Przetwarzanie potokowe

Albert Kaluga

Przetwarzanie potokowe

- Procesor bezpotokowy
- Przejście na przetwarzanie potokowe
  - Problemy implementacji
    - Hazardy
    - Skoki
    - Obsługa przerwań
- Obsługa przerwań
- Wielocyklowe operacje

MIPS - Przypomnienie

- Instrukcje mają stałą długość
  - OPT - Kod operacji 6 najstarszych bitów
  - RS, RT, RD - Numer rejestru (5 bitów każdy)
  - Shift - Przesunięcie (5 bitów)
  - Function - Dospecyfikowanie operacji (5 bitów)
  - IMM - 16 bitowa stała
- Instrukcje artretyczne (ALU)
  - Biorą RD i RS, wykonują obliczenia, wynik w RT
  - Biorą RS i IMM , wynik w RT

MIPS - Przypomnienie

- Instrukcje dostępu do pamięci
  - Biorą jeden rejestr(RS) i IMM do obliczenia efektywnego adresu
  - RT jest zapisywany w pamięci lub na odwrót.
- Instrukcje skoków
  - Służą do zmiany sterowania (PC)
  - Cechą wspólną jest stała będąca miejscem skoku
  - Porównywanie dwóch rejestrów
  - Porównywanie rejestru z zerem

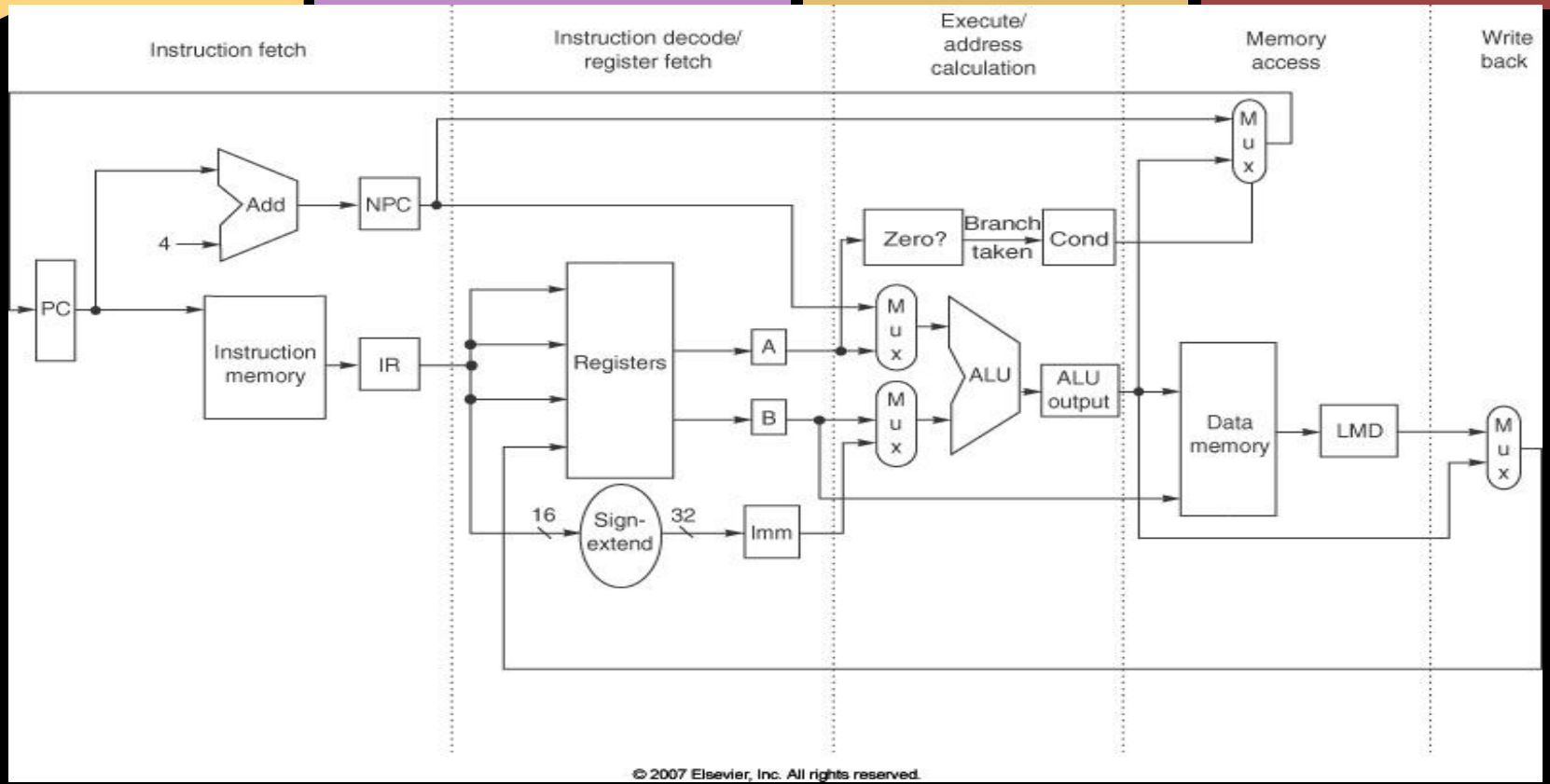
Procesor bezpotokowy

- CPI - Ilość cykli potrzebna na wykonanie instrukcji
  - Artmetyczne = Skoki =4, Dostęp do pamięci 5
- Czas wykonania instrukcji
  - T= Czas jednego cyklu x CPI
- Intuicja -> Linia montażowa gdzie w każdej fazie przesuwamy produkt na linii

Procesor bezpotokowy

- Fazy przetwarzania instrukcji
  - IF - Pobranie instrukcji z pamięci
  - ID - Dekodowanie instrukcji
  - EX - Obliczenia(MR,RR,RI)
  - MA - Dostęp do pamięci(Load,Store)
  - WB - Zapis do rejestrów

Procesor bezpotokowy



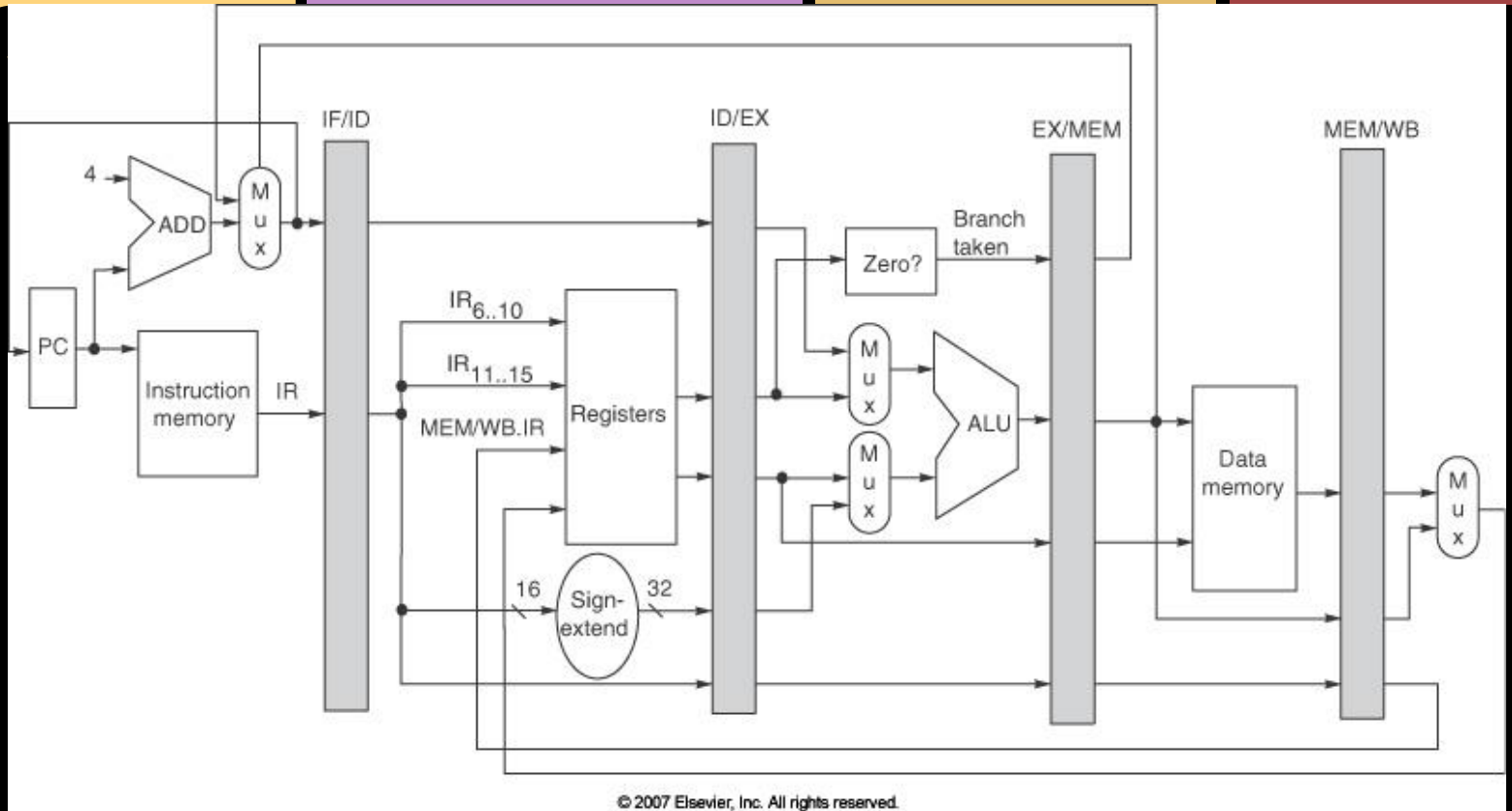
Procesor bezpotokowy

- Linia montażowa gdzie mamy dużo pracowników ale składamy tylko jeden samochód
- Czemu nie składać kilku samochodów na raz?

Procesor potokowy - poczatek

- Potok - wykonywanie kilku instrukcji równocześnie
- Każda instrukcja jest w innej fazie wykonania
- Co cykl jedna instrukcja się kończy (przypadek optymalny)
- CPI = 1

Procesor potokowy - początek



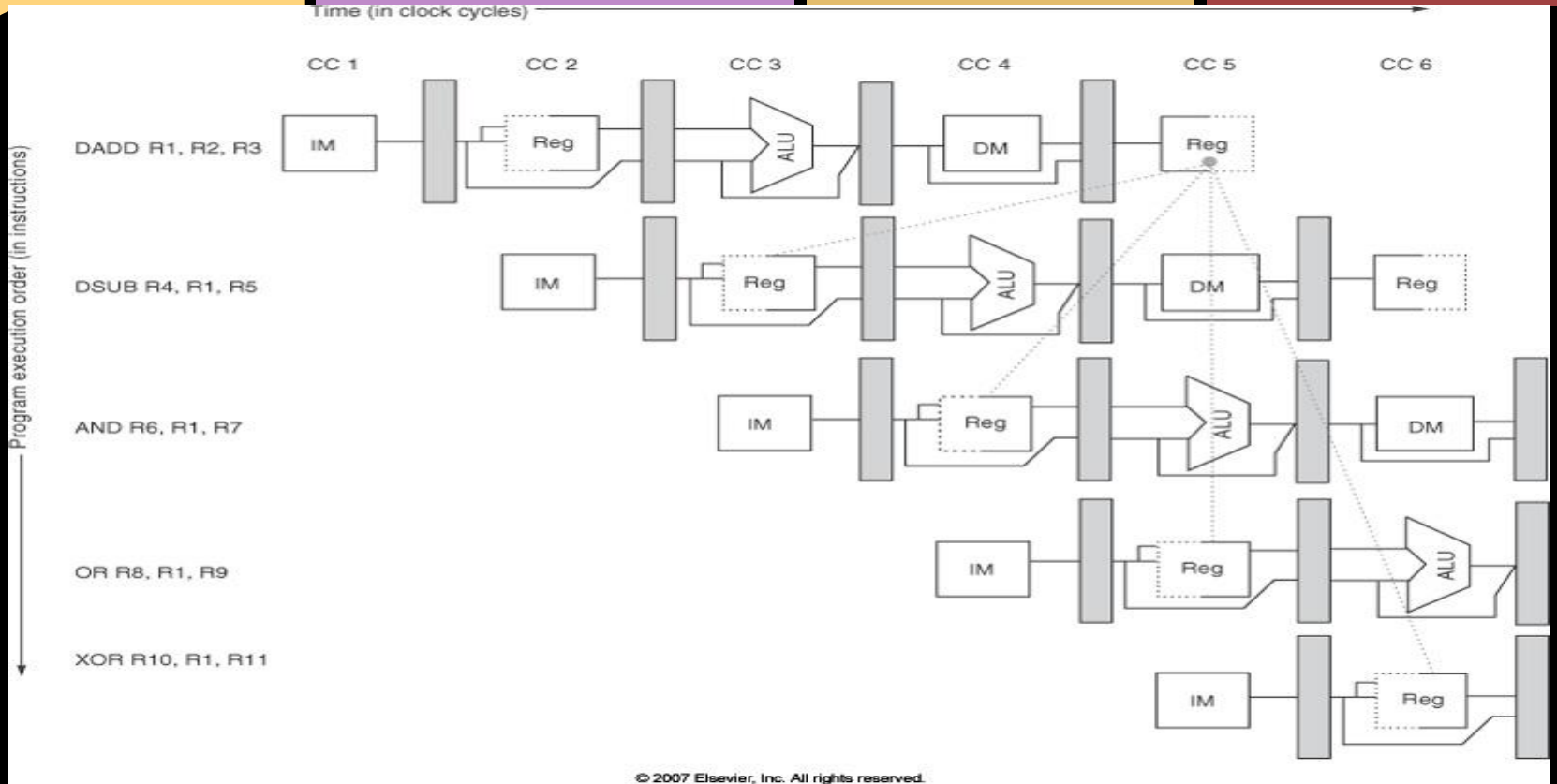
Procesor potokowy - początek

- Dodatkowe rejestry przechowujące wartości po każdej fazie
- IF/ID.NPC - Oznacza NPC po wyjściu z fazy IF przed fazą ID.
- Problem - Hazard danych

Problem - Hazard danych

1. LD R1,0(R2)
2. DSUB R4,R1,R5
3. AND R6,R1,R7
4. OR R8,R1,R9

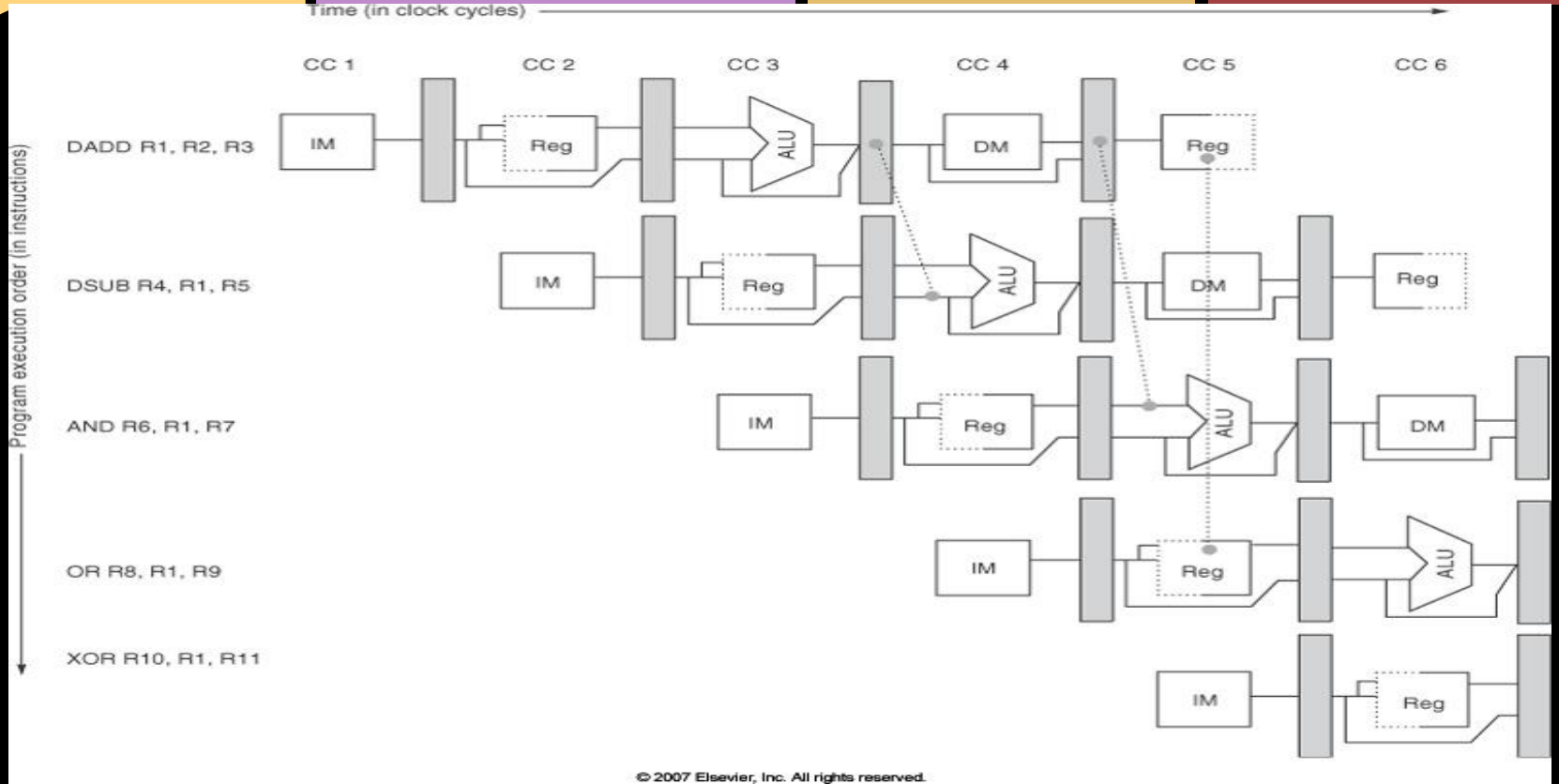
Problem - Hazard danych



Problem - Hazard danych

- Wstrzymywanie potoku
  - Prosto rozwiązuje problem hazardu danych. Wystarczy wykrywać
  - Zmniejsza wydajność o 1/(1+n) gdzie n to ilość wstrzymanych cykli.
- Forwarding
  - Dodatkowe połączenia pozwalają kontynuować potok
  - Nie wszystkie hazardy da się rozwiązać

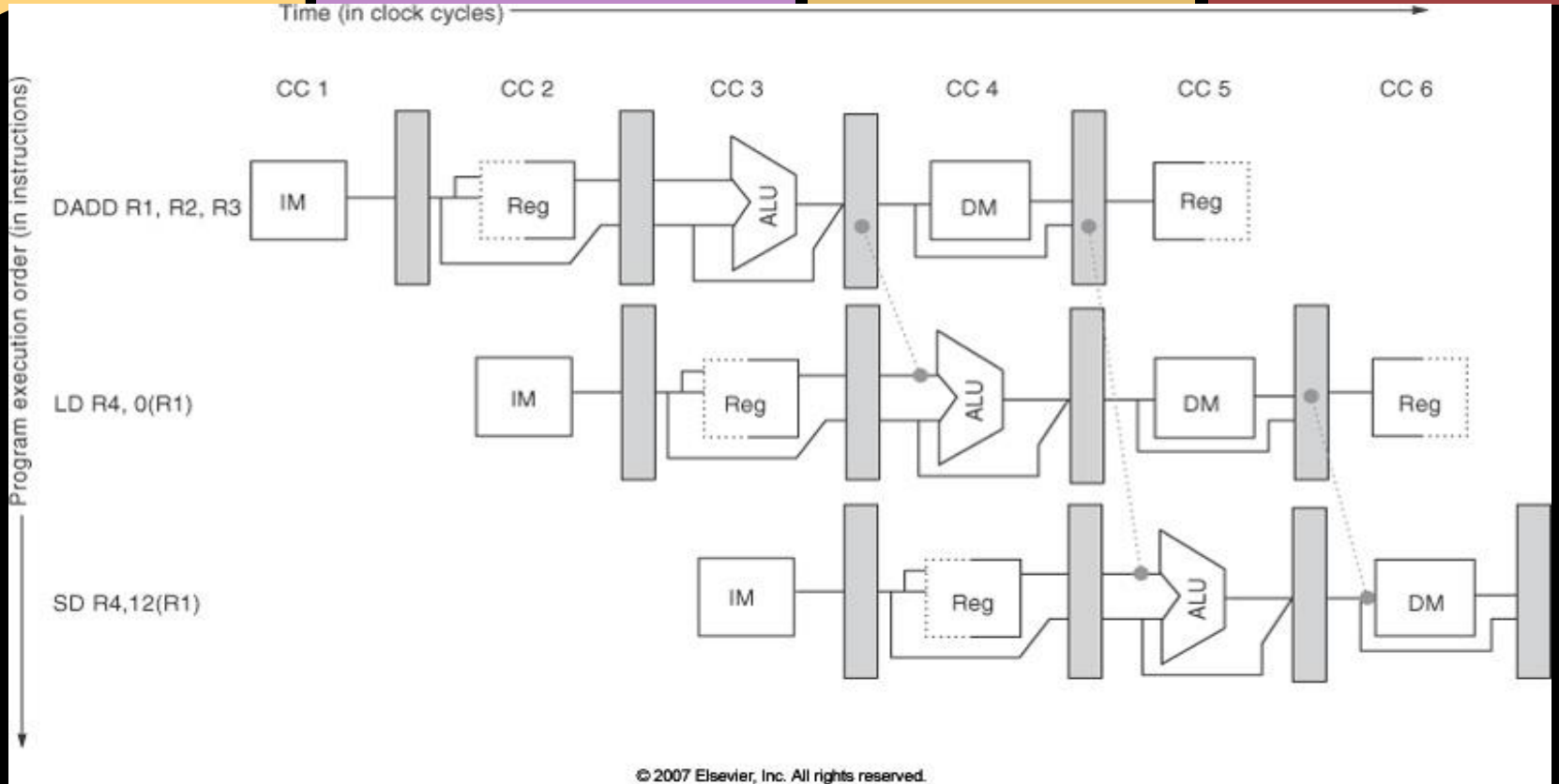
Problem - Hazard danych



Problem - Hazard danych

- Kolejne instrukcje pobierają R1 wartości z rejestrów tymczasowych
  - SUB pobiera wartość z EX/MA
  - AND pobiera wartość z MA/WB

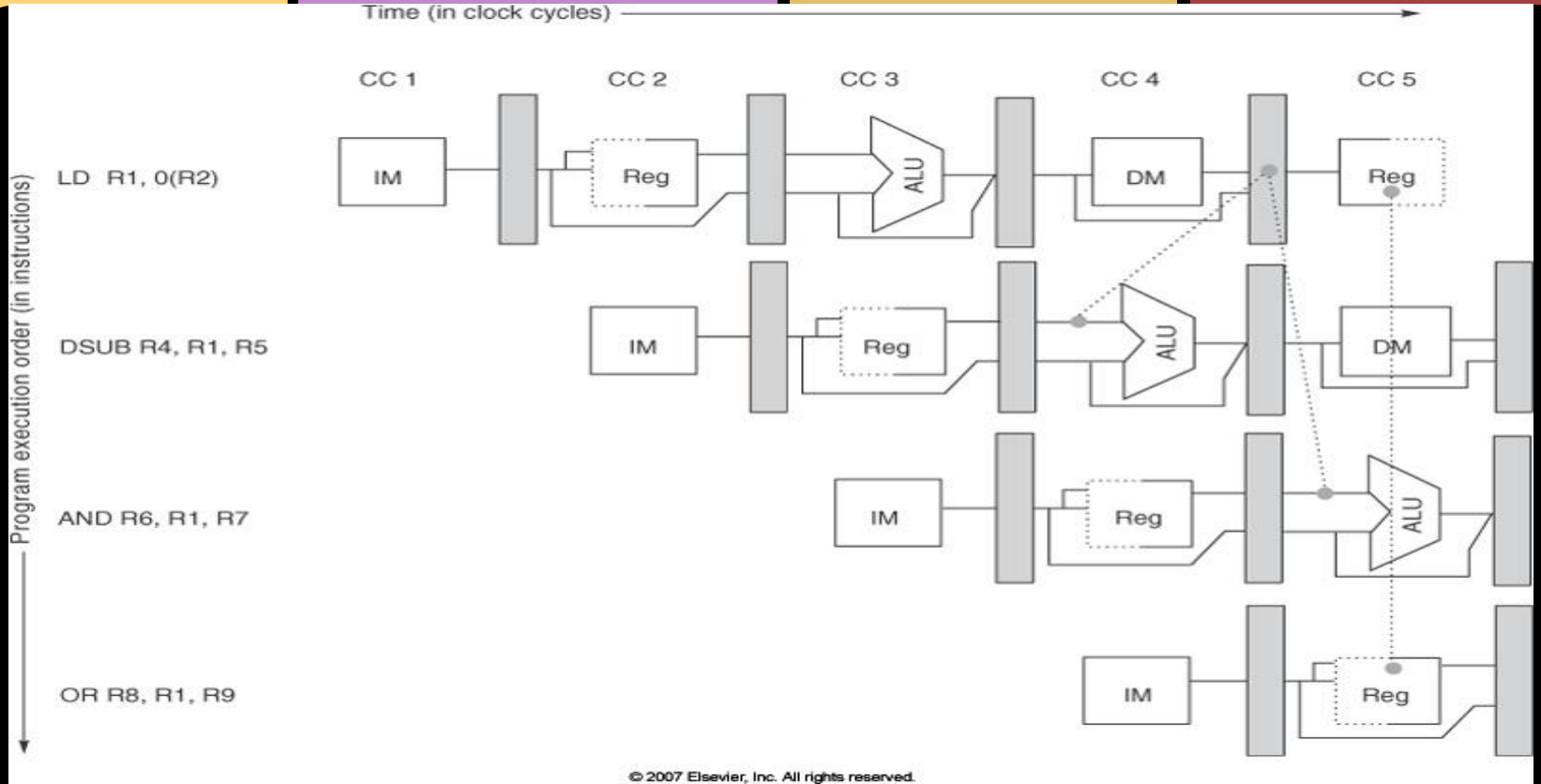
Problem - Hazard danych



Problem - Hazard danych

- Analogicznie jak w poprzednim przypadku
  - LD pobiera R1 z EX/MA
  - SD pobiera R4 z MA/WB

Problem - Hazard danych



Problem - Hazard danych

- SUB pobiera R1 z MA/WB
- AND też pobiera R1 z MA/WB
- OR ma już R1 w rejestrze

Problem - Hazard danych

SUB nie może pobrać wartości R1 z podanych rejestrów dopóki inżynierowie nie opracują technologii przenoszenia w czasie. Póki co potok musi zostać zatrzymany, do czasu zakończenia fazy MA dla LD.

Problem - Hazard danych

1. LD R1, 0(R3)
2. DADD R2, R2, R1
3. LD R4, 4(R3)
4. DADD R2, R2, R4
5. LD R1, 8(R3)
6. DADD R2, R2, R1
7. LD R4, 12(R3)
8. DADD R2, R2, R4

Zoptymalizuj!

EX/MEM.IR[rd] == ID/EX.IR[rs]

MEM/WB.IR[rt] == ID/EX.IR[rs]

Podaj przykład instrukcji które wymusi dany forwarding!

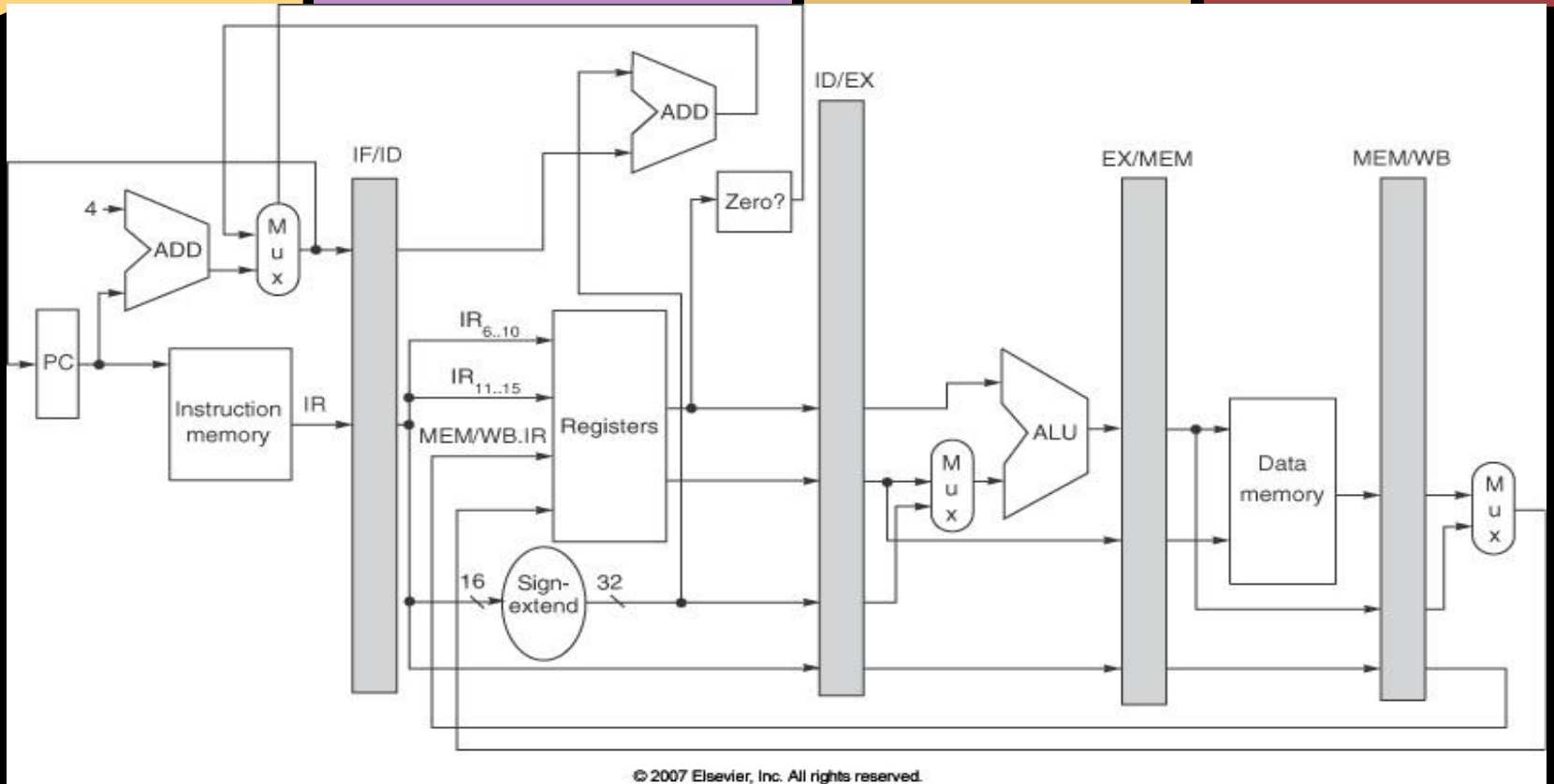
Problem - Hazard sterowania

- PC jest zwiększane o 4 w fazie IF
- Może się okazać może okazać się, że trzeba przenieść sterowanie gdzie indziej niż PC+4
- Problem - W potoku są już inne instrukcje

Problem - Hazard sterowania

- Im później wykryjemy/wykonamy skok tym więcej instrukcji trzeba wyzerować = mniejsza wydajność
- Pytanie : Kiedy najwcześniej można wykonać skok? Czego potrzeba by to zrobić?

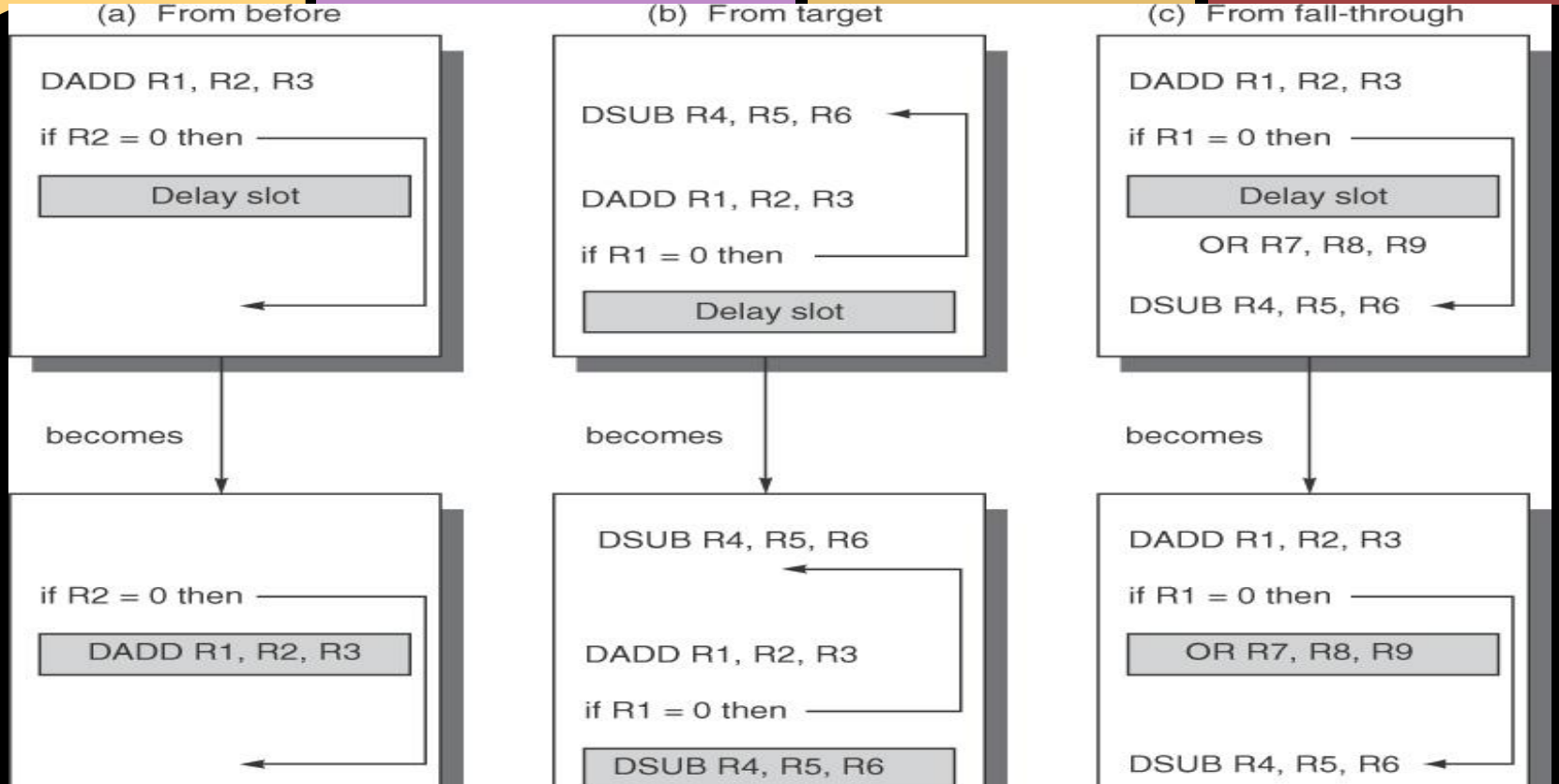
Problem - Hazard sterowania



Problem - Hazard sterowania

- Wciąż jedna instrukcja wczytuje się “niepotrzebnie”
- Różne techniki rozwiązania tego problemu
  - Skok zawsze się wykona
  - Skok nigdy się nie wykona
  - Opóźnianie skoku - wstawianie niezależnej od skoku instrukcji po skoku

Problem - Hazard sterowania



Obsługa przerwań

- Przerwanie - zdarzenie wyjątkowe
- Prawie nie do przewidzenia
- Zmniejszenie wydajności procesora
- “Przeźroczystość” obsługi

Obsługa przerwań - Typy przerwań

- Występujące w IF
  - Page fault
  - Misaligned memory access
- Występujące w ID
  - Undefined or illegal opcode
- Występujące w EX
  - Arithmetic exception
- Wytępujące w MA
  - Jak w IF ;-)

Obsługa przerwań - Typy przerwań

- Przerwania niezależne od fazy
  - I/O device request
  - Hardware malfunction
  - Power failure

Obsługa przerwań - Typy przerwań

Synchroniczne - Jeśli przerwanie jest powtarzalne w tych samych warunkach

Asynchroniczne - Jeśli jest niezależne od tego co robi procesor

Page fault? I/O device request?

Obsługa przerwań - Typy przerwań

User requested - Program sam wywołuje przerwanie

Coerced - Wymuszone przez sprzęt. Program nie ma na nie wpływu

Breakpoint? Integer arithmetic overflow?

Obsługa przerwań - Typy przerwań

Maskowalne - Mniej ważne przerwania które w razie potrzeby można wyłączyć.

Niemaskowalne - Ważne przerwania. Muszą zostać obsłużone.

Misaligned memory accesses? Page fault?

Obsługa przerwań - Typy przerwań

W instrukcji - przerwanie jest powodowane przez instrukcje. Najczęściej synchroniczne. Trudniejsze do implementacji.

Pomiędzy - przerwanie niezależne aktualnie wykonywanej instrukcji.

Czy istnieje asynchroniczne przerwanie w instrukcji?

Obsługa przerwań - Typy przerwań

Terminalne - Kończą działanie programu. Nie trzeba przywracać zawartości rejestrów. Łatwiejsze do implementacji.

Wznawialne - Po obsłużeniu przerwania trzeba przywrócić stan procesora do stanu przed przerwaniem.

Power failure? Undefined instructions?

Obsługa przerwań - Schemat

1. Wymuszenie przejścia w tryb nadzorcy
2. Zablokowanie zapisów z rejestrów tymczasowych do pamięci i rejestrów.
3. Zapisanie PC aktualnej instrukcji. Będzie użyte do powrotu po obsłudze.

Obsługa przerwań - Przeźroczystość

Precise exceptions - obsługa przerwań która pozwala zakończyć “dobre” instrukcje które wystąpiły przed przerwaniem. A po zakończeniu obsługi wykonać dalsze instrukcje od podstaw.

Tego typu obsługa może być 10 razy wolniejsza!

Obsługa przerwań - Przeźroczystość

Jeśli stosujemy opóźnianie skoków, nie ma możliwości odtworzenia tego stanu bez dodatkowych rejestrów

Jeśli PC wskazuje na skok, przed restartem sprawdza się warunek skoku i wczytuje właściwą instrukcję

Operacje wielocyklowe

- Operacje zmiennopozycyjne są znacznie bardziej skomplikowane
- Potrzeba znacznie więcej czasu lub znacznie więcej logiki a to kosztuje
- Rozwiązanie - Operacje wielocyklowe

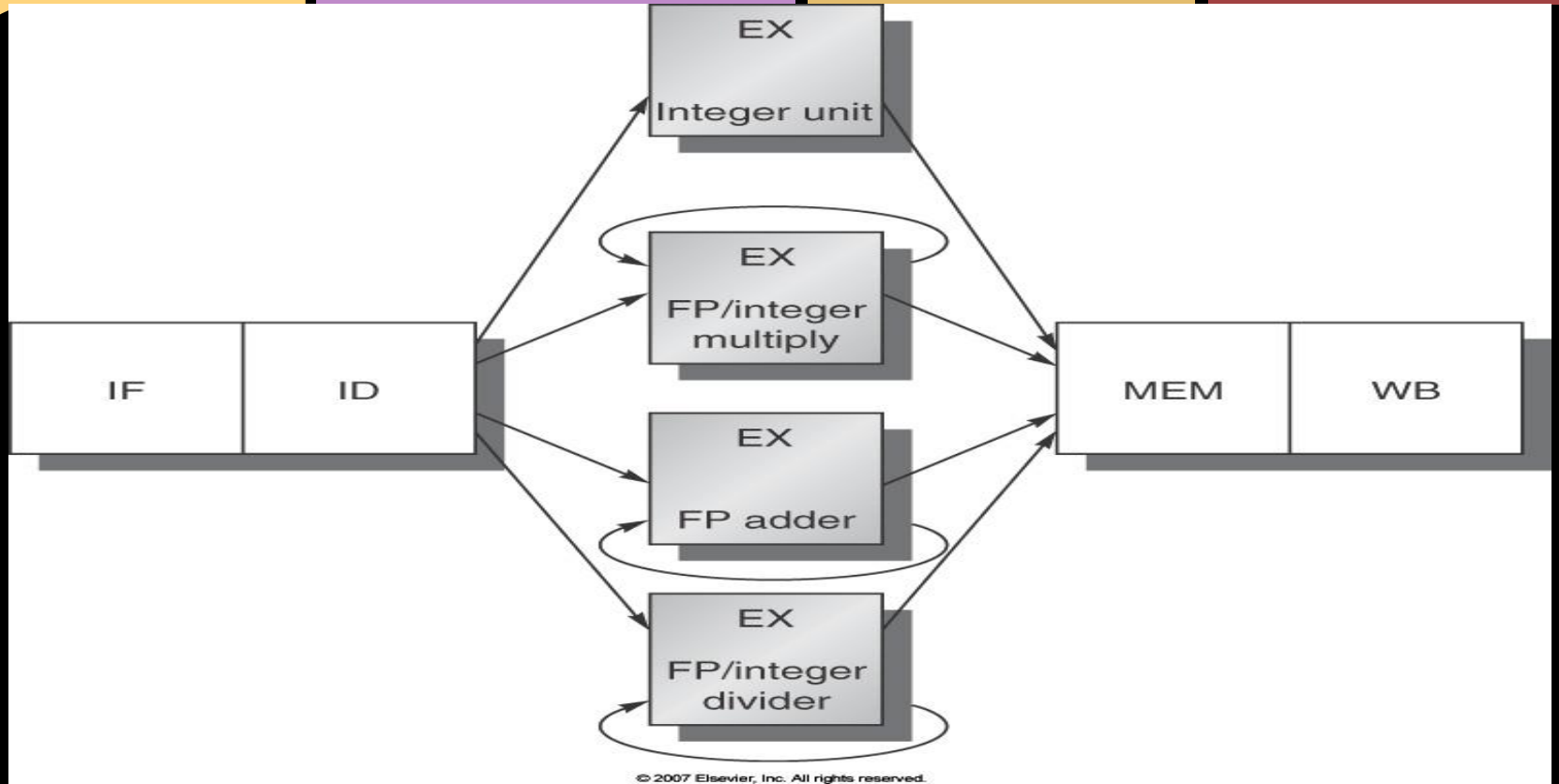
Operacje wielocyklowe- motywacja

- Potok stałopozycyjny działa dobrze, nie psujemy czegoś co działa
- Dodajemy dodatkowe jednostki obliczeniowe poza potokiem
- Pozwólmy żeby instrukcje mogły wykonywać się kilka cykli

Operacje wielocyklowe

- Jednostki EX:
  - ALU - pozostawiamy bez zmian
  - Multipler FP- do mnożenia liczb zmiennopozycyjnych
  - Adder FP- do dodawania i odejmowania liczb zmiennopozycyjnych
  - Divider FP - do dzielenia liczby zmiennopozycyjnych przez stałopozycyjną

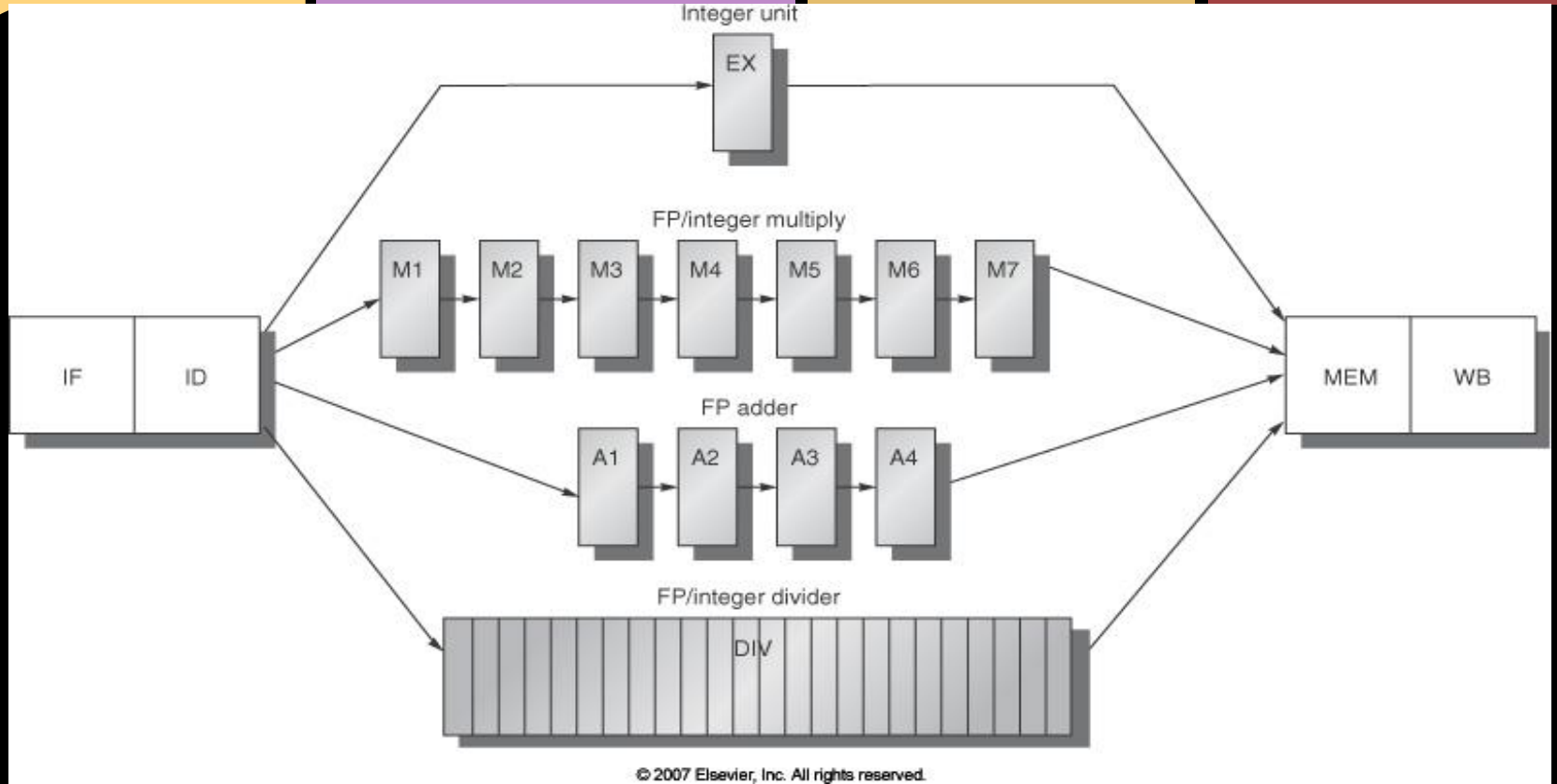
Operacje wielocyklowe



Operacje wielocyklowe

- Każda jednostka potrzebuje innej ilości iteracji do zakończenia działania operacji.
- Nie można zacząć nowej instrukcji wykorzystującej jednostkę dopóki poprzednia instrukcja z niej nie wyjdzie (Hazard Strukturalny)

Operacje wielocyklowe



Operacje wielocyklowe

- Blokada -Dwie jednostki moga skończyć obliczenia w tym samym czasie a jest jeden slot na zapis (Hazard strukturalny)
- Bardziej skomplikowana jednostka przewidująca hazardy danych
- Problem z odtwarzaniem stanu przy wyjątkach

Operacje wielocyklowe - Blokada

- Zwiększyć ilość slotów zapisów
  - Może okazać się kosztowne i zbędne
- Wykryć w fazie ID
  - Obliczamy kiedy instrukcja będzie potrzebowała zapisu do rejestrów
  - Jeśli inna instrukcja już zarezerwowała w tym czasie zapis opóźniamy wykonanie

Operacje wielocyklowe - Blokada

- Poczekać aż zdaży się konflikt
  - Mniej dodatkowych rejestrów i logiki
  - Trzeba określić politykę którą instrukcję wpuścić jaka pierwszą a którą zatrzymać

Operacje wielocyklowe - wyjątki

- W czasie zgłoszenia wyjątku może się okazać, że któraś krótsza instrukcja zniszczyła swoje zmienne
  - DIV.D F1, F2, F4
  - ADD.D F10, F10, F8
  - SUB.D F12, F12, F14
- Nie można wtedy odtworzyć stanu procesora

Operacje wielocyklowe - wyjątki

- Można nic z tym nie robić
  - Mniej problemów implementacji
  - Spadek kosztów
- Niestety jest to niedopuszczalne

Operacje wielocyklowe - wyjątki

- Zabraniać więcej niż 1 instrukcji używać któregokolwiek koprocesora
- Pamiętać wyniki operacji dopóki najdłuższa operacja się nie skończy
- Pamiętać ciąg operacji które nie są jeszcze skończone