Wpływ architektury procesora na system operacyjny

Prezentację przygotowali:

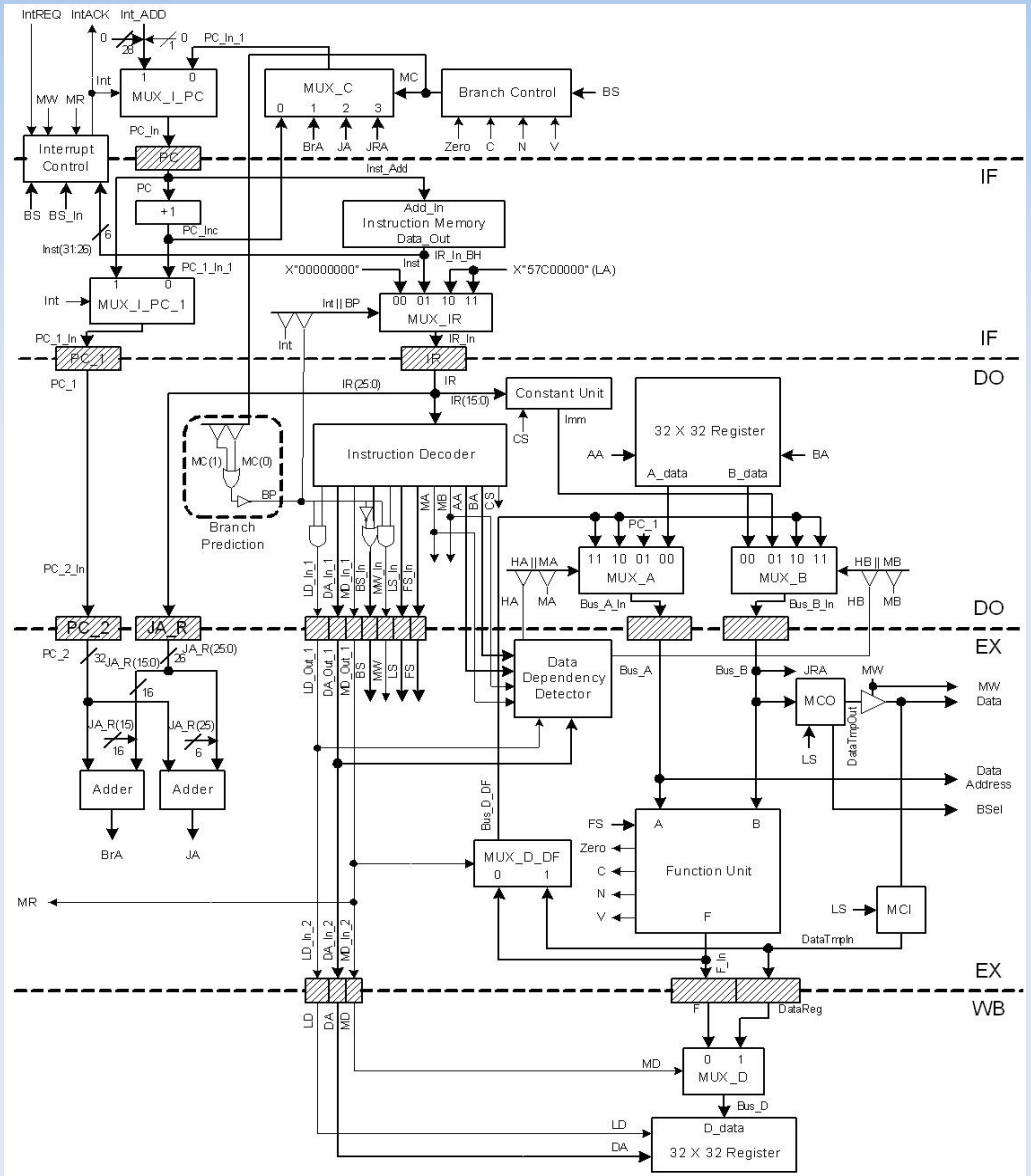
Łukasz Jagielski **Maciej Pazurkiewicz**

Treść prezentacji

- ogólnie o procesorach
- procesory CISC a RISC
- przetwarzanie potokowe
- wielordzeniowość
- 64-bitowość
  - ogólnie o architekturach 64-bitowych
  - x86-64 jako powszechna architektura 64-bitowa
  - 64-bitowe systemy operacyjne

Wpływ architektury procesora na system operacyjny

Procesor



Podział procesorów

- długość słowa maszynowego
- liczba rdzeni
- architektura
- przeznaczenie
- taktowanie zegara
- wielkość cache

Procesor a system operacyjny

- system operacyjny budowany jest z myślą o konkretnej platformie sprzętowej
- popularność konkretnych procesorów zależy w głównej mierze od tego, jakie systemy operacyjne będą na nich działać
- procesor wpływa przede wszystkim na szybkość działania systemu operacyjnego

Charakterystyka CISC

- duży zbiór instrukcji
- dużo (nieużywanych) trybów adresowania
- kody instrukcji o zmiennej długości
- wyspecjalizowane rejestry
- łatwiejsze kodowanie w językach asemblerowych

Przykłady CISC

- VAX
- PDP-11
- Motorola 68000
- x86

Charakterystyka RISC

- mała liczba instrukcji
- proste tryby adresowania
- kod instrukcji mieszczący się w słowie procesora
- dużo uniwersalnych rejestrów procesora
- jedynie store i load na danych w pamięci
- potokowanie

Przykłady RISC

- CDC 6600 (74 instrukcje)
- RISC-I, RISC-II (32/39 instrukcji)
- MIPS - PlayStation, PlayStation 2, Nintendo 64, PSP
- ARM - Apple iPhone, iPod, GameBoy Advance, niektóre telefony Nokia i Sony Ericsson
- Power – Macintosh, Xbox 360, PlayStation 3

Wpływ architektury procesora na system operacyjny

Przetwarzanie skalarne

źródło http://eng.wikipedia.org

Przetwarzanie potokowe

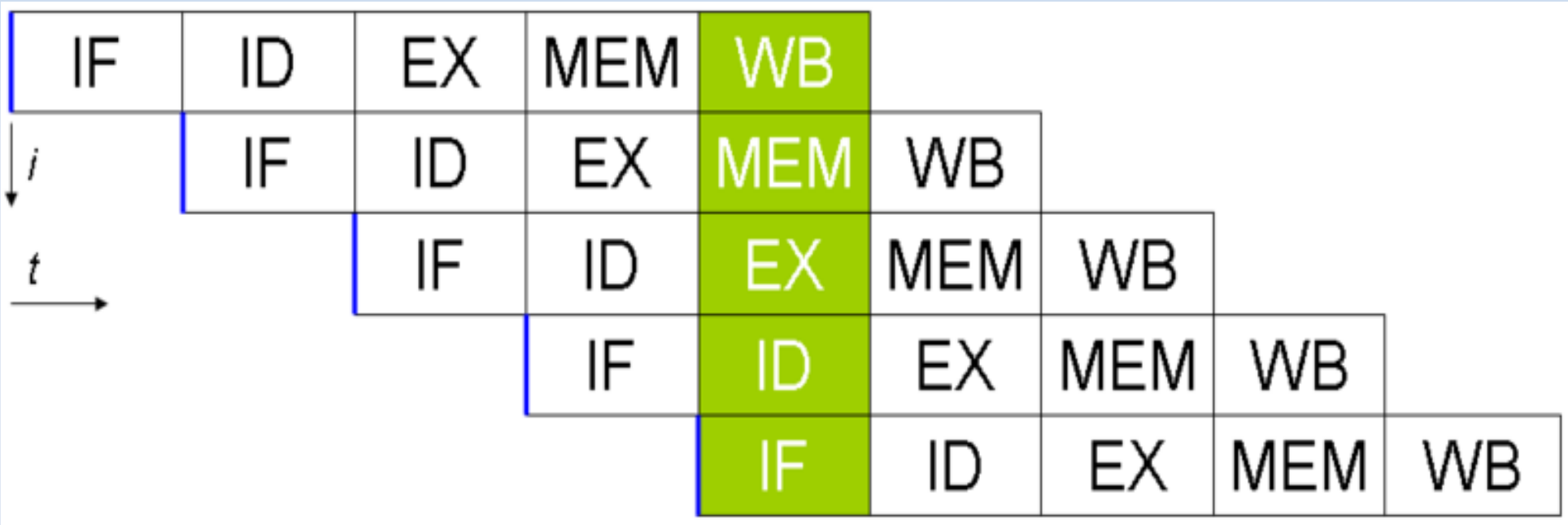
- technika skracająca czas wykonania ciągu instrukcji
- polega na podziale instrukcji na niezależne fazy, które mogą być wykonywane jednocześnie dla osobnych instrukcji
- zapoczątkowana w procesorach RISC

Klasyczne potokowanie RISC

- instruction fetch (pobanie instrukcji)
- instruction decode (interpretacja instrukcji)
- execute (wykonanie obliczeń, obliczenie adresu...)
- memory access (np. odczyt danych dla load)
- write back (zapis wyniku)

Wpływ architektury procesora na system operacyjny

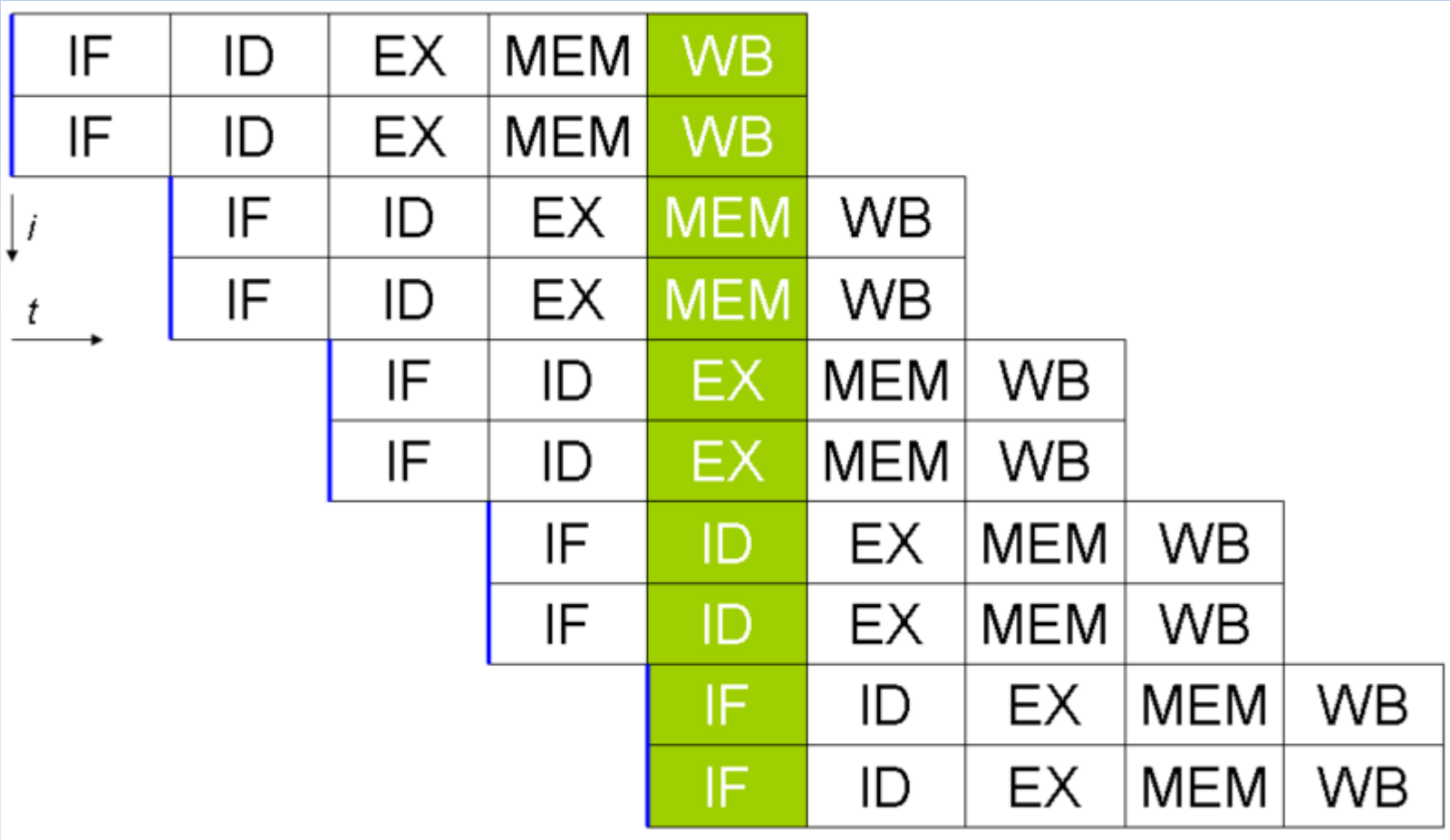
Klasyczne potokowanie RISC



źródło http://eng.wikipedia.org

Wpływ architektury procesora na system operacyjny

Przetwarzane superskalarne



źródło http://eng.wikipedia.org

Hazardy przy przetwarzaniu potokowym

- realizacja niektórych etapów może powodować konflikty dostępu do pamięci
- jeśli czasy trwania poszczególnych etapów mogą być niejednakowe, to na różnych etapach wystąpi pewne oczekiwanie
- w programie występują skoki (rozgałęzienia) warunkowe
- wystąpienie przerwania sprzętowego lub wyjątku procesora stanowi zdarzenie nieprzewidywalne i również pogarsza przetwarzanie potokowe

Metody rozwiązywania zastojów

- zmiana kolejności instrukcji
- przenazwanie rejestrów
- rozwijanie pętli
- zgadywanie dalszej kolejności rozkazów
- optymalizacja kodu

Zrównoleglanie obliczeń

- na poziomie instrukcji (potokowanie, VLIW, przetwarzanie superskalarne)
- na poziomie danych
- na poziomie procesów, wątków
  - Hyper Threading
  - wielordzeniowość
  - klastry, superkomputery, grid ...

Przyczyny rozwoju wielordzeniowości

- memory wall – szybszy rozwój procesorów od pamięci
- instruction level parallelism wall (ILP wall) – brak możliwości dalszego zrównoleglania poprzez potokowanie
- power wall – coraz większy pobór mocy wraz ze wzrostem mocy obliczeniowej

Co to jest procesor wielordzeniowy?

- kilka mikroprocesorów zintegrowanych na jednej matrycy
- każdy z mikroprocesorów posiada własną pamięć podręczną
- dostęp do zasobów komputera wspólny dla mikroprocesorów

Przykłady

- AMD Athlon64, Athlon 64 FX, Athlon 64 X2, Sempron X2, Turion X2
- Power 4, Power 5, Power 6
- Intel Core Duo, Core 2 Duo, Core 2 Quad, Core i7, Itanium 2, Pentium D
- Nvidia GeForce

Wielordzeniowość a system operacyjny

- możliwość obsługi większej liczby procesów bez konieczności przełączania kontekstu
- szybsza praca przy wielu procesach
- możliwość zrównoleglenia obliczeń na poziomie instrukcji pomiędzy procesorami

Zastosowania

- silniki graficzne gier wspierające obliczenia na wielordzeniowych procesorach
- biblioteki pozwalające tworzyć programy rozproszone:
  - Clik++
  - OpenMP
  - MPI

64-bitowość

- co to znaczy, że procesor jest 64-bitowy?
- historia procesorów 64-bitowych
- x86-64 jako przykład powszechnej architektury 64-bitowej
- 64-bitowe systemy operacyjne
  - przykłady: Windows, Linux
  - zalety
  - wady i problemy

64-bitowy procesor (w teorii):

- 64-bitowe rejestry ogólnego przeznaczenia
- 64-bitowe szyny (adresów i danych)
- 64-bitowe integery i adresy
- może zaadresować pamięć rozmiaru 2^{64}

64-bitowość: co to znaczy?

64-bitowy procesor (w praktyce):

- szerokość szyn od 32 do 64 bitów
- odpowiednio mniejsza adresowalna przestrzeń

Czasem nazywa się tak nawet procesory 32-bitowe o 64-bitowej szynie danych!

Wpływ architektury procesora na system operacyjny

64-bitowość: historia

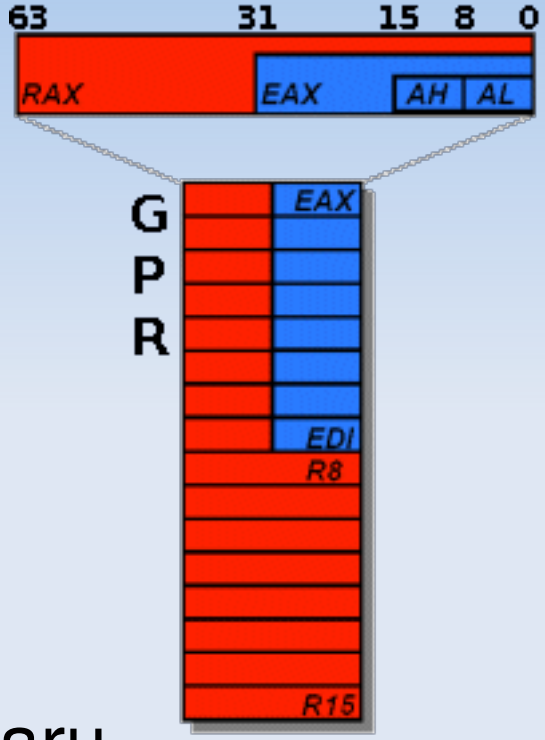
- już w latach 60. 64-bitowe CPU w superkomputerach (tymczasem architektura 32-bitowa jako standard dopiero od lat 80.)
- od początku lat 90. 64-bitowe procesory coraz częściej w serwerach i stacjach roboczych opartych na architekturze RISC
- od 2003 powszechne w komputerach domowych dzięki architekturom PowerPC 64 i x86-64

Architektura x86-64

- zaprojektowana przez AMD
- wprowadzona przez Intela pod nazwą EM64T
- obecnie AMD64 i Intel64
- powszechna we współczesnych pecetach:
  - Intel Core 2
  - AMD Athlon 64, Turion 64, Opteron

64-bitowość: architektura x86-64

Najważniejsze cechy:



- nadzbiór instrukcji i rejestrów 32-bitowej architektury x86
- 8 zupełnie nowych rejestrów
- istniejące implementacje mają:
  - wirtualną przestrzeń adresową rozmiaru aż do 256 TB
  - fizyczną przestrzeń adresową rozmiaru 1 TB (ograniczenie wynika z rozmiaru szyny adresowej)
  - ostatnią wartość można zwiększyć do 4 PB (ograniczenie wynika z algorytmu stronicowania)

64-bitowość: architektura x86-64

Tryby pracy

Tryb		System	Konieczność przekompilowania aplikacji	Domyślny rozmiar adresu	Domyślny rozmiar operandów[2]	Rozmiar rejestrów ogólnych	Rozszerzone rejestry[1]
Long	***64-bit***	64-bitowy	Tak	64	32	64	Tak
	Compatibility		Nie	16 lub 32	16 lub 32	32	Nie
Legacy		16 lub 32-bitowy	Nie	16 lub 32	16 lub 32	16 lub 32	Nie

[1] dostęp do 8 dodatkowych rejestrów ogólnego przeznaczenia (R8 - R15)
[2] rozmiar można oczywiście modyfikować przez ustawienie odpowiedniego bitu bądź używanie prefiksów instrukcji

Windows x64: najważniejsze cechy:

- 8 TB przestrzeni adresowej dla każdego procesu użytkownika
- 8 TB przestrzeni adresowej dla jądra
- można zaadresować od 128 GB do 1 TB RAMu (w zależności od wersji)
- model danych LLP64:
  - 32-bitowe integery i longi
  - 64-bitowe wskaźniki

Windows x64: uruchamianie programów:

- wszystkie procesy trybu jądra muszą być 64-bitowe (nie można instalować 32-bitowych sterowników)
- można uruchamiać 32-bitowe procesy działające w trybie użytkownika
  - jeśli są skompilowane z opcją *'large address aware'* będą mogły korzystać z 4 GB przestrzeni adresowej zamiast 2 GB domyślnych dla Windows

Linux: najważniejsze cechy:

- wsparcie dla procesorów 64-bitowych od wersji 2.4 jądra
- fizyczna przestrzeń adresowa aż do 64 TB
- wirtualna przestrzeń adresowa aż do 128 TB
- model danych LP64:
  - 32-bitowe integery
  - 64-bitowe longi
  - 64-bitowe wskaźniki

64-bitowość: Zalety

Zalety 64-bitowego SO czyli...

To nieprawda, że 64-bitowy system jest przydatny tylko wtedy, gdy ma się więcej niż 4 GB RAMu!

Wirtualna przestrzeń adresowa

- część przestrzeni adresowej procesu jest rezerwowana na biblioteki i komponenty systemu operacyjnego – dlatego też w 32-bitowym systemie procesy mają mniej niż 4 GB ”dla siebie”
  - w Windowsach domyślnie proces ma do dyspozycji 2 GB, w Linuxach około 3 GB
  - uniemożliwia to na przykład mapowanie dużych plików na pamięć

Fizyczna przestrzeń adresowa

- w 32–bitowym systemie można zaadresować co najwyżej 4 GB pamięci fizycznej
- jednak część adresów potrzebna jest urządzeniom wejścia/wyjścia (np. PCI, PCI Express)
- pozostałych adresów może więc być za mało, aby zaadresować 4, a nawet 3 GB pamięci fizycznej – pozostanie ona niewykorzystana

Tych ograniczeń nie ma w 64-bitowych systemach operacyjnych!

Inne zalety

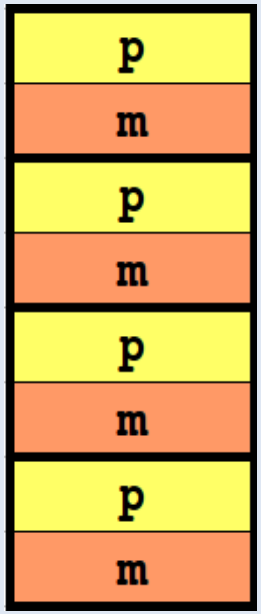
- szybsze w niektórych zadaniach obliczeniowych
  - silnie dużych liczb – do dwóch razy szybciej
  - szyfrowanie danych – nawet kilkukrotnie szybciej
- wydajniejsze w manipulacji danymi dużego rozmiaru
    - ogromne bazy danych
    - obliczenia naukowe
    - obróbka wideo
- sprawniejsze jeśli chodzi o wielozadaniowość i pracę przy dużym obciążeniu

Wady i problemy

- puchnięcie danych i programów
  - 32-bit: `sizeof(struct task_struct) ~ 2000`
  - 64-bit: `sizeof(struct task_struct) ~ 4000`
  - uwaga na wyrównanie danych

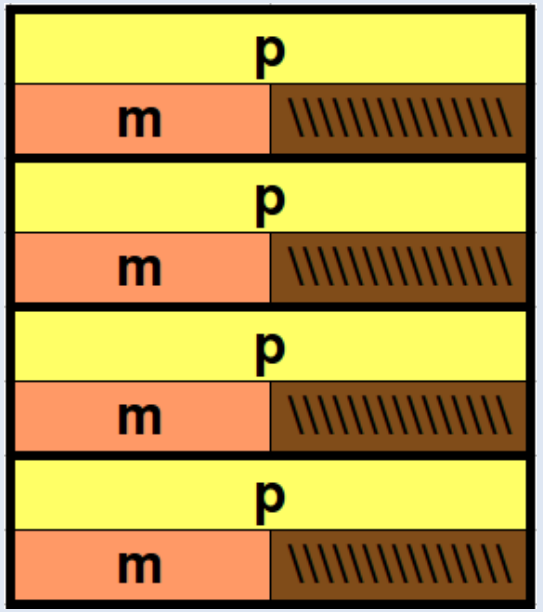
```
struct str {
   void* p;
   int m;
};

struct str tab[4];
```



32-bit

`sizeof(tab) = 32`



64-bit

`sizeof(tab) = 64`

Wady i problemy

- brak niektórych programów i sterowników w wersjach 64-bitowych
  - np. Adobe Flash Player
- czasem problemy ze sprzętem
- czasem aplikacje 32-bitowe mogą działać wolniej
  - x86-64 – różnica prawie niezauważalna (po prostu nie używa się niektórych rejestrów i instrukcji)
  - Itanium i inne architektury w pełni 64-bitowe – różnica może być znacząca (potrzeba emulować środowisko 32-bitowe)

Problemy przy przenoszeniu aplikacji

- używanie ”magicznych liczb”
  - alokowanie pamięci

```
size_t arraySize = N * 4;
long *array = (long *) malloc(arraySize);
```

  - arytmetyka

```
long l = 0 ^ 0x80000000;
printf("%ld\n", l);
```

32-bit	64-bit (LP64)
-2147483648=LONG_MIN	2147483648 = ?

Problemy przy przenoszeniu aplikacji

- indeksowanie tablic

```
int i = 0;
while (ogromnaTablica[i] != id)
  i++;
```

  - to nie zadziała dla tablic większych od INT_MAX
- operacje na bitach

```
ptrdiff_t setBit(ptrdiff_t val, unsigned bitNum){
  ptrdiff_t mask = 1 << bitNum;
  return val | mask;
}
```

  - `setBit(0,32) = 0x0 != 0x100000000`
  - nie działa, bo ”1” jest typu int i przy shifcie następuje przepełnienie

Problemy przy przenoszeniu aplikacji

- projektując aplikację na 64-bitowy system operacyjny należy pamiętać o tym, że być może użytkownik będzie chciał korzystać z bardzo dużych zestawów danych (prawdopodobnie dlatego właśnie zainstalował taki system)
  - tablice o ponad INT_MAX elementach
  - pliki większe niż 4GB
- należy to uwzględnić także w czasie testów, gdyż niektóre błędy mogą się ujawnić dopiero przy danych bardzo dużych rozmiarów
    - patrz przykłady na poprzednich slajdach
- należy pamiętać też o tym, że być może przyjdzie czas architektur 128-bitowych!

Źródła

- Wikipedia
  http://en.wikipedia.org/ http://pl.wikipedia.org/
- RISC Architecture
  http://cse.stanford.edu/class/sophomore-college/projects-00/risc/
- Zarządzanie pamięcią w 32- i 64-bitowych systemach Windows http://www.microsoft.com/poland/technet/article/art0092_01.mspx
- AMD, x86-64TM Technology White Paper
  www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/x86-64_wp.pdf
- AMD 64 FAQ
  http://www.amd.com/us-en/Processors/ProductInformation/0,,30_118_9331_13278,00.html
- 20 issues of porting C++ code on the 64-bit platform
  http://www.codeproject.com/KB/architecture/20ISSUES64BIT.aspx