologii szybko okazało się, że liczba adresów IP już niedługo będzie niewystarczająca. Oprócz wprowadzenia oszczędności w przyznawaniu adresów IP z puli publicznej rozpoczęto prace nad kolejną – 6 wersją IP, tzw. IPv6. W wersji 6 protokołu adres IP reprezentowany jest przez liczbę 128-bitową, co daje już 2^{128} a więc około $3,4 \times 10^{38}$ unikalnych kombinacji. Taka ilość adresów w

chwili obecnej wydaje się być nie do wykorzystania. Zapis adresu IP w wersji 6 różni się od zapisu swojego poprzednika. Adres reprezentuje się za pomocą cyfr heksadecymalnych z dwukropkiem co 16 bitów a więc 4 znaki, np.: b091:2120:a300:0000:0201:021f:e234:2001 Dozwolone jest opuszczenie wiodących zer, co w granicznym przypadku bloku składającego się samych zer spowoduje pominięcie całego bloku (w takim przypadku w adresie znajdują się dwa znaki dwukropka jeden po drugim). Oznacza to, że poniższy zapis jest prawidłowy i równoważny z wcześniejszym:
b091:2120:a300::201:21f:e234:2001

PODSIECI

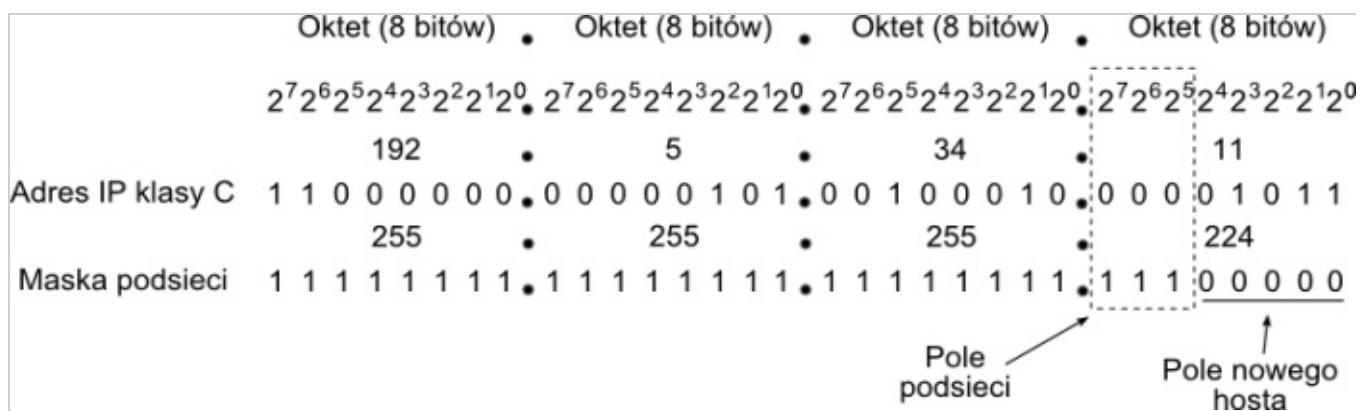
Początkowo dwupoziomowa hierarchia Internetu zakładała, że każdy węzeł będzie należał tylko do jednej sieci. Stąd też, każdy węzeł wymagałby tylko jednego połączenia z Internetem. Założenia te początkowo były bezpieczne. Na przestrzeni czasu przemysł sieciowy jednak dojrzał i rozszerzył się. W roku 1985 nie było już bezpiecznie zakładać, że firma będzie miała tylko jedną sieć, ani że jedno połączenie z Internetem spełni jej potrzeby. Zaistniała potrzeba wprowadzenia mechanizmów różnicujących wiele logicznych sieci, które powstawały jako podgrupy drugiej warstwy Internetu.

Administratorzy sieci czasem muszą dzielić sieci, zwłaszcza te duże, na mniejsze. Sieci powstałe w wyniku tego podziału noszą nazwę podsieci i zapewniają elastyczność w adresowaniu. Koncepcja dzielenia na podsieci bazuje na potrzebie trzeciego poziomu w hierarchii adresowej Internetu.

W środowiskach wielosieciowych, każda podsieć jest połączona z Internetem za pośrednictwem wspólnego punktu, ogólnie rzecz biorąc routera. Szczegóły dotyczące wewnętrznego środowiska sieciowego nie są ważne dla Internetu. Stanowią prywatną sieć, która może dostarczać własne datagramy. Dlatego Internet koncentruje się tylko na tym, jak dotrzeć do sieciowej bramy routera połączonego z Internetem. Wewnątrz prywatnej sieci, część hosta w adresie IP może być podzielona w celu utworzenia podsieci. Główną przyczyną użycia podsieci jest redukcja rozmiaru domeny rozgłoszeń. Rozgłoszenia są wysyłane do sieci lub podsieci, co w przypadku tych drugich może znacznie obniżyć ich ilość, a więc wykorzystywane przez nie pasmo łącza.

ADRESY PODSIECI

Adresy wykorzystywane w podsieciach składają się z części sieci klas A, B i C, pola podsieci oraz pola hosta. Decyzję, jak podzielić sieć na podsieci podejmuje administrator, ma on dosyć sporą elastyczność adresowania. Aby utworzyć adres podsieci, administrator „pożyczca” początkowe bity z części hosta adresu i przypisuje je jako pole podsieci.



Zasada „pożyczania” bitów z części hosta adresu.

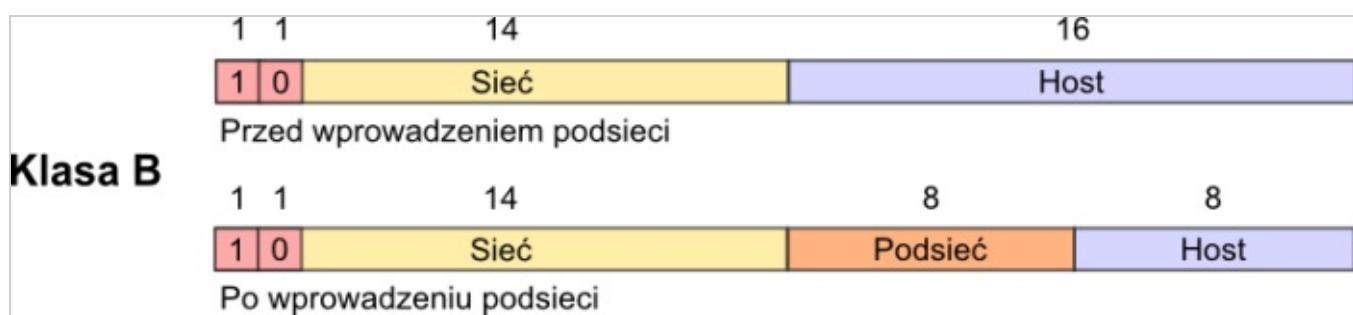
MASKA PODSIECI

Maska podsieci (formalny termin – rozszerzony prefiks sieci) jest związana z danym adresem IP i określa, która część adresu jest polem sieci, a która polem hosta. Maska podsieci ma długość 32 bitów i składa się z 4 oktetów, tak samo jak adres IP. Maska podsieci przedstawiona w postaci binarnej zawiera jedynki w części sieci i podsieci adresu oraz zera w części hosta. Maska podsieci zawsze zawiera nieprzerwany ciąg jedynek z lewej strony oraz nieprzerwany ciąg zer z prawej strony. Zmniejsza to

ilość kombinacji i możliwych wartości każdego oktetu. Standardowe maski sieci klas A, B i C zawierają jedynki w części sieci adresu a więc są to 255.0.0.0, 255.255.0.0 i 255.255.255.0. Maski podsieci dla tych klas zawsze będą zawierały wartości na kolejnych, standardowo zerowych oktetach. Notację zapisu maski często zastępuje się jej uproszczoną wersją, która określa sumę jedynek występujących w masce zapisanej w postaci binarnej. Maskę dla klasy B 255.255.0.0, w postaci binarnej 11111111.11111111.00000000.00000000, można zapisać za pomocą liczby /16. Całkowity zapis przykładowego adresu IP wraz z jego maską może przedstawiać się zatem następująco: 167.34.124.32/16. Analogicznie oznacza się maski dla klas A i C - /8 i /24, jak również dla dowolnej maski podsieci.

TWORZENIE PODSIECI

Aby utworzyć podsieci, trzeba rozszerzyć część routingu w adresie. Internet „widzi” sieć jako całość, identyfikowaną adresem klasy A, B lub C, który jest zdefiniowany przez 8, 16 lub 24 bity routingu (numer sieci). Pole podsieci reprezentuje dodatkowe bity routingu, dzięki którym routery mogą rozpoznać różne lokalizacje, czyli podsieci, w obrębie jednej sieci.



Przykładowy podział adresu IP przy tworzeniu sieci z wykorzystaniem 8 bitów dla klasy B.

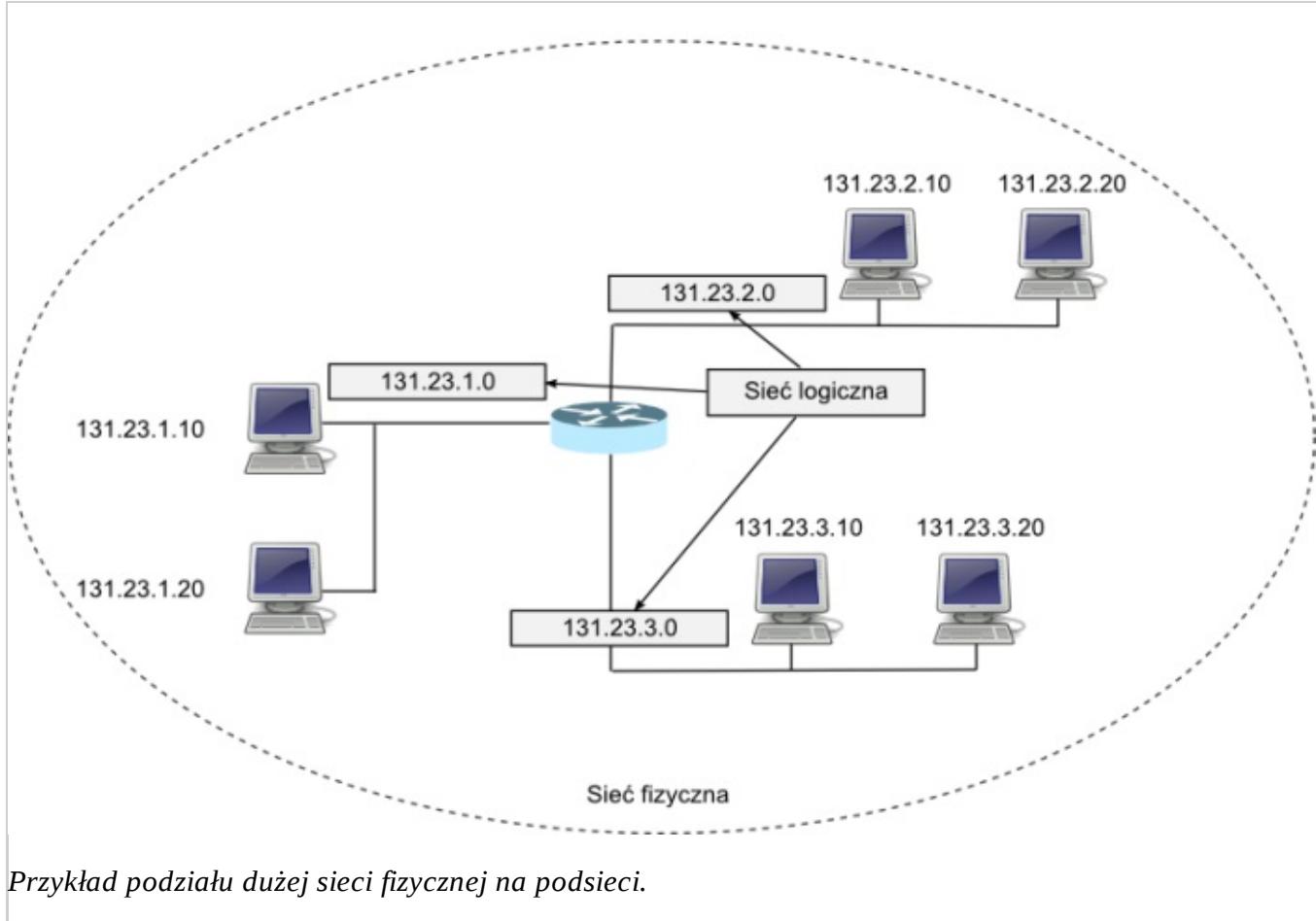
Liczba podsieci i możliwa dla podsieci ilość adresów do wykorzystania dla hostów zależy od ilości bitów zapożyczanych z części hosta. Dla poszczególnych klas adresów IP mamy dostępną różną liczbę możliwych do pożyczania bitów. Minimalna ilość bitów do wykorzystania w podsieci to 1, maksymalna to liczba bitów hosta w danej klasie adresu – 2. Przeznaczenie wszystkich bitów hosta oprócz jednego na utworzenie podsieci nie ma sensu, ponieważ pozostawałoby tylko 2 możliwe adresy w takiej podsieci. Pierwszy adres zawsze reprezentuje numer sieci a ostatni adres rozgłoszeniowy dla danej sieci, nie pozostało by zatem żadnego adresu do przydzielenia hostom w takiej sieci.

Klasa adresu	Rozmiar domyślnego pola hosta	Minimalna liczba bitów podsieci	Maksymalna liczba bitów podsieci	Maksymalna liczba podsieci
A	24	1	22	$2^{22} = 4194304$
B	16	1	14	$2^{14} = 16384$
C	8	1	6	$2^6 = 64$

Liczba bitów do wykorzystania przy tworzeniu podsieci.

Za każdym razem, gdy pożyczamy następny bit z części hosta na utworzenie podsieci, liczba podsieci podwaja się. Dla 1 pożyczonego bitu, liczba sieci wynosi 2, dla 2 pożyczonych bitów, mamy 4 możliwe

podsieci, dla 3 bitów podsieci jest już 8 itd. W analogiczny sposób zmniejsza się liczba adresów hostów w każdej z utworzonych podsieci. Przykładowo dla adresu klasy C, standardowo mamy $2^8 - 2 = 256 - 2 = 254$ możliwe adresy hostów. Jeżeli pożyczymy 1 bit dla podsieci mamy $2^{7-2} = 128 - 2 = 126$ dostępnych adresów. Pożyczając 2 bity mamy zaledwie 62 możliwe adresy hostów itd. Tworząc podsieć o maksymalnej liczbie pożyczonych bitów zostają nam tylko 2 adresy przeznaczone dla hostów (pierwszy i ostatni adres zarezerwowane są dla numeru sieci i adresu broadcast).



Przykład podziału dużej sieci fizycznej na podsieci.

Podczas konfiguracji routerów należy połączyć każdy segment sieci z osobnym interfejsem sieciowym (może być logiczny). Każdy z tych segmentów staje się wtedy odrębną podsiecią. Z każdej z tych podsieci należy wybrać adres, który zostanie przypisany interfejsowi routera podłączonemu z daną podsiecią. Z reguły jest to pierwszy adres z puli adresów hostów a więc np. 131.23.1.1 dla pierwszej z podsieci z powyższej ilustracji. Każdy segment sieci musi mieć inny numer sieci/podsieci.

MASKI PODSIECI O ZMIENNEJ DŁUGOŚCI

Chociaż podział na podsieci to wartościowy dodatek do architektury adresowania internetowego, początkowo miał jedno fundamentalne ograniczenie – dla całej sieci można było zastosować tylko jedną maskę podsieci. Po dokonaniu wyboru maski podsieci, która bezpośrednio określa dostępną liczbę adresów hostów przypadających na każdą z nich, nie można było obsługiwać podsieci o różnych rozmiarach. Rozwiązaniem są maski podsieci o zmiennej długości VLSM (ang. Variable-Length Subnet Mask) opracowane w późniejszym czasie. VLSM umożliwia bardziej efektywne wykorzystanie przestrzeni adresowej IP, którą dysponuje administrator sieci. Może on dostosować rozmiar maski do wymagań każdej podsieci.

PODSIECI – JAK TO SIĘ ROBI W PRAKTYCE?

W praktyce to administrator sieci zarządza konfiguracją podsieci. Jeżeli zmuszony jest podzielić sieć na podsieci lub z jakichś przyczyn stwierdzi, że takie rozwiązańe będzie bardziej efektywne, projektuje on podział logiczny. Podział logiczny może być i z zazwyczaj jest związany z fizycznym położeniem poszczególnych segmentów sieci (różne lokalizacje, budynki, piętra, pokoje, itp.). W swoim projekcie administrator powinien uwzględnić wszystkie hosty podłączone do sieci oraz przyporządkować je do poszczególnych podsieci. Konfiguracja podsieci odbywa się na identycznej zasadzie jak pojedynczej sieci. Ustawienia adresów IP, maski podsieci i pozostałych parametrów połączenia sieciowego konfiguruje się ręcznie na każdym z węzłów lub dynamicznie poprzez usługę DHCP. Wpisanie nieprawidłowej maski podsieci lub adresu hosta z innej podsieci w konfiguracji IP uniemożliwi danemu hostowi poprawne łączenie się ze swoją jak i pozostałymi podsieciami.

PROTOKOŁY I USŁUGI STOWARZYSZONE

W czystej definicji protokołu TCP/IP, jego rola kończy się wraz z adresowaniem. Jednakże, ponieważ taka funkcjonalność w dobie internetu jest niewystarczająca, oraz ponieważ pewne standardy już się wykryształowały, często można spotkać włączanie do definicji TCP/IP zestawu Wyspecjalizowanych protokołów wyższych warstw. My nie włączamy ich do standardu, natomiast nazwiemy protokołami stowarzyszonymi, i postaramy się przybliżyć te najważniejsze i najczęściej wykorzystywane w Internecie

DNS

Internet oparty jest na hierarchicznym schemacie adresowania. Umożliwia to routing bazujący na grupach adresów a nie pojedynczych adresach. Dla większości ludzi zapamiętanie 32 bitowej liczby reprezentującej adres IP jest bardzo trudne a na dłuższą metę praktycznie niemożliwe. Zapisanie adresu IP w postaci czterech liczb dziesiętnych oddzielonych kropkami pomaga ale przy konieczności zapamiętania wielu różnych adresów również jest niewystarczające. System nazw domen DNS (ang. Domain Name System) został stworzony w celu wyeliminowania tego problemu. Nazwa domeny to łatwy do zapamiętania ciąg znaków lub liczb, zazwyczaj nazwa lub skrót, który reprezentuje numeryczny adres węzła internetowego (np. www.okno.pw.edu.pl). Domeny mają również schemat hierarchiczny, na pierwszym poziomie są ściśle kontrolowane i nie jest możliwe dodanie dowolnej nazwy. Nazwy pierwszego poziomu odnoszą się do klas zastosowania adresu lub kraju przynależności, np. .com – komercyjne, .org – organizacje społeczne, .edu – edukacyjne, .pl – polskie, .fr – francuskie itd. Dalsze poziomy jak i docelowe adresy domen należące do nich zarządzane są przez specjalne wyznaczone do tego celu organizacje. Aby zarezerwować własną domenę, np. kowalski.com.pl, należy zgłosić się do organizacji zarządzającej węzłem .pl lub .com.pl. Może się zdarzyć, że taka domena została już zarezerwowana przez inną osobę lub instytucję. Rezerwując domenę automatycznie stajemy się właścicielami wszystkich możliwych nazw znajdujących się w hierarchii poniżej. Jeżeli uda nam się zarezerwować domenę kowalski.com.pl to automatycznie posiadamy również adresy jan.kowalski.com.pl czy też anna.kowalski.com.pl, ale nie anna.kowalska.com.pl. Posiadanie samej domeny na niewiele się przyda. Aby była do czegokolwiek przydatna, należy posiadać również adres IP z publicznej puli, do którego chcemy aby nasz nazwa domeny prowadziła. Musimy postarać się też o odpowiedni wpis w specjalnych serwerach DNS, które zajmują się tłumaczeniem nazw domen na powiązane z nimi adresy IP. Usługa taka zazwyczaj udostępniana jest wraz z rezerwacją domeny, możemy jednak postanowić uruchomić własny serwer DNS – szczególnie, jeżeli chcemy mieć szeroko rozwiniętą podhierarchię nazw. Serwery DNS również zorganizowane są w strukturę hierarchiczną, jeżeli odpytywany przez nas serwer DNS nie potrafi przetłumaczyć podanej przez nas nazwy na adres, wysyła zapytanie do serwera znajdującego się wyżej w hierarchii. Proces ten jest powtarzany do momentu, gdy nazwa zostanie przetłumaczona na adres IP lub gdy osiągnięty zostanie serwer najwyższego poziomu. Jeżeli na serwerze najwyższego poziomu nie można odnaleźć nazwy domeny, sytuacja taka traktowana jest jako błąd i do klienta zwracany jest odpowiedni komunikat o błędzie.

LDAP

Jest protokołem umożliwiającym dostęp do usług katalogowych. LDAP (ang. Lightweight Directory Access Protocol) jest oparty na protokole TCP. Usługa katalogowa LDAP umożliwia przechowywanie danych w postaci drzewa a sam protokół dostarcza mechanizmy bardzo szybkich odpowiedzi na żądania zgłasiane przez klienta. Informacje w katalogu są przechowywane w postaci wpisów (Entries). Każdy wpis jest obiektem jednej lub wielu klas. Klassy mogą być dziedziczone. Każda klasa składa się z jednego lub wielu atrybutów, które mogą być opcjonalne lub obowiązkowe. Istnieje wiele podstawowych typów atrybutów. Atrybuty mogą mieć więcej niż jedną wartość. Można tworzyć swoje

klasy i atrybuty. Dostęp do wpisu chroniony jest poprzez listy kontroli dostępu (ACL – Access Control List). Można tworzyć uprawnienia dla kontekstów, wpisów oraz poszczególnych atrybutów. Wpisy mogą być eksportowane / importowane do / z plików tekstowych w specjalnych formacie LDIF (LDAP Data Interchange Format). Format LDIF jest to format wymiany danych protokołu LDAP.

Z uwagi na swoje właściwości katalogi LDAP i usługi związane z protokołem często stosowane są do przechowywania danych oraz uprawnień użytkowników systemów informatycznych. LDAP daje szerokie możliwości utworzenia drzewa odzwierciedlającego np. schemat organizacyjny firmy (hierarchia podległości) oraz zdefiniować prawa dostępu na każdym poziomie drzewa.

NetBIOS

Jest protokołem opartym na TCP działającym w warstwie sesji modelu OSI. NetBIOS (ang. *Network Basic Input/Output System*) zapewnia interfejs dla komunikacji pomiędzy aplikacjami znajdującymi się na różnych komputerach w sieci lokalnej. Protokół NetBEUI wykorzystywany przez systemy rodziny Microsoft Windows jest implementacją protokołu NetBIOS.

SNMP

Producenci sprzętu sieciowego oferują szeroki wachlarz różnego rodzaju narzędzi służących do zarządzania oprogramowaniem sieciowym. Narzędzia te służą do monitorowania węzłów sieci, poziomów ruchu sieciowego, obserwacji wąskich gardeł sieci, śledzenia mierników programowych oraz gromadzenia informacji diagnostycznych.

Protokół SNMP (ang. Simple Network Management Protocol) jest najczęściej używanym i najlepiej znanim narzędziem służącym do zarządzania oprogramowaniem sieciowym. W celu pobrania informacji, protokół SNMP wykorzystuje technikę nazywaną zbiorem MIB (ang. Management Information Base). Oznacza to, że przechodzi przez kolejne urządzenia sieciowe, gromadząc informacje na temat ich stanu a następnie kopiuje te informacje oraz lokalną bazę MIB każdego z tych urządzeń.

Jedną z zalet SNMP jest to, że urządzenia sieciowe nie muszą zgłaszać wystąpienia problemu. Zadanie to przejmuje od nich SNMP. W dużych sieciach, składających się z wielu urządzeń i zasobów sieciowych, technika gromadzenia informacji przez SNMP może jednak okazać się wadą. Przy bardzo dużej ilości węzłów spowoduje znaczny wzrost ruchu w sieci spowalniając transfer innych informacji.

POP₃, SMTP, IMAP₄

POP3 (ang. *Post Office Protocol v. 3*), SMTP (ang. *Simple Mail Transfer Protocol*) oraz IMAP4 (ang. *Internet Message Access Protocol v. 4*) to protokoły służące do obsługi poczty elektronicznej. Protokołem POP3 posługują się aplikacje kliencie poczty elektronicznej (takie jak np. Microsoft Outlook Express czy Pegasus Mail) do ściągania wiadomości e-mail z serwerów pocztowych na lokalny komputer. Protokół SMTP jest wykorzystywany przez serwery pocztowe do przekazywania między sobą wiadomości oraz przez programy pocztowe do wysyłania wiadomości e-mail z komputera klienckiego. Protokół IMAP4 jest następcą protokołu POP3. Ma on dużo większe możliwości, m.in. przeglądania poczty na serwerze bez konieczności pobierania wiadomości na komputer lokalny. Wszystkie protokoły mogą wykorzystywać mechanizmy autoryzacji dostępu (zarówno przy pobieraniu jak i wysyłaniu poczty elektronicznej). Tak samo jak w przypadku pozostałych programów i usług do działania potrzebny jest serwer pocztowy (komputer z zainstalowanym oprogramowaniem serwera pocztowego), na którym użytkownicy otrzymują konta pocztowe oraz aplikacje kliencie służące do wysyłania i odbieranie e-maili. W ostatnich latach bardzo popularne stały się darmowe konta pocztowe

z dostępem przez WWW. Podejście to nie odbiega od wspomnianego schematu klient-serwer. W tym przypadku serwerem jest serwis obsługujący konta pocztowe, który zarazem udostępnia na własnej stronie internetowej interfejs do obsługi konta pocztowego. Tego typu interfejsy niewiele różnią się od podobnych aplikacji lokalnych oprócz tego, że obsługuje się je za pomocą przeglądarki internetowej.

Oprogramowanie serwera mailowego dla systemów rodziny Unix to na przykład: Postfix czy najbardziej popularny Sendmail.

HTTP, HTTP-s

HTTP (ang. *Hyper Text Transfer Protocol*) to protokół związany z siecią WWW (ang. *World Wide Web*), najszybciej rozwijającą się i najczęściej używaną częścią Internetu. Jedną z głównych przyczyn tak niezwykłego rozrostu sieci WWW jest łatwość, z jaką udostępnia się informacje. Przeglądarka sieci WWW to oczywiście aplikacja klient-serwer, co oznacza, że do funkcjonowania wymaga komponentu klienta i serwera. Przeglądarka sieci WWW prezentuje dane w formatach multimedialnych na stronach oferujących tekst, grafiki, dźwięk oraz filmy. Strony sieci WWW są tworzone w języku HTML (ang. *Hyper-Text Markup Language*). HTML dostarcza przeglądarce informacje na temat strony internetowej, dzięki którym można odtworzyć wygląd strony we właściwy sposób. Dzięki hiperłączom (ang. link) nawigacja w sieci WWW jest bardzo prosta. Hiperłącze to obiekt (słowo, fraza lub grafika) na stronie internetowej, które po kliknięciu przenosi użytkownika na nową stronę.

Ponieważ zamieszczana na stronach grafika i multimedia mają formy plików, za pomocą HTTP można również przesyłać pliki. Nie jest to protokół wyspecjalizowany do tego zadania więc ma dwa podstawowe ograniczenia – można ściągać tylko pojedynczy plik oraz nie można wysłać pliku na serwer.

Protokół HTTPS jest szyfrowaną wersją protokołu HTTP. Wszystkie dane przesyłane między serwerem a przeglądarką klienta w sesji HTTPS są szyfrowane.

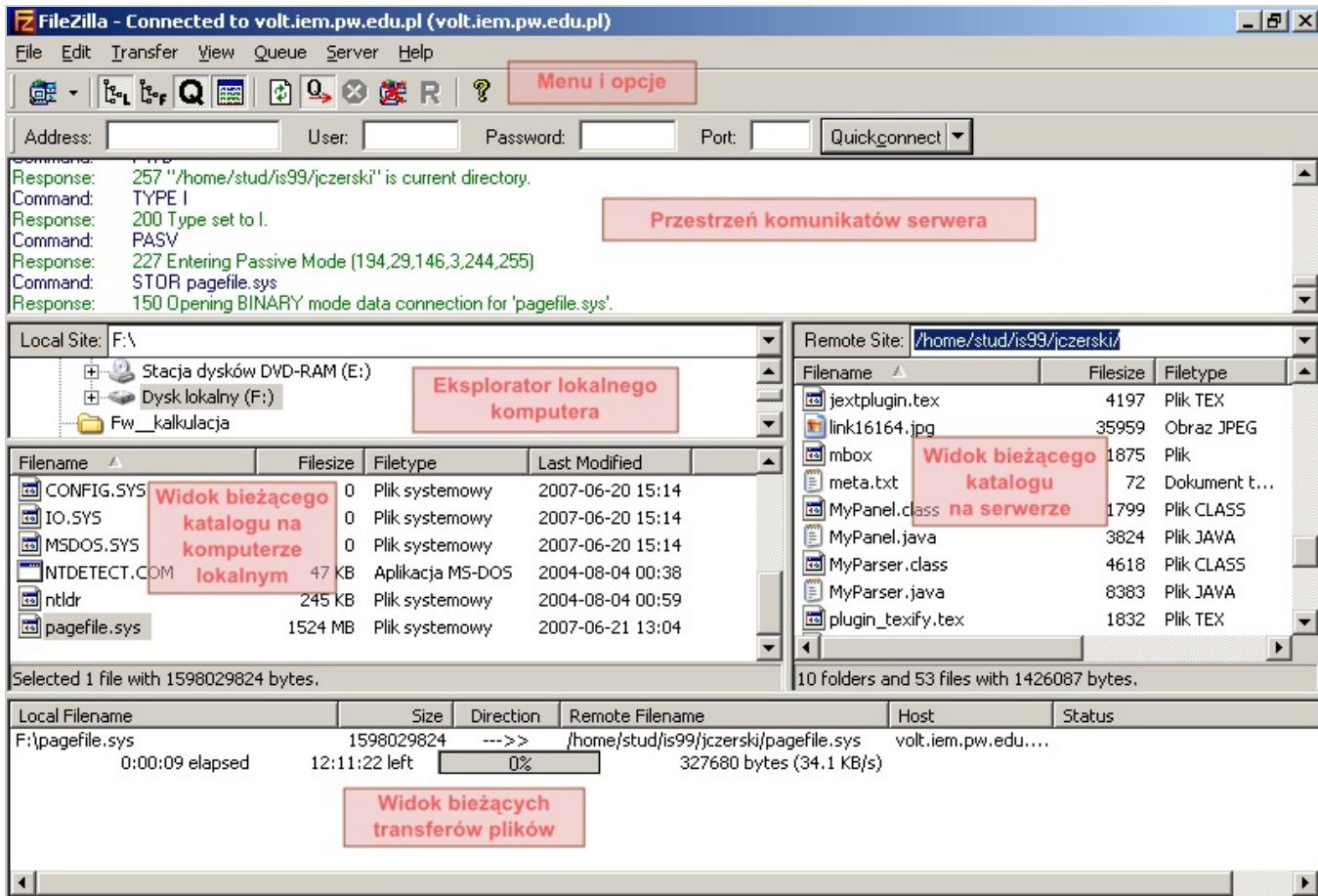
Jeżeli chodzi o oprogramowanie związane z HTTP, to niemal cały rynek serwerów WWW opanował jeden produkt – Serwer Apache. Jest to bardzo popularny, darmowy i szeroko stosowany serwer umożliwiający udostępnianie stron WWW w sieci. Systemy serwerowe Microsoft Windows również udostępniają własne rozwiązanie – IIS (ang. *Internet Information Server*). Oprogramowaniem klienckim dla WWW są przeglądarki internetowe, które w ostatnich kilku latach zaczęły wyrastać jak grzyby po deszczu. Najpopularniejsze to Internet Explorer firmy Microsoft dostarczana wraz z systemem Windows oraz Netscape Navigator. Coraz bardziej doceniane jednak zaczynają być inne przeglądarki, np. Mozilla, Opera, FireFox. Systemy rodziny Unix dostarczają również tekstowe przeglądarki przeznaczone dla terminali tekstowych, są to Lynx oraz Links.

TFTP, FTP

Są to protokoły służące do przesyłania plików przez sieć. TFTP (ang. Trivial File Transfer Protocol) jest bardzo prostym protokołem służącym do przesyłania plików opartym na protokole UDP. Oznacza to, że posiada on wszystkie wady jak i zalety UDP, a więc jest „lekki” ale nie oferuje retransmisji i pewności dostarczenia wszystkich datagramów. Dlatego wykorzystywany jest tylko do przesyłu niewielkich pojedynczych plików. FTP (ang. File Transfer Protocol) to protokół oparty na TCP, a więc posiadający pewność i bezpieczeństwo z nim związane. Protokół ten ma już bardzo duże możliwości, jego główną funkcją jest pobieranie pliku/plików ze zdalnej lokalizacji (serwera) na komputer lokalny oraz wysyłanie pliku/plików na serwer. Do działania FTP potrzebuje oprogramowania na komputerze docelowym – tzw. klienta FTP oraz na serwerze docelowym – tzw. serwer FTP.

Istnieje bardzo dużo programów tego typu, zarówno dla strony klienta jak i serwera, zarówno

darmowych jak i komercyjnych. Systemy operacyjne zawierają przeważnie programy kliencie ftp uruchamiane w trybie tekstowym, jednak ich używanie jest uciążliwe, ponieważ wszystkie polecenia trzeba wpisywać z klawiatury. Ulepszoną wersją programu ftp uruchamianego w linii komend jest wget, ma on nieco szersze możliwości, jednak polecenia nadal wydaje się za pomocą klawiatury. Przykładami pełnowartościowych „okienkowych” klientów FTP są: CuteFTP (komercyjny), FileZilla (darmowy) czy też darmowy SmartFTP. Dodatkowo klienta FTP mają wbudowane inne programy użytkowe, np. TotalCommander czy przeglądarki internetowe. Oprogramowanie serwerowe FTP to na przykład: BulletProof FTP Serwer (komercyjny), G6 FTP Serwer (również komercyjny). Systemy rodziny Unix posiadają zwykle wbudowane oprogramowanie serwera FTP, można również ściągnąć z sieci i zainstalować bardziej wyrafinowane darmowe oprogramowanie tego typu (np. Proftpd).



Ekran przykładowego klienta FTP podczas pracy – FileZilla.

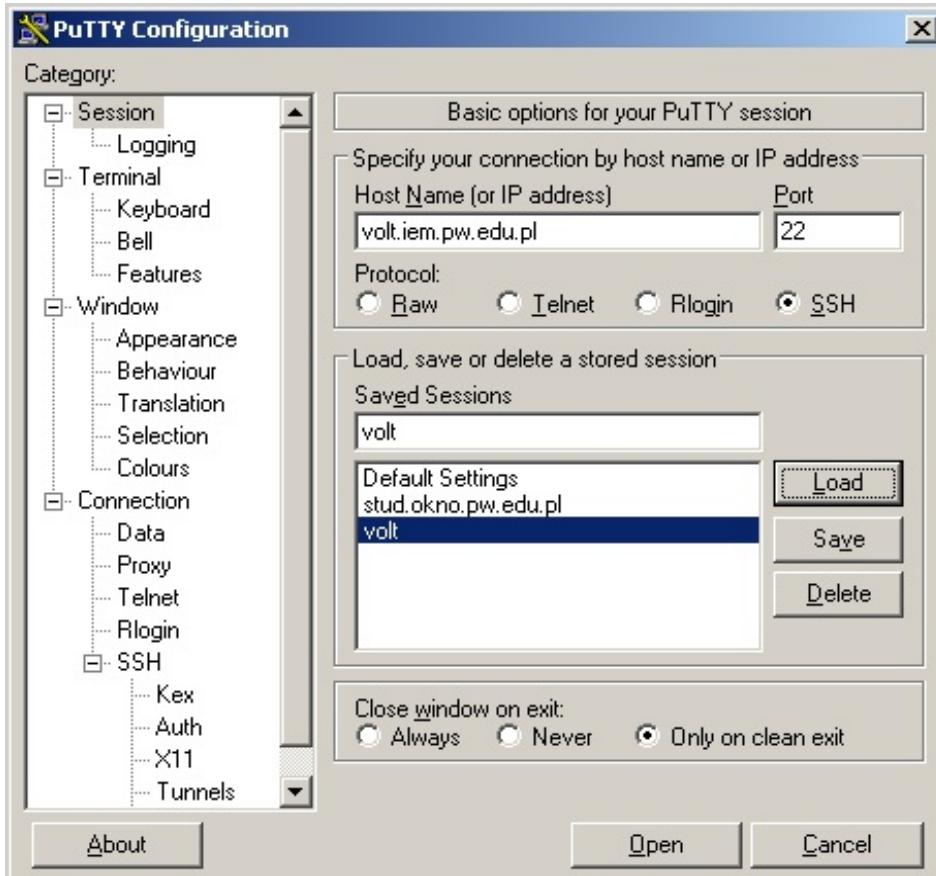
Programu FileZilla na systemie Windows używa się jak każdego programu. Transfer plików i folderów z i do serwera uruchamia się poprzez „przeciąganie” za pomocą myszy odpowiednich obiektów pomiędzy widokami bieżącego katalogu na lokalnym komputerze i bieżącego katalogu na maszynie zdalnej.

TELNET, SSH

Protokoły TELNET (ang. *Terminal Emulation*) i SSH (ang. *Secure Shell*) umożliwiają zdalne połączenie się do terminala innego komputera w taki sposób, jakbyśmy pracowali na komputerze zdalnym. Osoba zalogowana przez terminal telnet lub ssh ma takie same prawa jak osoba siedząca przy komputerze zdalnym. Do pracy, jak wszystkie aplikacje i usługi TCP/IP, potrzebne są serwer i aplikacja kliencka. Oprogramowanie udostępniające terminale telnet i ssh po stronie serwera nazywane są demonami. Są to programy pracujące non-stop w tle systemu operacyjnego, oczekujące nadchodzących z zewnątrz

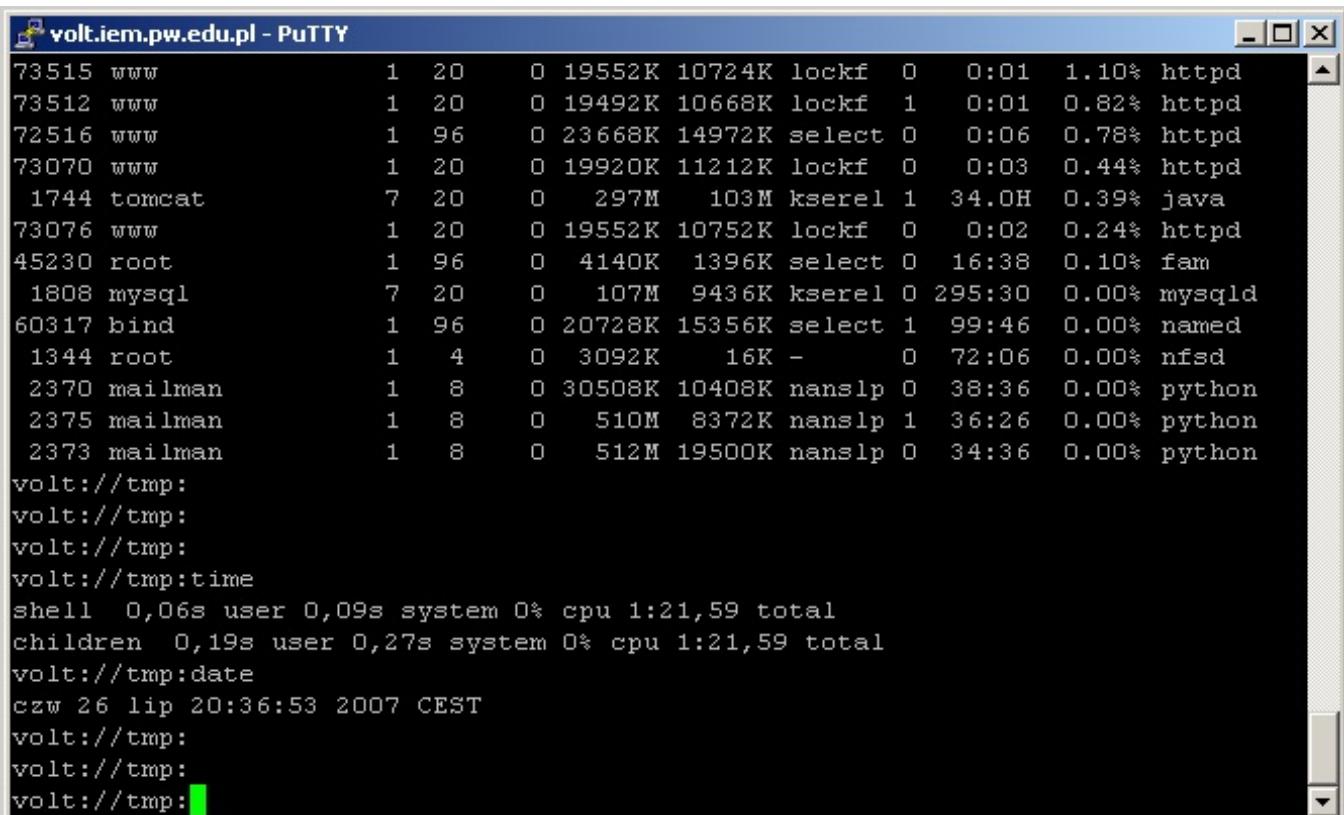
połączeń. Po ustanowieniu połączenia i dokonaniu autoryzacji użytkownika zostaje on dopuszczony do wydawania poleceń systemowi operacyjnemu. Różnica pomiędzy sesją telnet a ssh najprościej mówiąc jest taka, że sesja ssh jest szyfrowana a telnet nie. Po stronie serwera mamy zatem programy nazywane demonami, udostępniające daną usługę terminalową. Programy te są wbudowane w systemy rodziny Unix, nie ma większego sensu starać się uruchomić je na systemach Windows choćby z tego powodu, że w systemie Windows większość poleceń wydaje się za pomocą myszy a nie poleceń tekstowych. Systemy Windows oferują rozwiązań działające na podobnej zasadzie i umożliwiające pracę w środowisku graficznym – zdalny pulpit.

Jeżeli chodzi o oprogramowanie klienckie to program telnet wbudowany jest nawet w systemy Windows. Na rynku istnieje wiele rozwiązań tego typu, zdecydowanie najpopularniejszym jest program PuTTY, który może łączyć się z serwerem za pomocą protokołów RAW, RLOGIN, TELNET oraz SSH. Inne programy tego typu to np. OpenSSH, WinSCP czy SecureCRT.



Ekran powitalny programu PuTTY.

PuTTY umożliwia nam skonfigurowanie nawiązywanej sesji z serwerem. Możemy wybrać odpowiedni protokół, określić adres serwera oraz port docelowy.



```
volt:~ voltmie@volt:~$ top
 73515 www          1  20      0 19552K 10724K lockf  0   0:01  1.10% httpd
 73512 www          1  20      0 19492K 10668K lockf  1   0:01  0.82% httpd
 72516 www          1  96      0 23668K 14972K select  0   0:06  0.78% httpd
 73070 www          1  20      0 19920K 11212K lockf  0   0:03  0.44% httpd
 1744 tomcat       7  20      0  297M   103M kserel 1  34.0H  0.39% java
 73076 www          1  20      0 19552K 10752K lockf  0   0:02  0.24% httpd
 45230 root         1  96      0  4140K  1396K select  0  16:38  0.10% fam
 1808 mysql         7  20      0  107M   9436K kserel 0  295:30  0.00% mysqld
 60317 bind         1  96      0 20728K 15356K select  1  99:46  0.00% named
 1344 root          1   4      0  3092K   16K -     0   72:06  0.00% nfsd
 2370 mailman       1   8      0 30508K 10408K nanslp 0  38:36  0.00% python
 2375 mailman       1   8      0   510M   8372K nanslp 1  36:26  0.00% python
 2373 mailman       1   8      0   512M  19500K nanslp 0  34:36  0.00% python
volt://tmp:
volt://tmp:
volt://tmp:
volt://tmp:time
shell 0,06s user 0,09s system 0% cpu 1:21,59 total
children 0,19s user 0,27s system 0% cpu 1:21,59 total
volt://tmp:date
czw 26 lip 20:36:53 2007 CEST
volt://tmp:
volt://tmp:
volt://tmp:
```

Otwarta sesja SSH z serwerem za pomocą programu PuTTy.

Po utworzeniu połączenia dostajemy się do konsoli systemu operacyjnego posiadając uprawnienia zgodne z naszymi prawami użytkownika.

NTP

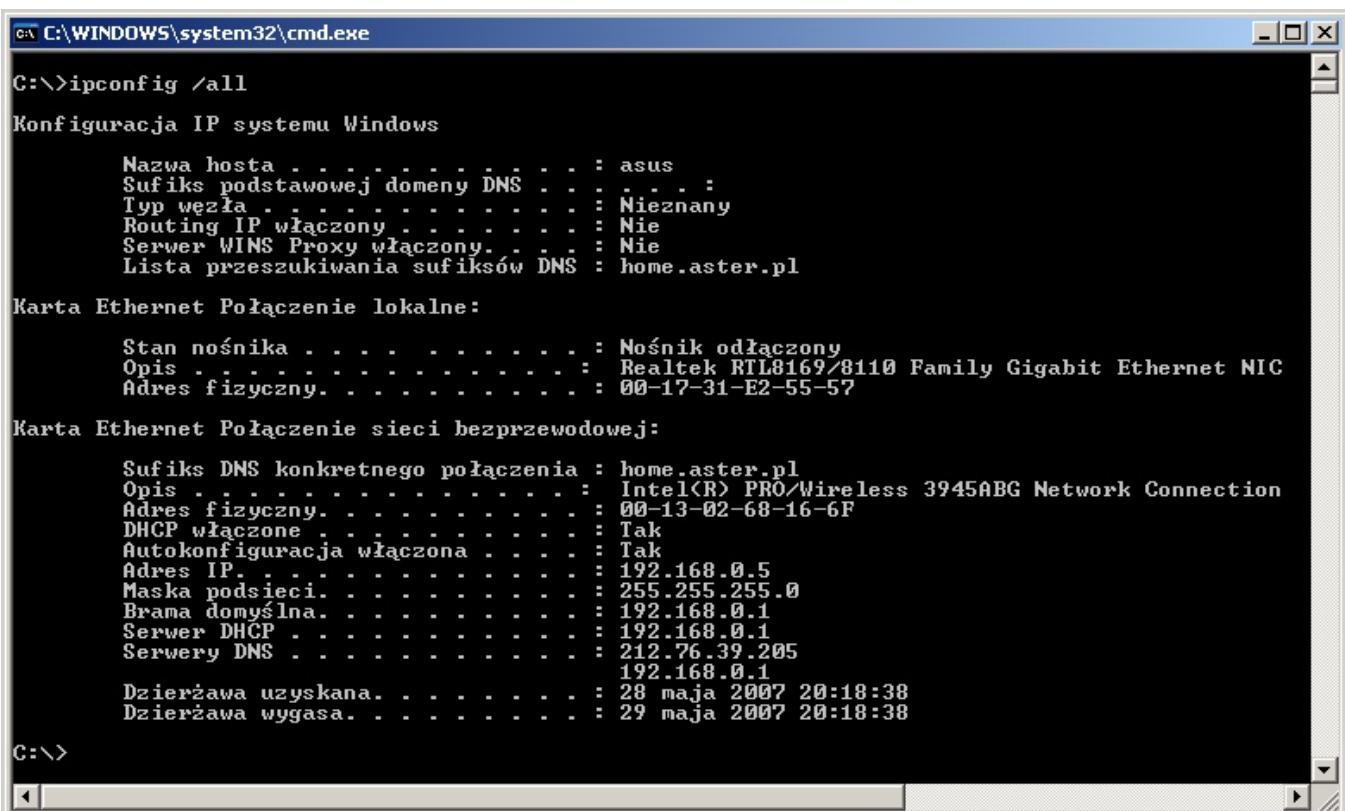
NTP (ang. *Network Time Protocol*) to protokół wykorzystywany do synchronizacji czasu pomiędzy komputerami. Możemy skonfigurować nasz komputer lub serwer aby wykorzystując protokół NTP synchronizował czas systemowy z czasem podawanym przez specjalny serwer czasu (ang. *Time Server NTP*). Synchronizacja czasu jest zagadnieniem niebanalnym w informatyce, i chociaż dla zwykłego użytkownika wydaje się mało istotna, w niektórych przypadkach może mieć kluczowe znaczenie.

DIAGNOSTYKA SIECI TCP/IP

Zarówno systemy rodziny Windows czy Unix oferują podstawowe narzędzia do diagnostyki sieci, śledzenia tras pakietów czy sprawdzania bieżącej konfiguracji sprzętowej.

KONFIGURACJA IP

Do wyświetlania bieżącej konfiguracji TCP/IP systemu służy polecenie ipconfig w nowszych systemach Windows, winipcfg w starszych systemach Windows oraz ifconfig w systemach rodziny Unix. Wykonanie każdego z powyższych poleceń w odpowiednim systemie będzie miało podobny rezultat – wyświetli konfigurację IP wszystkich aktywnych interfejsów sieciowych znajdujących się w komputerze. Wygląd i kolejność poszczególnych informacji może się nieco różnić, informacje powinny być bardzo podobne.



```
C:\>ipconfig /all

Konfiguracja IP systemu Windows

    Nazwa hosta . . . . . : asus
    Sufiks podstawowej domeny DNS . . . . . :
    Typ węzła . . . . . : Nieznany
    Routing IP włączony . . . . . : Nie
    Serwer WINS Proxy włączony . . . . . : Nie
    Lista przeszukiwania sufiksów DNS : home.aster.pl

Karta Ethernet Połaczenie lokalne:

    Stan nośnika . . . . . : Nośnik odłączony
    Opis . . . . . : Realtek RTL8169/8110 Family Gigabit Ethernet NIC
    Adres fizyczny. . . . . : 00-17-31-E2-55-57

Karta Ethernet Połaczenie sieci bezprzewodowej:

    Sufiks DNS konkretnego połączenia : home.aster.pl
    Opis . . . . . : Intel(R) PRO/Wireless 3945ABG Network Connection
    Adres fizyczny. . . . . : 00-13-02-68-16-6F
    DHCP włączone . . . . . : Tak
    Autokonfiguracja włączona . . . . . : Tak
    Adres IP . . . . . : 192.168.0.5
    Maska podsieci. . . . . : 255.255.255.0
    Brama domyślna. . . . . : 192.168.0.1
    Serwer DHCP . . . . . : 192.168.0.1
    Serwery DNS . . . . . : 212.76.39.205
                                192.168.0.1
    Dzierżawa uzyskana. . . . . : 28 maja 2007 20:18:38
    Dzierżawa wygasła. . . . . : 29 maja 2007 20:18:38

C:\>
```

Wyświetlenie przykładowej konfiguracji IP w systemie Windows.

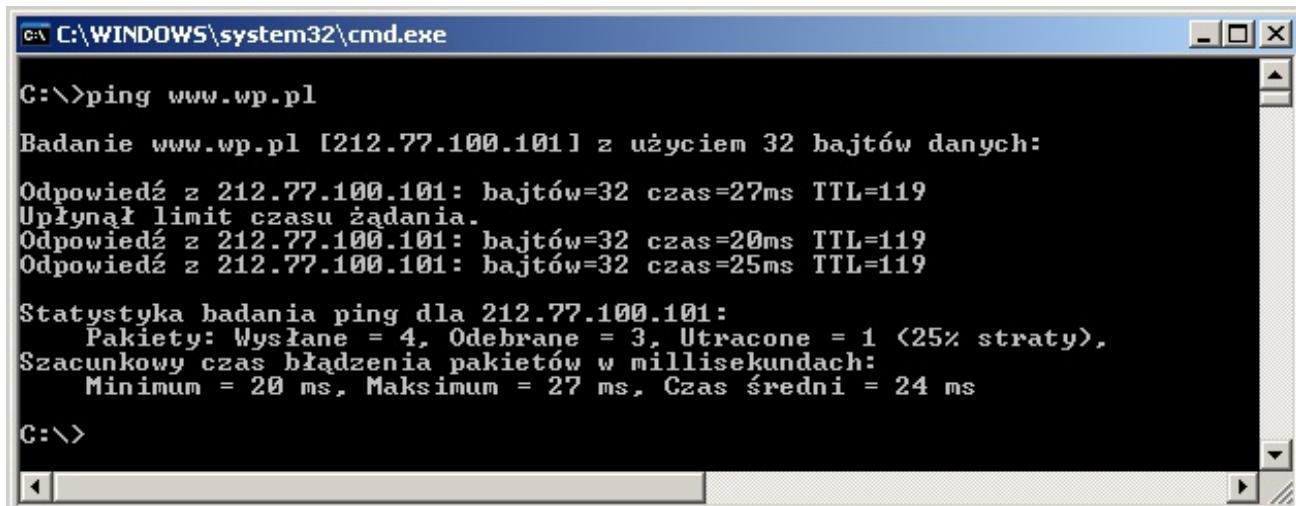
Po wykonaniu polecenia a opcję all, system wyświetla informacje ogółem oraz następujące informacje szczegółowe dla każdego z interfejsów sieciowych:

- Stan interfejsu – jeżeli nieaktywny;
- Informacja na temat karty sieciowej, opis oraz adres MAC wykorzystywany w warstwie 2 modelu OSI;
- Informacja, czy karta sieciowa konfigurowana jest ręcznie, czy jest pobierana z serwera DHCP;
- Adres IP interfejsu sieciowego;
- Maska sieci lub podsieci;
- Domyślna brama (tam trafiają wszystkie pakiety kierowane na adres spoza sieci, w której się znajduje komputer);
- Adres serwera DHCP – tylko w przypadku konfiguracji dynamicznej;/LI>

- Adres podstawowego i zapasowego serwera DNS;
- Informacje o dzierżawie adres IP (po wygaśnięciu nastąpi ponowna autokonfiguracja za pomocą DHCP) – tylko w przypadku konfiguracji dynamicznej.

DIAGNOSTYKA

Podstawowym narzędziem do diagnostyki sieci jest program ping. Program ten wykorzystuje protokół ICMP i wysyła do adresu docelowego zapytanie (ping) o stan urządzenia. Jeżeli urządzenie zlokalizowane pod docelowym adresem sieciowym działa prawidłowo i ma włączoną obsługę ICMP odpowiedzie (pong) o swoim prawidłowym stanie, co objawi się komunikatem "Odpowiedź z adres_docelowy: ...". Jeżeli wystaje problem z siecią lub w urządzeniu docelowym otrzymamy komunikat o błędzie (urządzenie docelowe zwróci odpowiedni kod błędu) lub komunikat o przekroczonym czasie oczekiwania na odpowiedź. Przyczyn takiej odpowiedzi może być wiele ale daje nam informacje o poprawności działania sieci. Po skończeniu działania program zwraca statystykę prawidłowych i nieprawidłowych odpowiedzi adresu docelowego. Program ping wbudowany jest w każdy system operacyjny, może się jedynie nieco różnić dodatkowymi możliwościami czy postacią zwracanych komunikatów.



```
C:\>ping www.wp.pl

Badanie www.wp.pl [212.77.100.101] z użyciem 32 bajtów danych:

Odpowiedź z 212.77.100.101: bajtów=32 czas=27ms TTL=119
Upłynął limit czasu żądania.
Odpowiedź z 212.77.100.101: bajtów=32 czas=20ms TTL=119
Odpowiedź z 212.77.100.101: bajtów=32 czas=25ms TTL=119

Statystyka badania ping dla 212.77.100.101:
  Pakiety: Wysłane = 4, Odebrane = 3, Utracone = 1 (25% straty),
  Szacunkowy czas błądzenia pakietów w milisekundach:
    Minimum = 20 ms, Maksimum = 27 ms, Czas średni = 24 ms

C:\>
```

Przykładowy wynik wykonania polecenia ping.

Drugim programem przeznaczonym do diagnozowania sieci jest program do śledzenia trasy wysłanego przez nas pakietu. W systemie Windows jest to program tracert, w systemach rodziny Unix program traceroute. Programy te zwracają komunikat przy każdym przejściu przez węzeł sieciowy wyświetlając podstawowe informacje o węźle jak również czasy przejść pomiędzy węzłami. Jest to przydane narzędzie dla administratorów dużych sieci, które pozwala zweryfikować poprawność tras pakietów (może okazać się, że wybierana przez routery trasa jako najlepsza jest tak naprawdę wysoce nieoptymalna).

```
C:\>tracert www.onet.pl
Trasa śledzenia do www.onet.pl [213.180.130.200]
przewyższa maksymalną liczbę przeskoków 30

 1      4 ms      1 ms      1 ms  192.168.0.1
 2      5 ms      6 ms      7 ms  254-tor-4.acn.waw.pl [62.121.71.254]
 3      7 ms      6 ms      8 ms  Tor1-do-TorCORE.net.aster.pl [212.76.35.85]
 4     23 ms      8 ms     10 ms  TorCORE-do-MiaCORE.net.aster.pl [212.76.35.65]
 5      *      8 ms      *      MiaCORE-do-DomCORE.net.aster.pl [212.76.35.49]
 6      *      * ms      *      Upłynął limit czasu żądania.
 7     20 ms     13 ms     12 ms  v100.rtr-head02.autocom.pl [213.134.161.246]
 8     24 ms     13 ms     21 ms  do-onet.rtr-head02.autocom.pl [213.134.162.22]
 9     18 ms     13 ms     14 ms  dab2v7.onet.pl [213.180.143.34]
10     15 ms     14 ms     20 ms  f1virt.onet.pl [213.180.130.200]

Śledzenie zakończone.

C:\>
```

Wynik przykładowego wykonania programu tracert pod Windows.

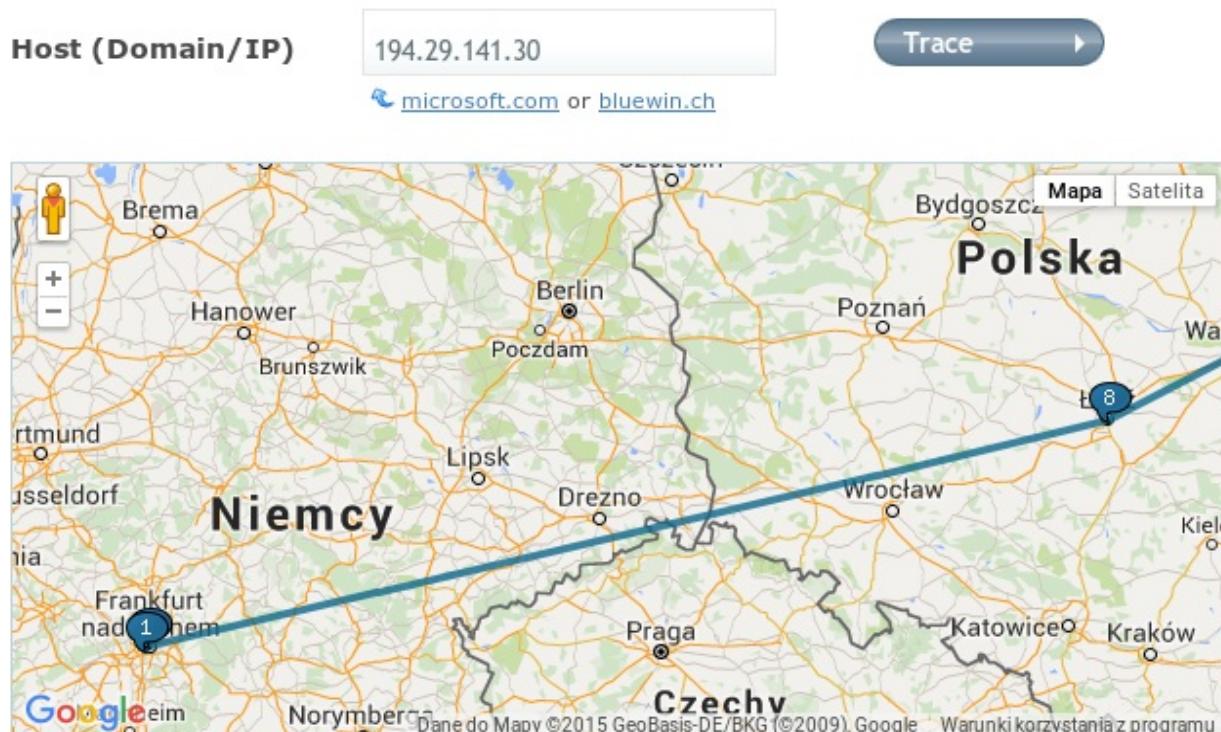
W Internecie dostępne są narzędzia do graficznej prezentacji trasy badanego pakietu. Przykładem może być chociażby witryna <http://en.dnstools.ch/visual-traceroute.html>. Takich stron znajdziecie wiele - wystarczy w wyszukiwarce internetowej podać hasło *visual traceroute*.

Traceroute on a map

About

Traceroute determines which IP-Router the data packets take to get to the target computer. However, traceroute does not always show the actual route. The result may be influenced by firewalls, flawed implementation of IP-stacks, Network Address Translation and IP tunnels.

Parallel to the traceroute query, locations of the nodes are also determined and represented on the map.



Terminal — bash

```
guest@dnstools.ch:~> traceroute 194.29.141.30
1 gw.f.netclusive.de (89.110.131.1) 0.685 ms
2 89.202.113.81 (89.202.113.81) 2.131 ms
3 te2-1-111.bg1.eu.equinix.net (217.68.155.165) 0.275 ms
4 xe-8-2-0.edge4.Frankfurt1.Level3.net (212.162.24.73) 0.485 ms
5 ae-21-3204.car1.Berlin1.Level3.net (4.69.161.6) 164.100 ms
6 ae-21-3204.car1.Berlin1.Level3.net (4.69.161.6) 164.004 ms
7 212.162.10.82 (212.162.10.82) 46.742 ms
8 z-poznan-gw3.nask.10Gb.rtr.pionier.gov.pl (212.191.224.74) 24.208 ms
9 welcome-at.pw.edu.pl (148.81.253.70) 24.283 ms
10 194.29.132.162 (194.29.132.162) 24.600 ms
11 194.29.129.59 (194.29.129.59) 31.708 ms
12 coi-mchtr.rtr.pw.edu.pl (194.29.132.130) 24.472 ms
13 coi-mchtr.rtr.pw.edu.pl (194.29.132.130) 24.437 ms
```

Przykładowa trasa przebiegu pakietu przedstawiona na tle mapy świata.

KONFIGURACJA PRZYKŁADOWEGO ROUTERA TCP/IP

W tej sekcji postaramy się przybliżyć konfigurację prostego routera opartego o połaczenie ADSL. Wybrany został popularny model D-Link DI-524. Ten router przeznaczony do domowego użytku zawiera łącze WAN typu ADSL (podłączane kablem typu skrętka), 4 porty sieci Ethernet 100 Mb/s oraz access point sieci bezprzewodowej w standardzie 802.11g o szybkości 54 Mb/s.

Połączenia tego typu są wykorzystywane np. przez telewizje kablowe dostarczając sygnał przez kabel telewizyjny. Podłączając odpowiednie urządzenie filtrujące (modem) otrzymane od dostawcy Internetu dostajemy skrętkę z zalożonym złączem RJ45. Można ten kabel podłączyć bezpośrednio do karty sieciowej naszego komputera i od razu możemy się cieszyć z posiadania Internetu. Co zrobić w przypadku, gdy mamy więcej niż jeden komputer w domu i chcemy aby wszystkie miały dostęp do Internetu? Być może jednym z komputerów jest laptop, którego chcemy używać w różnych pokojach mieszkania, nie koniecznie co chwilę przełączając lub przedłużając kabel.

Z rozwiązaniem przychodzą firm produkujące proste wielofunkcyjne urządzenia sieciowe. Router ADSL z Wireless D-Link DI-524 to koszt rzędu 150-250 złotych. Nieduży zważywszy na możliwości, które nam daje. Za pomocą tego typu urządzenia jesteśmy w stanie stworzyć niewielką sieć domową z 4 hostami podłączonymi za pomocą kabla skrętki oraz kolejnymi za pomocą łączności bezprzewodowej. Zasięg transmisji bezprzewodowej nie jest może bardzo duży ale na potrzeby pokrycia mieszkania czy domu powinien w zupełności wystarczyć.

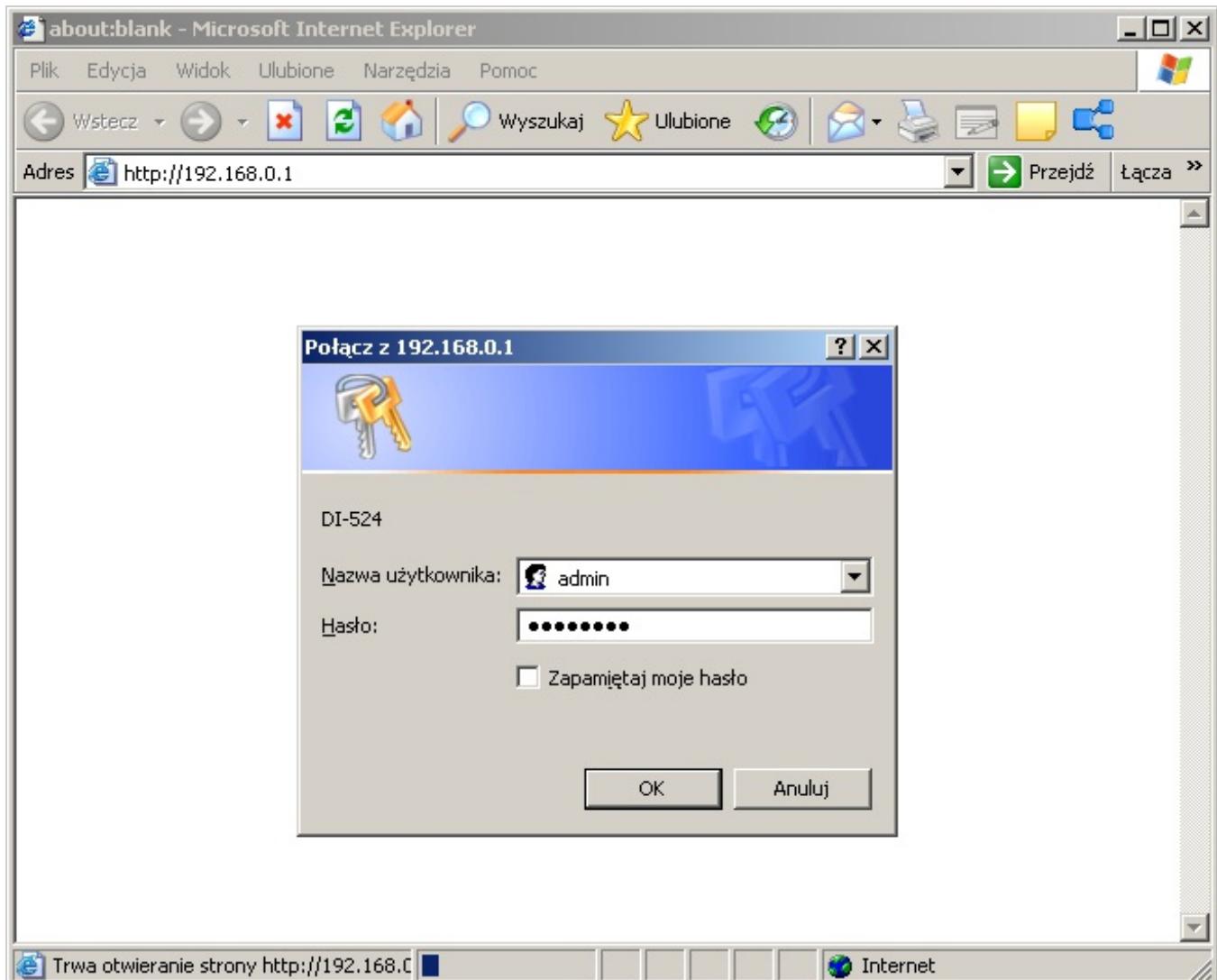


Router D-Link DI-524.

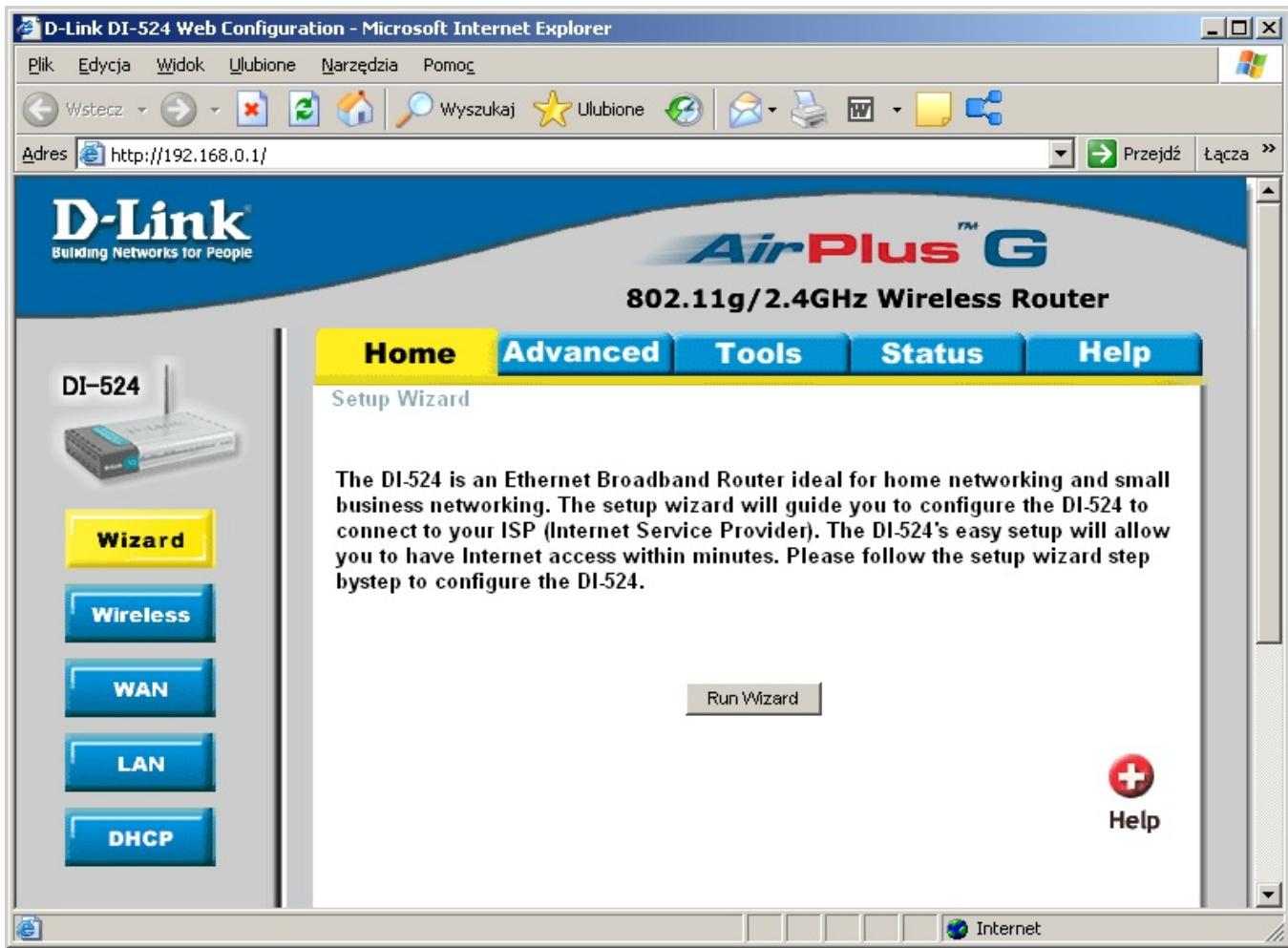
Aby skonfigurować naszą sieć domową, musimy połączyć się z routерem i ustawić jego parametry. Aby otworzyć konfigurację routera należy podłączyć się do niego za pomocą kabla typu skrętka (sieć bezprzewodowa domyślnie nie jest włączona ponieważ byłoby to niebezpieczne zakładając, że przy pierwszym uruchomieniu routera nie ma on zdefiniowanych haseł zabezpieczających).

Oprogramowanie konfiguracyjne routera oparte jest na stronach WWW. Należy zatem uruchomić przeglądarkę internetową i wpisać domyślnie ustawiony adres interfejsu routera: <http://192.168.0.1>

Pojawi się monit o zalogowanie administratora routera. Nazwa użytkownika to admin, hasło standardowe, podane w instrukcji do urządzenia. Po pierwszym zalogowaniu należy zmienić hasło administratora na inne niż standardowe! W momencie uruchomienia sieci bezprzewodowej bez zmiany hasła grozi nam przejęcie routera przez osobę trzecią.



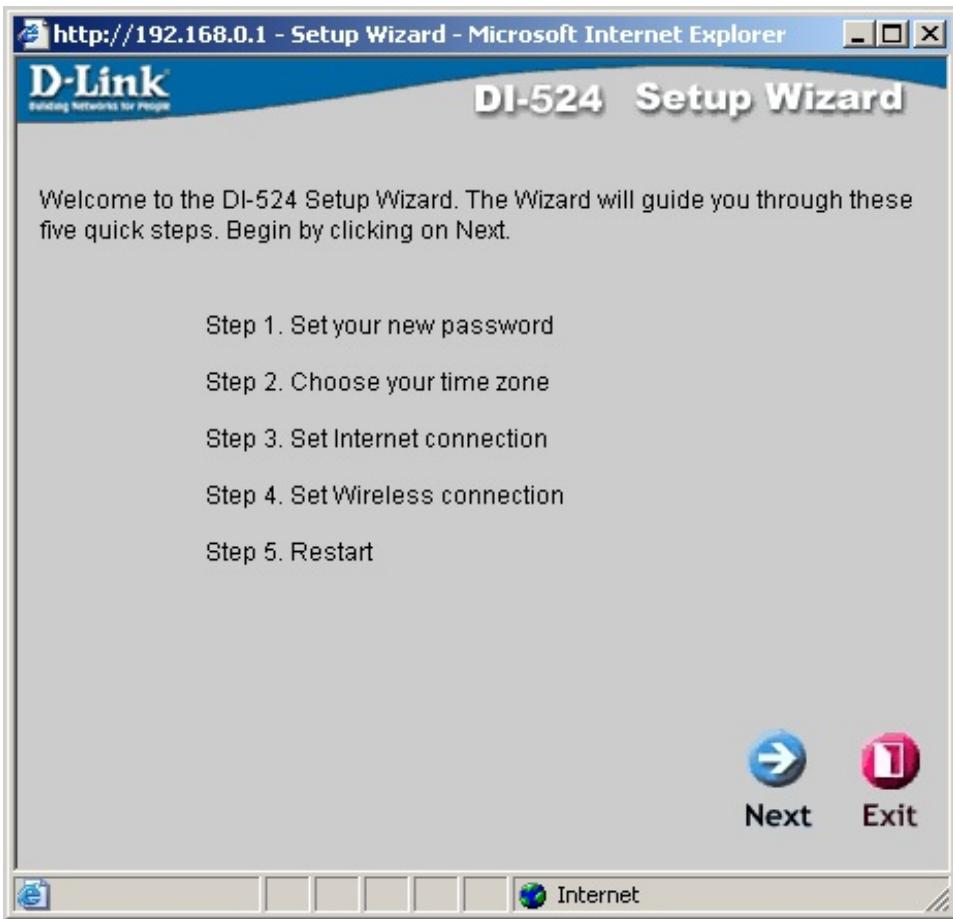
Jeżeli wpisaliśmy prawidłową nazwę użytkownika oraz hasło, zostaniemy przekierowani na stronę główną obsługi urządzenia. W górnej części strony oprócz informacji dotyczących modelu routera mamy belkę z dostępnymi opcjami oprogramowania. Po lewej stronie dla bieżącej opcji wyświetlane są możliwe do wykonania operacje.



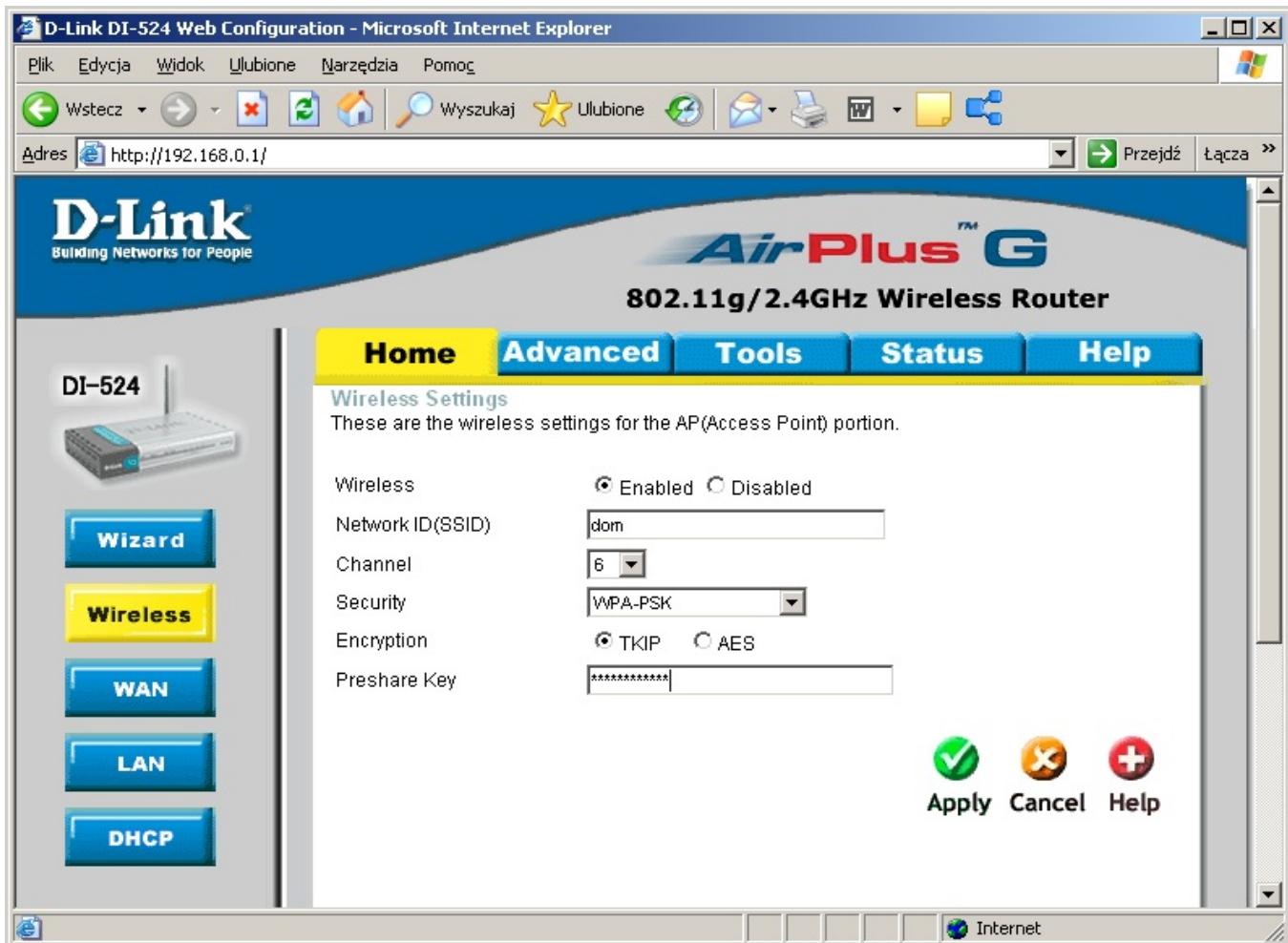
Strona startowa routera oferuje nam krótki opis urządzenia oraz przycisk "Run Wizard", który uruchomi tzw. Wizard konfiguracji urządzenia. Wizard to uruchamiane w ustalonej kolejności mniejsze kroki większej konfiguracji. Wizard konfiguracji routera zawiera następujące kroki:

- Ustanowienie nowego hasła użytkownika;
- Ustawienie strefy czasowej i czasu urządzenia;
- Konfiguracja połączenia z Internetem (port WAN);
- Konfiguracja sieci bezprzewodowej;
- Zapisanie konfiguracji i restart urządzenia.

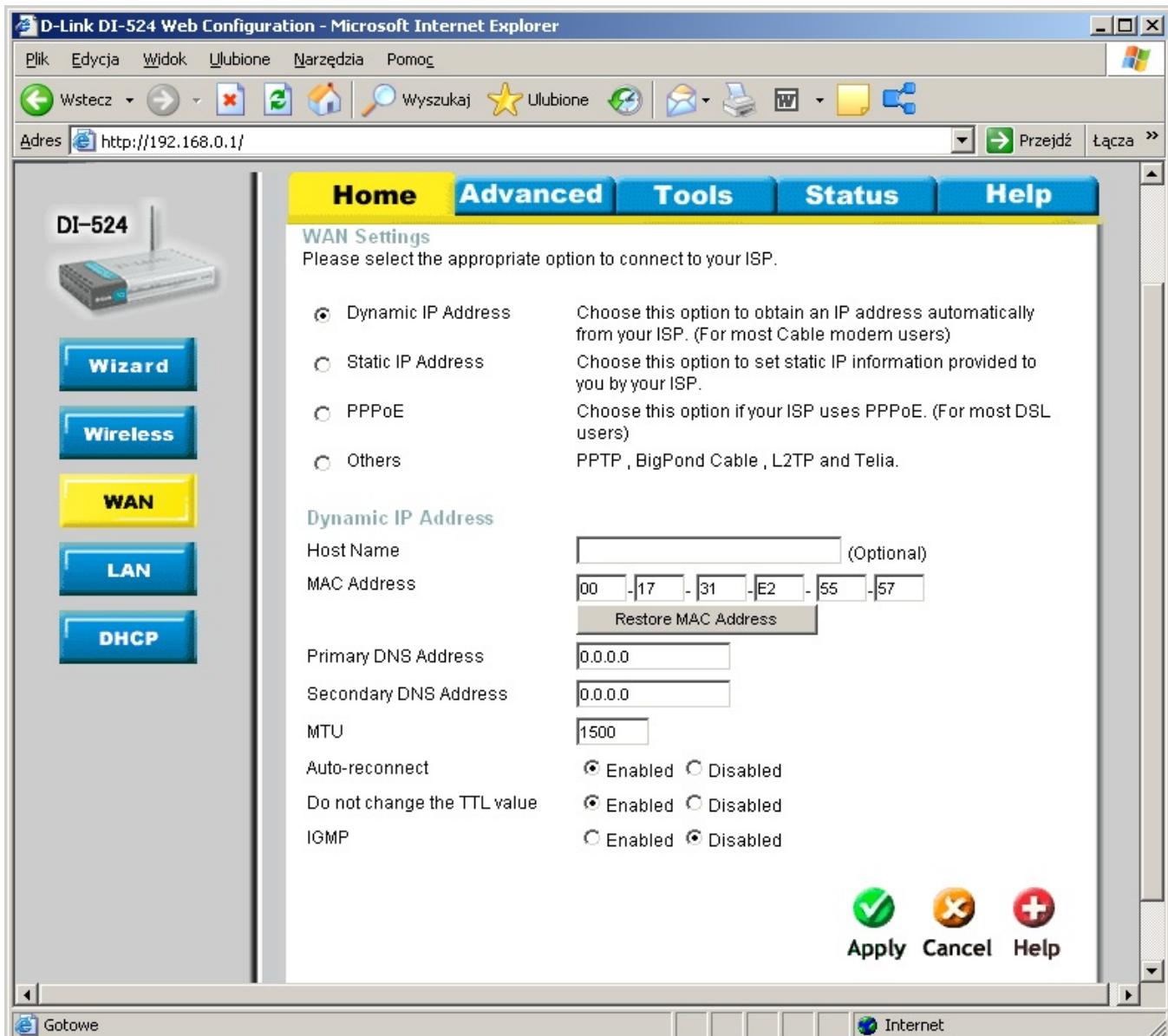
W przypadku podstawowej konfiguracji przy użyciu Wizarda, nie uwzględniono konfiguracji portów Ethernet ponieważ są tak ustawione aby połączenie do nich hosta automatycznie nadawało adres IP a więc ich standardowe działanie w większości przypadków powinno być prawidłowe.



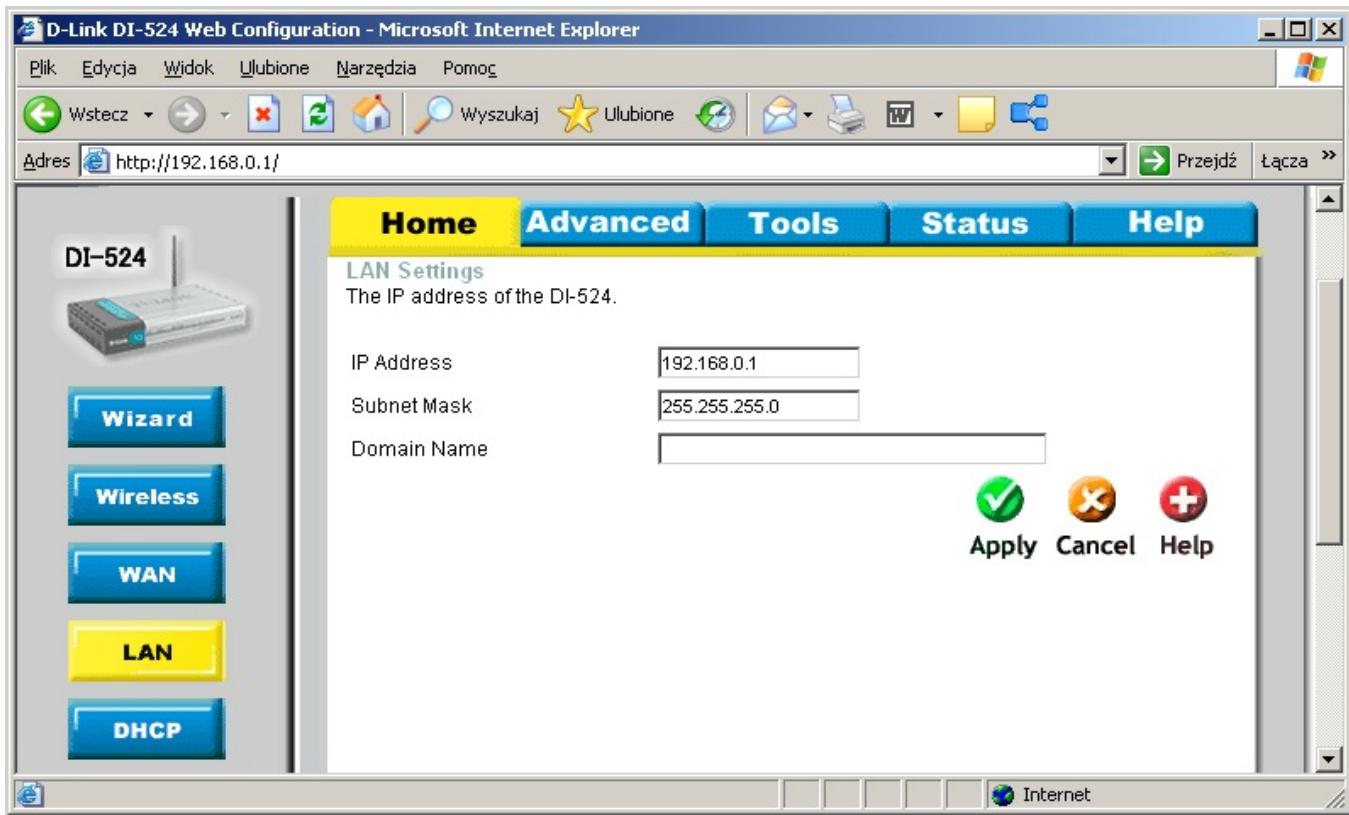
Dla najprostrzej z możliwych wariantów sieci, konfiguracja urządzenia za pomocą Wizarda powinna być wystarczająca. Jeżeli jednak chcemy ustawić dodatkowe parametry naszego routera, należy to zrobić ręcznie, przechodząc przez kolejne opcje dostępne w interfejsie routera.



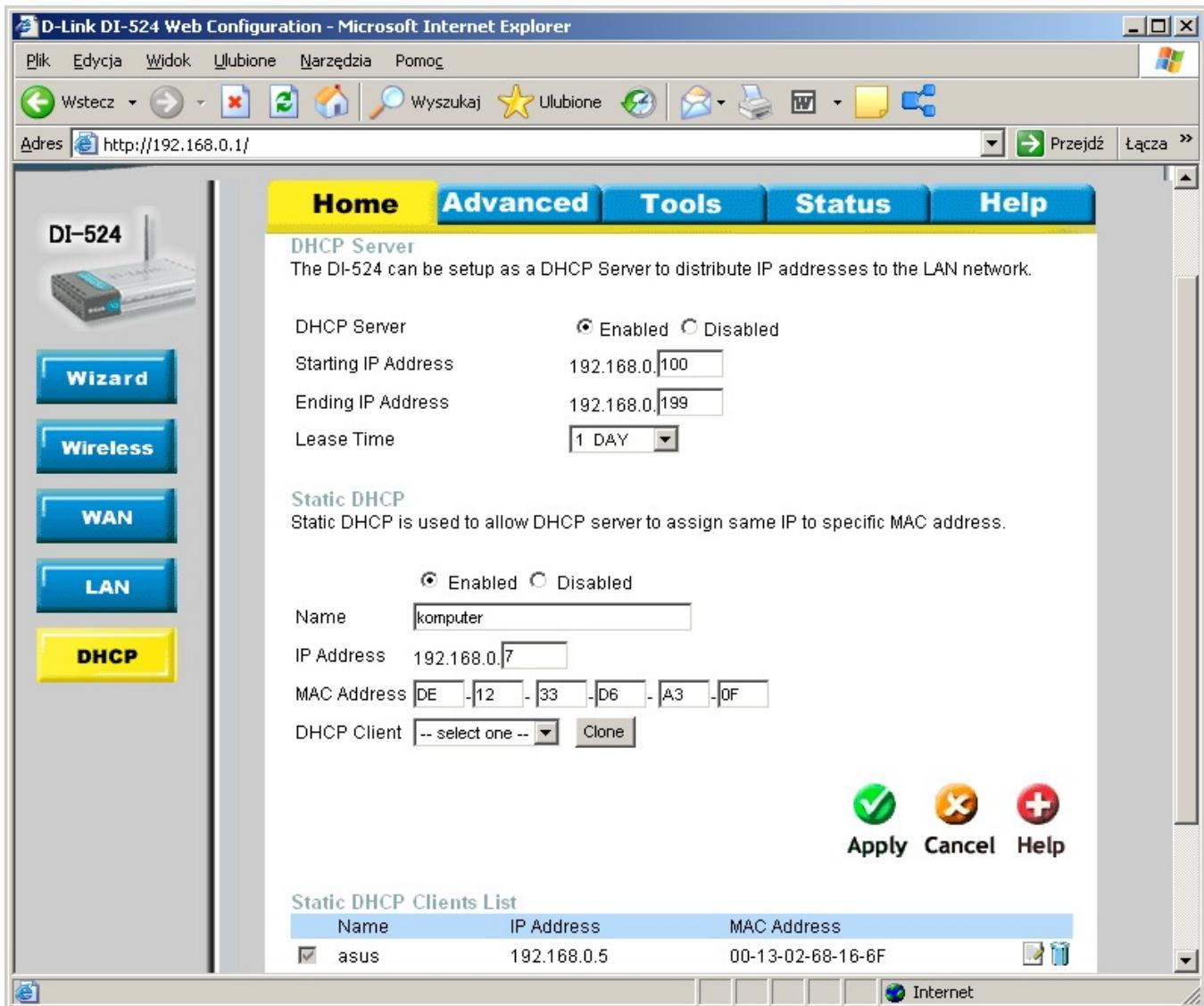
Pod przyciskiem **Wireless** zakładki **Home** dostępna jest podstawowa konfiguracja sieci bezprzewodowej. Możemy przede wszystkim włączyć lub wyłączyć działanie sieci WiFi w naszym routerze. Oprócz tego możemy nadać identyfikator sieci SSID, ustawić kanał transmisji, wybrać rodzaj zabezpieczenia sieci i metodę szyfrowania oraz wprowadzić hasło sieciowe, które będzie trzeba wpisać na komputerze hostów przy pierwszym połączeniu się z naszą siecią. Na tej jak i każdej innej stronie, zmiany zapisujemy za pomocą przycisku **Apply** a wycofujemy za pomocą przycisku **Cancel**.



Pod przyciskiem **WAN** zakładki **Home** dostępna jest konfiguracja łącza do Internetu a więc jedynego portu WAN. Mamy możliwość wybrania metody nadawania adresu IP i pozostałych ustawień naszemu interfejsowi sieciowemu po stronie WAN. Zwykle będziemy chcieli aby był on dynamicznie przydzielany przez naszego dostawcę usług telekomunikacyjnych. W tym celu zaznaczamy opcję **Dynamic IP Address** w górnej sekcji strony. Adres IP przyznawany jest przez DHCP naszego prowidera na podstawie adresu MAC otrzymanej przez nas karty sieciowej lub modemu. Jeżeli sieć jest zabezpieczona wymagana może być podmiana adresu MAC interfejsu WAN routera na taki sam, jaki posiada otrzymana przez nas karta sieciowa. W przypadku wybrania opcji **Static IP Address** będziemy zmuszeni wprowadzić wszystkie ustawienia ręcznie.



Kolejna strona, kryjąca się pod przyciskiem **LAN** zakładki **Home** pozwala na ustawienie głównego interfejsu sieci lokalnej (zarówno Ethernet jak i Wireless). W tym miejscu możemy utworzyć podsieć wpisując adres i maskę podsieci zamiast standardowy adres i maskę dla sieci 192.168.0.0. W tym miejscu skonfigurować możemy również nazwę domeny wewnątrz sieci.

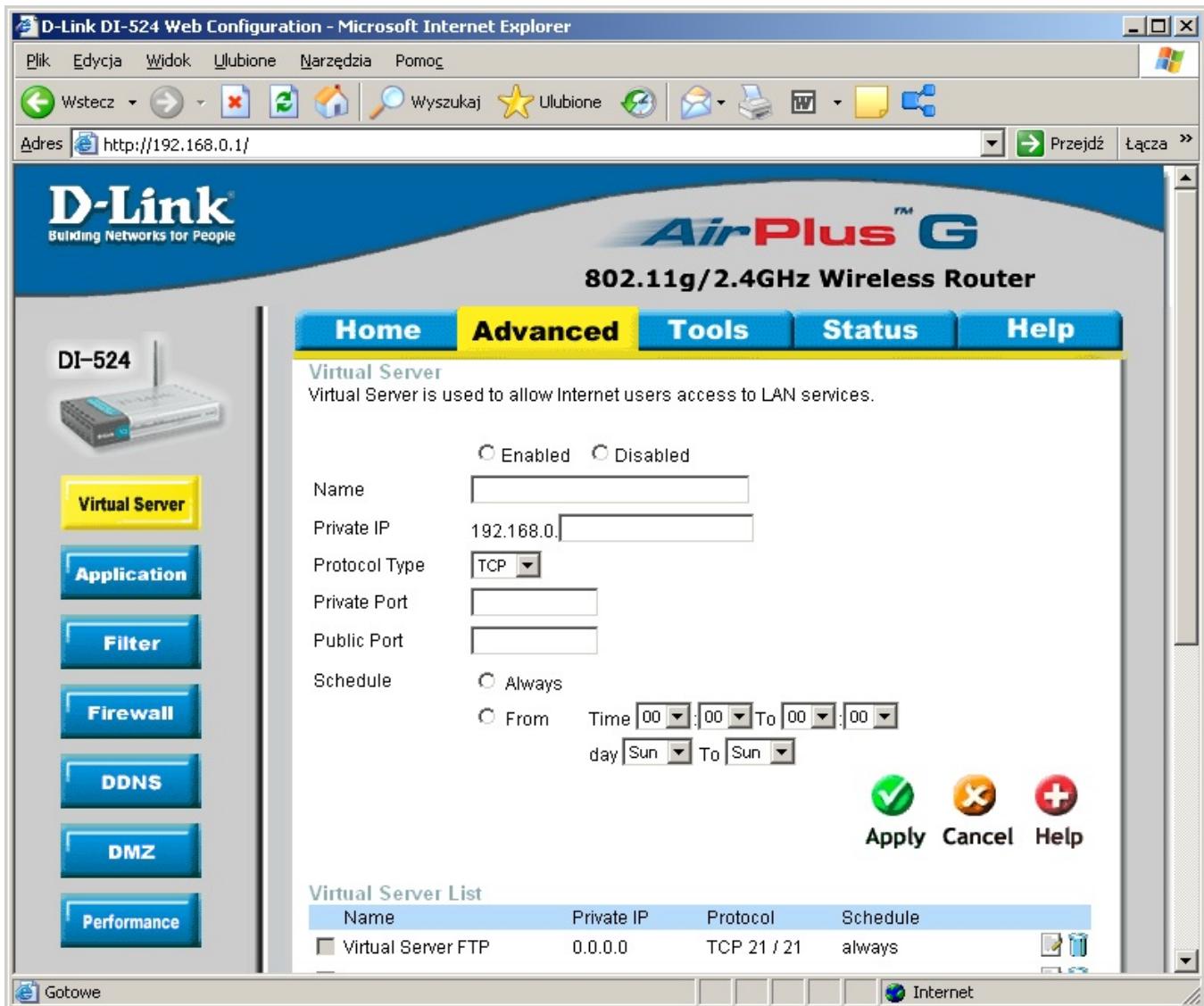


Konfiguracja wbudowanego w router serwera DHCP dostępna jest pod przyciskiem **LAN** zakładki **Home**. Konfiguracja ta dotyczy przydzielania adresów IP hostom znajdującym się wewnętrz naszej sieci lokalnej (zarówno Ethernet jak i WiFi).

W górnej części strony możemy włączyć lub wyłączyć działanie serwera DHCP, ustawić zakres dynamicznie przydzielanych adresów ID oraz okres dzierżawy adresu (domyślnie jeden dzień). Po tym czasie adres IP hosta zostanie pobrany ponownie z naszego serwera DHCP.

Poniżej znajdują się formularz do konfiguracji stałego przydzielania adresów IP na podstawie adresów MAC urządzeń sieciowych hostów. Skonfigurowanie wpisu dla jakiegoś komputera wewnętrz sieci spowoduje przydzielanie temu komputerowi zawsze tego samego, ustalonego adresu IP. Jeżeli nie będzie tu wpisu dla danego komputera, zostanie on obsłużony przez mechanizm dynamicznego (losowego) przydzielania adresu z ustalonej wcześniej puli, przez co po każdym uruchomieniu komputera lub wygaśnięciu dzierżawy, komputer może otrzymać inny adres IP. Sytuacja taka może być niepożądana. W dolnej części strony widnieją zdefiniowane wpisy dla statycznego przydzielania adresów IP jak również bieżące dynamicznie przydzielone adresy z puli. Należy zwrócić uwagę, że konfiguracja ta dotyczy zarówno sieci Ethernet jak i sieci bezprzewodowej, ponieważ z punktu widzenia serwera DHCP interfejsy dostępu do tych sieci są identyczne. Każdy ma unikalny adres MAC, na podstawie którego można przydzielić odpowiedni adres IP.

Na tym kończy się konfiguracja podstawowych parametrów routera, pozwalająca mu w pełni funkcjonalnie działać. Dla administratora urządzenia dostępne są dodatkowe funkcje wspierające. Te funkcje opiszemy ogólnie, nie wchodząc w szczegóły poszczególnych opcji.



Funkcja **Virtual Server** zakładki **Advanced** pozwala na skonfigurowanie tunelu sieć WAN - host w sieci wewnętrznej. Umożliwia to udostpnenie w sieci na zewnątrz usługi uruchomionej na którymś z hostów w sieci (a nie na serwerze jak to zwykle bywa). Jest to dosyć ciekawa funkcjonalność, w rzeczywistych warunkach raczej rzadko wykorzystywana.

Funkcja **Application** zakładki **Advanced** pozwala na skonfigurowanie otwierania wielu portów na potrzeby działania aplikacji.

Funkcja **Filter** zakładki **Advanced** pozwala na filtrowanie ruchu pomiędzy siecią wewnętrzną a Internete. Filtrowanie może odbywać się na podstawie adresów IP, adresów URL, adresów MAC oraz nazw domen. Można również zablokować wybrane adresy IP sieci wewnętrznej w taki sposób, aby w ogóle nie miały dostępu do Internetu.

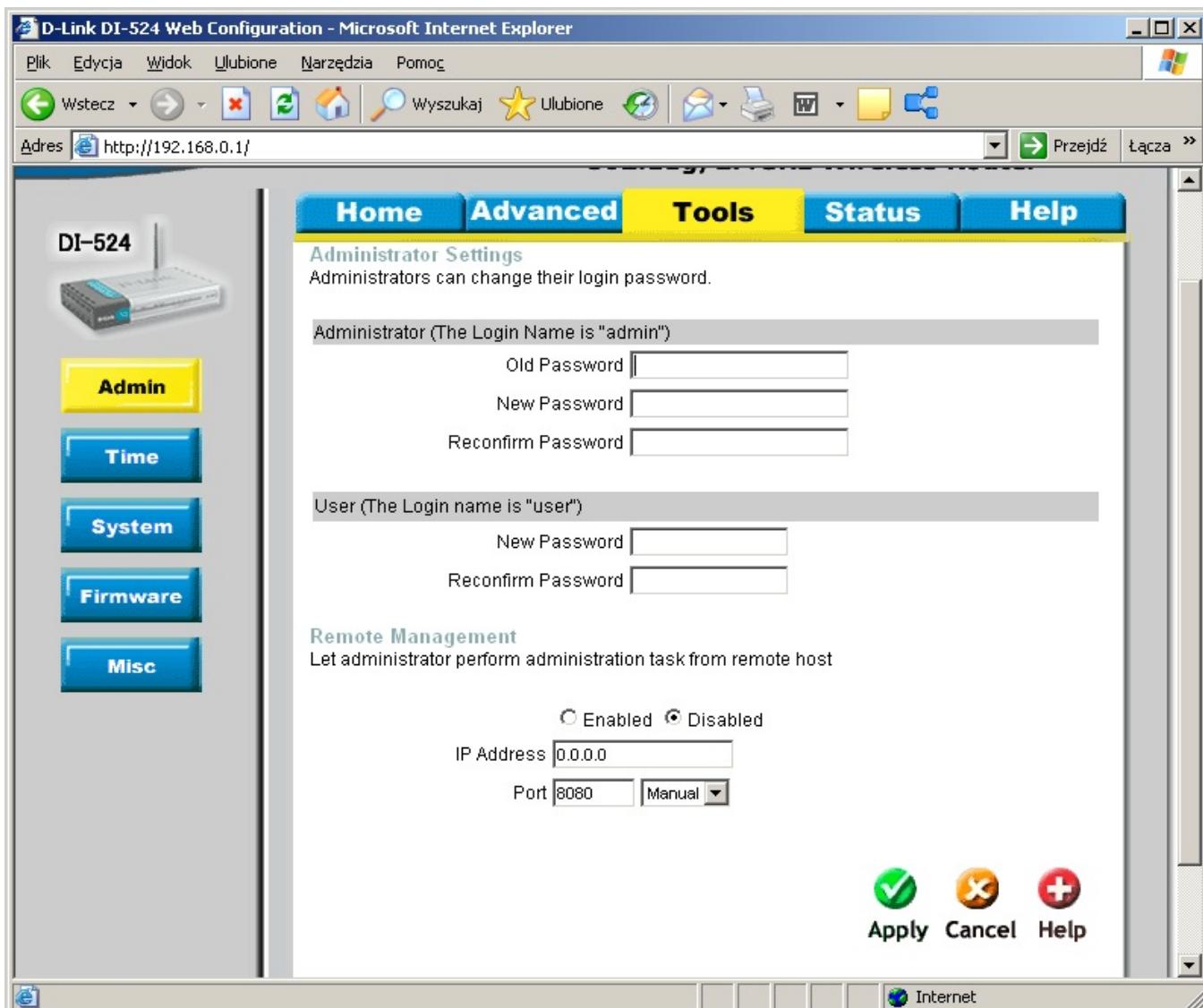
Funckja **Firewall** zakładki **Advanced** pozwala na skonfigurowanie zabezpieczeń typu "zapora ogniodziałająca" na routerze. Możemy filtrować (dopuszczać lub odrzucać) przychodzące i wychodzące pakiety na podstawie protokołów, portów i adresów IP.

Funckja **DDNS** (ang. *Dynamic DNS*) zakładki **Advanced** pozwala na skonfigurowanie nazwy domeny dla naszej sieci w wypadku, gdy nasz zewnętrzny adres IP nie jest stały. Usługi takie są oferowane przez serwery w Internecie.

Funckja **DMZ** (ang. *Demilitarized Zone*) zakładki **Advanced** - strefa zdemilitaryzowana pozwala na całkowite "wystawienie" wybranego komputera lokalnego na świat. Oznacza to przekazywanie wszystkich pakietów do z tego komputera, w sposób jakby był on bezpośrednio połączony do sieci.

Internet.

Funckja **Performance** zakładki **Advanced** pozwala na ustawienie parametrów routera wpływających na jego działanie i wydajność.



Funckja **Admin** zakładki **Tools** umożliwia zmianę hasła administratora routera oraz dodatkowego konta użytkownika z ograniczonymi uprawnieniami. Oprócz tego na tej stronie można uaktywnić możliwość zdalnego dostępu do panelu administratora z sieci zewnętrznej. W tym celu należy podać adres IP komputera, z którego będzie można się połączyć z routerem oraz port.

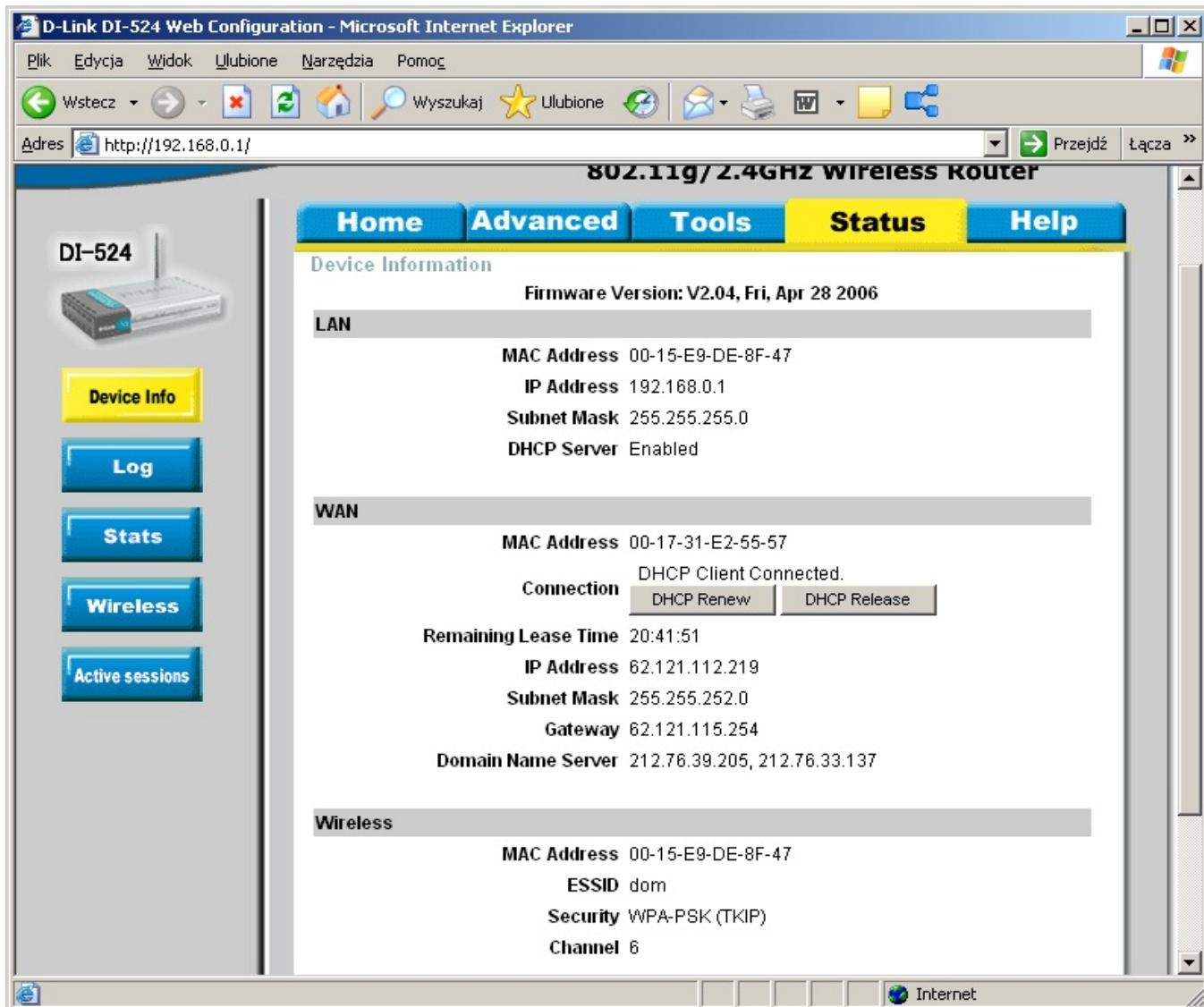
Funckja **Time** zakładki **Tools** umożliwia ręczne wprowadzenie aktualnego czasu routera oraz skonfigurowanie go, aby wykorzystywał protokół NTP do synchronizacji z serwerem czasu. W tym celu należy podać adres serwera NTP oraz strefę czasową, w jakiej znajduje się urządzenie.

Funckja **System** zakładki **Tools** umożliwia zapisanie wykonanej konfiguracji routera do pliku na komputerze, z którego łączymy się z panelem administratora, wgranie konfiguracji z pliku oraz odtworzenie ustawień fabrycznych (domyślnych).

Funckja **Firmware** zakładki **Tools** pozwala na upgrade oprogramowania urządzenia. Często nowszą wersję oprogramowania (tzw. firmware'u) można pobrać ze strony producenta. Nowsza wersja oprogramowania może zawierać poprawki wykrytych błędów oprogramowania oraz może na przykład bardziej wydajnie zarządzać urządzeniem.

Funckja **Misc** zakładki **Tools** umożliwia m.in. wykonanie programu ping z routera do dowolnego hosta z sieci lokalnej lub zewnętrznej oraz skonfigurowanie blokowania odpowiedzi na zapytania ping

kierowane do routera z innych komputerów. Na stronie tej dostępna jest również opcja restartu routera.



Funckja **Device info** zakładki **Status** prezentuje podstawowe informacje dotyczące urządzenia: wersję i datę wypuszczenia oprogramowania routera, ustwienia sieci LAN, ustawienia portu zewnętrznego WAN (oraz opcje odświeżenia i zwolnienia adresu jeżeli otrzymany został przez DHCP), ustawienia sieci Wireless oraz aktualną datę i godzinę urządzenia.

Funckja **Log** zakładki **Status** oferuje zapisane komunikaty dotyczące zdarzeń związanych z ruchem TCP/IP występujących na routerze. Są to między innymi nieautoryzowane próby dostępu do interfejsów sieciowych routera.

Funckja **Stats** zakładki **Status** zawiera statystyki odebranych i wysłanych pakietów przechodzących przez router. Administrator może na bieżąco odświeżać statystyki oraz zresetować je, aby były zliczane od zera.

Funckja **Wireless** zakładki **Status** prezentuje aktywne sesje bezprzewodowe na routerze, czyli podłączone za pomocą WiFi komputery sieci lokalnej. Strona prezentuje informacje o adresie MAC połączonego hosta oraz moment połączenia z routera.

Funckja **Active sessions** zakładki **Status** przedstawia wszystkie aktywne sesje zestawionych połączeń pomiędzy komputerami znajdującymi się w sieci wewnętrznej, z komputerami znajdującymi się po stronie portu WAN routera. Te połączenia musi utrzymywać stosując mechanizm NAT.

Zakładka **Help** zawiera stronę pomocy dla każdej z wymienionych opcji.

Indeks