Architektura systemów komputerowych

Mariusz Wiśniewski

Politechnika Świętokrzyska w Kielcach
Katedra Informatyki

Przetwarzanie potokowe II

Plan wykładu

1. Zjawisko hazardu.
2. Projekt μP – kontroler w potoku.
3. Potok w rodzinie x86.

Cele

Poznanie pojęcia pracy potokowej. Poznanie techniki projektowania procesora z przetwarzaniem potokowym. Omówienie niekorzystnych zjawisk związanych z zastosowanie potoku w CPU.

Zjawisko hazardu

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

- wstęp
- problem hazardu

Przetwarzanie potokowe zwiększa wydajność procesora, jednak takie rozwiązanie posiada również niekorzystne właściwości.

Pojęcia:
- zjawisko hazardu
- operacje na potoku
- wyjątki i pamięć cache

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

- wstęp
- problem hazardu

Przetwarzanie potokowe zwiększa wydajność procesora, jednak takie rozwiązanie posiada również niekorzystne właściwości.

Pojęcia:
- zjawisko hazardu
- operacje na potoku
- wyjątki i pamięć cache

W odniesieniu do potoku hazard oznacza sytuację, gdy kolejna instrukcja nie może zostać wykonana w ciągu jednego cyklu od instrukcji poprzedniej w potoku.

Wyróżnia się następujące rodzaje hazardów:
- hazard strukturalny,
- hazard danych,
- hazard sterowania.

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

- wstęp
- problem hazardu

Przetwarzanie potokowe zwiększa wydajność procesora, jednak takie rozwiązanie posiada również niekorzystne właściwości.

Pojęcia:
- zjawisko hazardu
- operacje na potoku
- wyjątki i pamięć cache

W odniesieniu do potoku hazard oznacza sytuację, gdy kolejna instrukcja nie może zostać wykonana w ciągu jednego cyklu od instrukcji poprzedniej w potoku.

Wyróżnia się następujące rodzaje hazardów:
- hazard strukturalny,
- hazard danych,
- hazard sterowania.

W normalnym trybie pracy potok przekazuje dane między stopniami w odpowiednich jednostkach czasu. Ze względu na możliwość pojawienia się hazardów kontroler potoku może:
- unieważnić instrukcje – po osiągnięciu ostatniego stopnia potoku wyniki są porzucane,
- zarządzić wstrzymanie pracy wybranych stopni potoku (ang. stall / bubble) – opr. NOP,
- opróżnić potok – operacja FLUSH.

Powyższe czynności mają bezpośredni wpływ na wydajność potoku i są stosowane, gdy nie można uniknąć pojawienia się hazardu.

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

– wstęp

– problem hazardu

Przetwarzanie potokowe zwiększa wydajność procesora, jednak takie rozwiązanie posiada również niekorzystne właściwości.

Pojęcia:
- zjawisko hazardu
- operacje na potoku
- wyjątki i pamięć cache

W odniesieniu do potoku hazard oznacza sytuację, gdy kolejna instrukcja nie może zostać wykonana w ciągu jednego cyklu od instrukcji poprzedniej w potoku.

Wyróżnia się następujące rodzaje hazardów:
- hazard strukturalny,
- hazard danych,
- hazard sterowania.

W trakcie wykonania programu mogą pojawiać się wyjątki, które z punktu widzenia procesora mogą zostać obsłużone tak jak skok bezwarunkowy. Z punktu widzenia potoku takie zdarzenie obsługuje się następująco:
- wyjątek ma charakter instrukcji – kontroler symuluje wykonanie instrukcji dalekiego wywołania podprogramu,
- obsługę wyjątku realizuje się na stopniu write-back,
- instrukcje znajdujące się już w potoku są unieważniane.

W normalnym trybie pracy potok przekazuje dane między stopniami w odpowiednich jednostkach czasu. Ze względu na możliwość pojawienia się hazardów kontroler potoku może:
- unieważnić instrukcje – po osiągnięciu ostatniego stopnia potoku wyniki są porzucane,
- zarządzić wstrzymanie pracy wybranych stopni potoku (ang. stall/bubble) – opr. NOP,
- opróżnić potok – operacja FLUSH.

W CPU wyposażonych w pamięć cache zachodzi potrzeba uzupełniania takiej pamięci (ang. cache miss handling). Odnośnie potoku taka czynność wymaga wstrzymania jego pracy, co można osiągnąć dwojako:
- poprzez wprowadzenie globalnego sygnału blokowania – należy zadbać, aby blokowanie odbyło się jednocześnie dla wszystkich stopni potoku,
- poprzez wygenerowanie wyjątku.

Powyższe czynności mają bezpośredni wpływ na wydajność potoku i są stosowane, gdy nie można uniknąć pojawienia się hazardu.

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

- wstęp
- problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

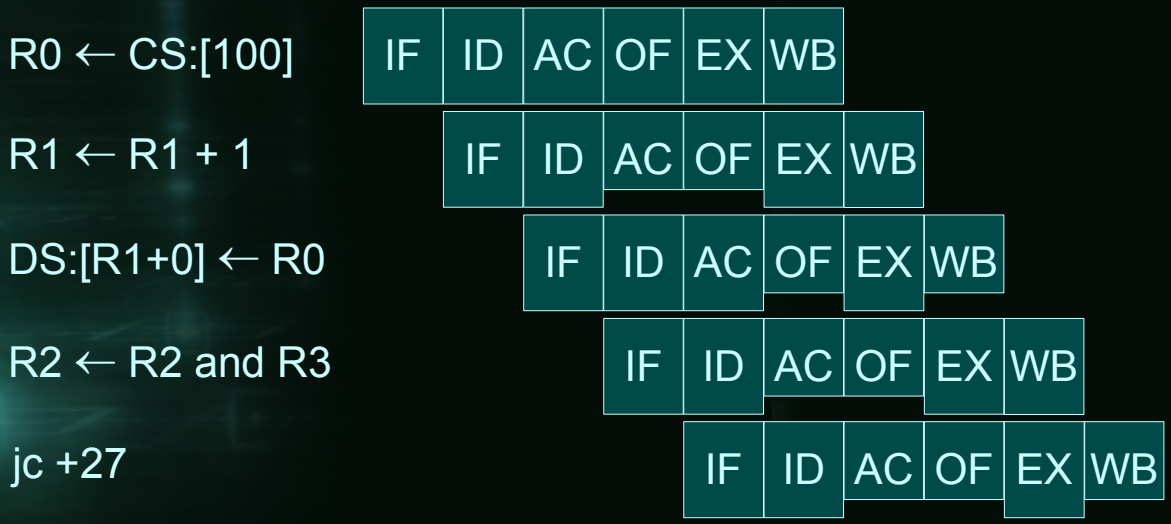
– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard strukturalny występuje podczas próby wykorzystania tych samych zasobów przez stopnie potoku.



Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

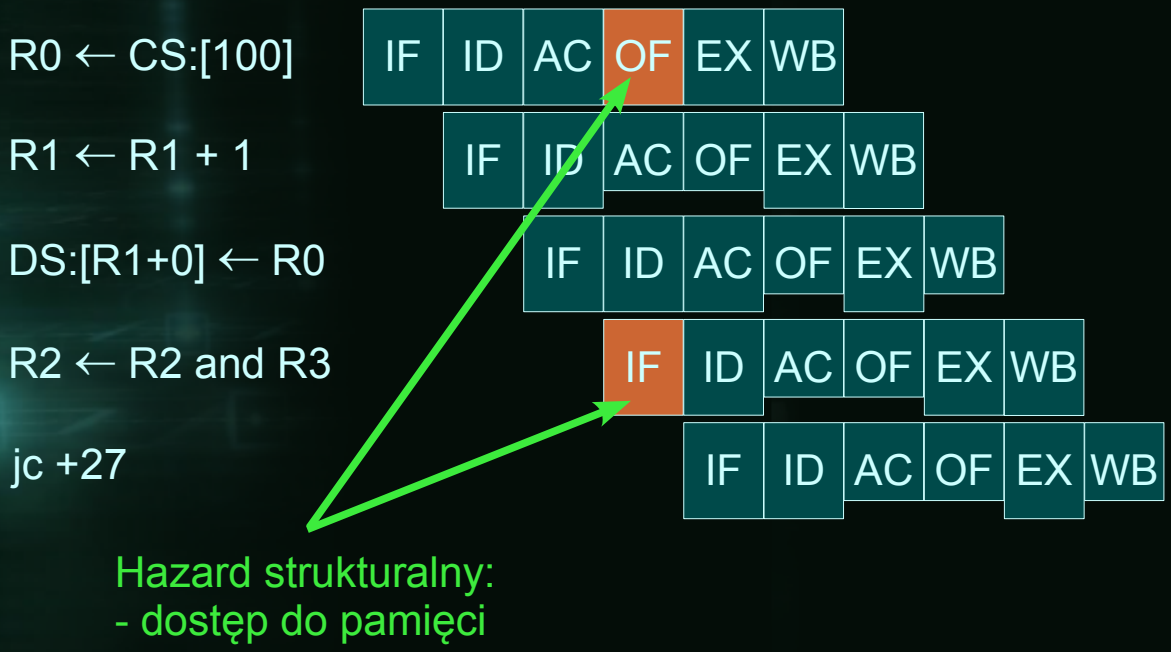
– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard strukturalny występuje podczas próby wykorzystania tych samych zasobów przez stopnie potoku.



Zjawisko hazardu

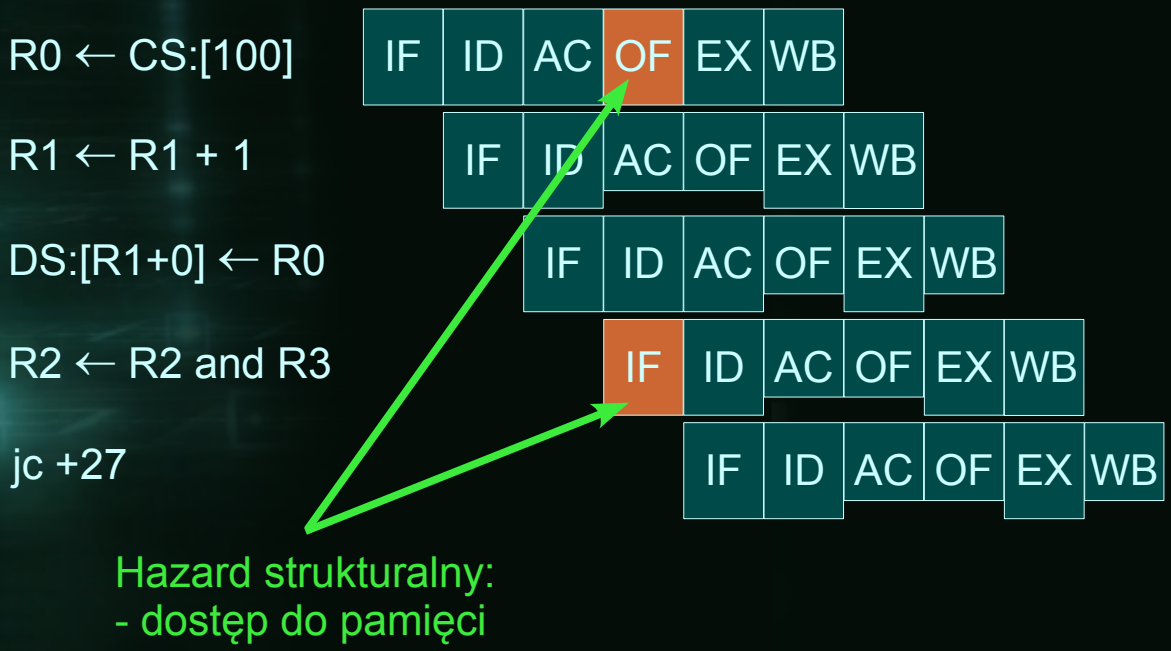
omówienie problemów związanych z pracą potokową w CPU

- wstęp
- problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard strukturalny występuje podczas próby wykorzystania tych samych zasobów przez stopnie potoku.



Rozwiązania problemu:
- powielenie zasobów,
- zmiana kolejności wykonania instrukcji,
- wprowadzenie opóźnień (stall / bubble).

Zjawisko hazardu

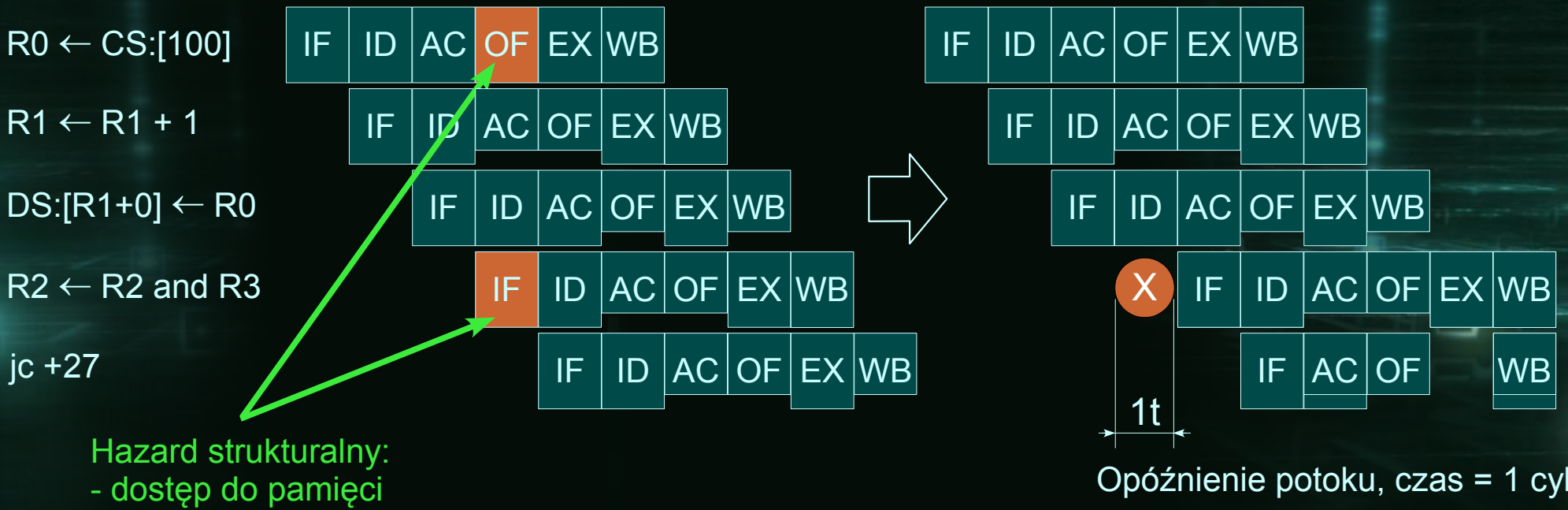
omówienie problemów związanych z pracą potokową w CPU

- wstęp
- problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard strukturalny występuje podczas próby wykorzystania tych samych zasobów przez stopnie potoku.



Zjawisko hazardu

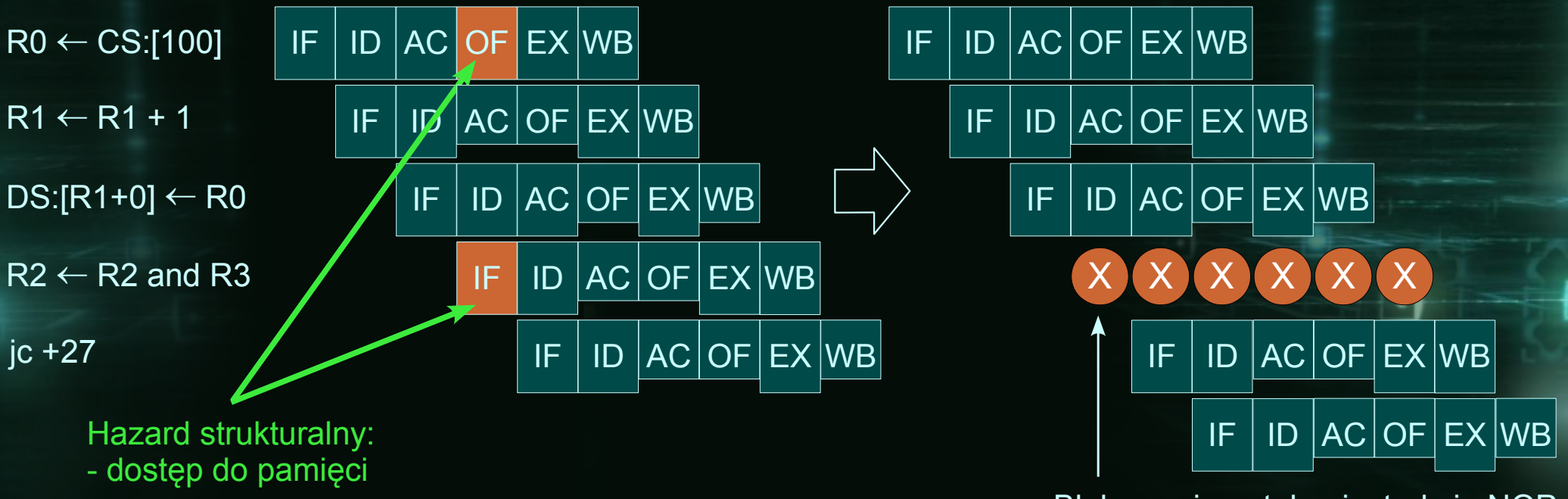
omówienie problemów związanych z pracą potokową w CPU

- wstęp
- problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard strukturalny występuje podczas próby wykorzystania tych samych zasobów przez stopnie potoku.



Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

- wstęp
- problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard danych występuje przy dostępnie do pamięci lub rejestrów. Hazard występuje w odmianach:
- RAW – read after write – przy odczycie danych zanim zapis zostanie zakończony,
- WAR – write after read – zapis wykonano zanim nastąpił odczyt,
- WAW – write after write – zapisy danych następują w niewłaściwej kolejności.

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

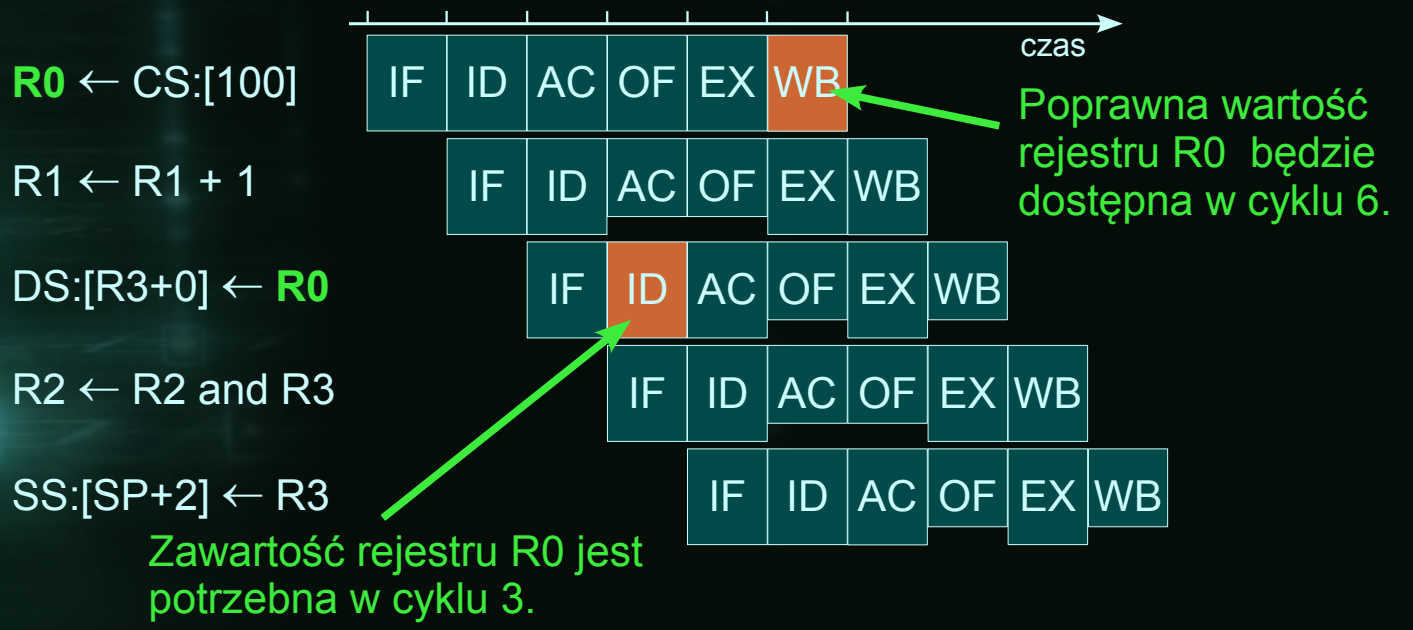
– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard danych występuje przy dostępnie do pamięci lub rejestrów.
Hazard występuje w odmianach: RAW, WAR i WAW.



Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

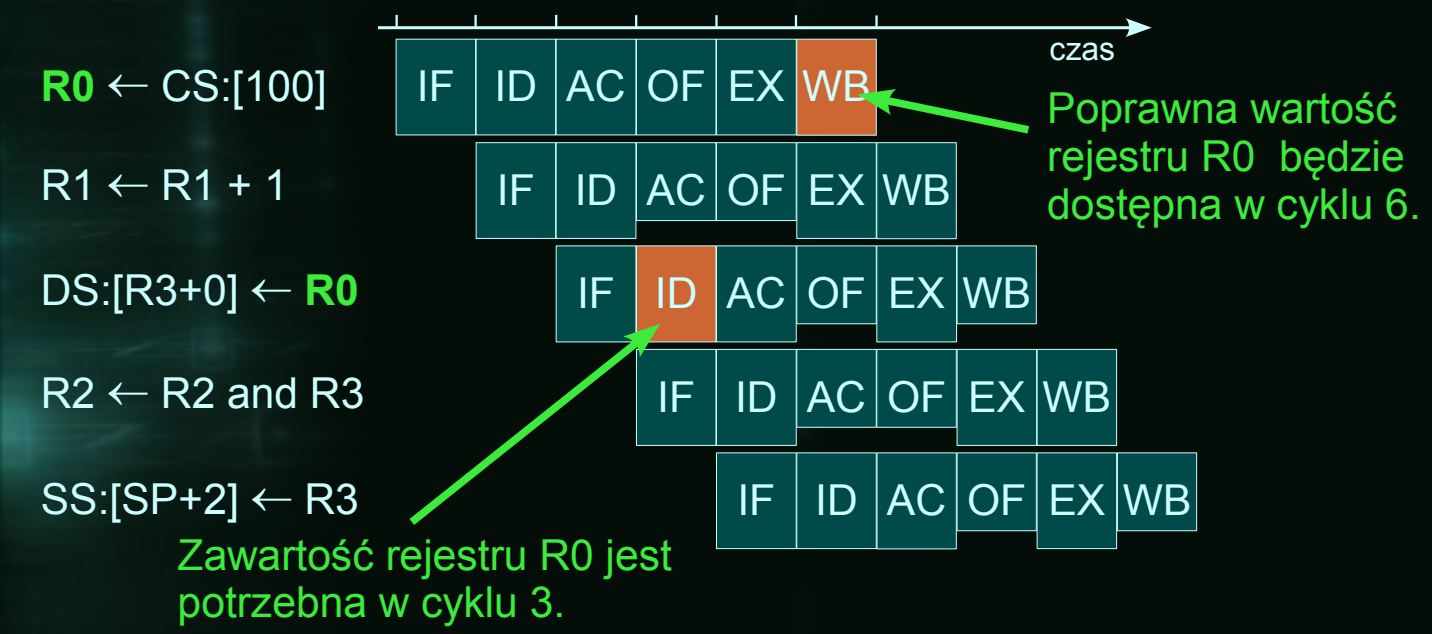
– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard danych występuje przy dostępnie do pamięci lub rejestrów.
Hazard występuje w odmianach: RAW, WAR i WAW.



Rozwiązania problemu:
- przekierowania – np. połączenie wyjścia ALU i wejścia układu adresowego,
- zmiana kolejności wykonania instrukcji,
- wprowadzenie opóźnień (stall / bubble),
- każde z powyższych.

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

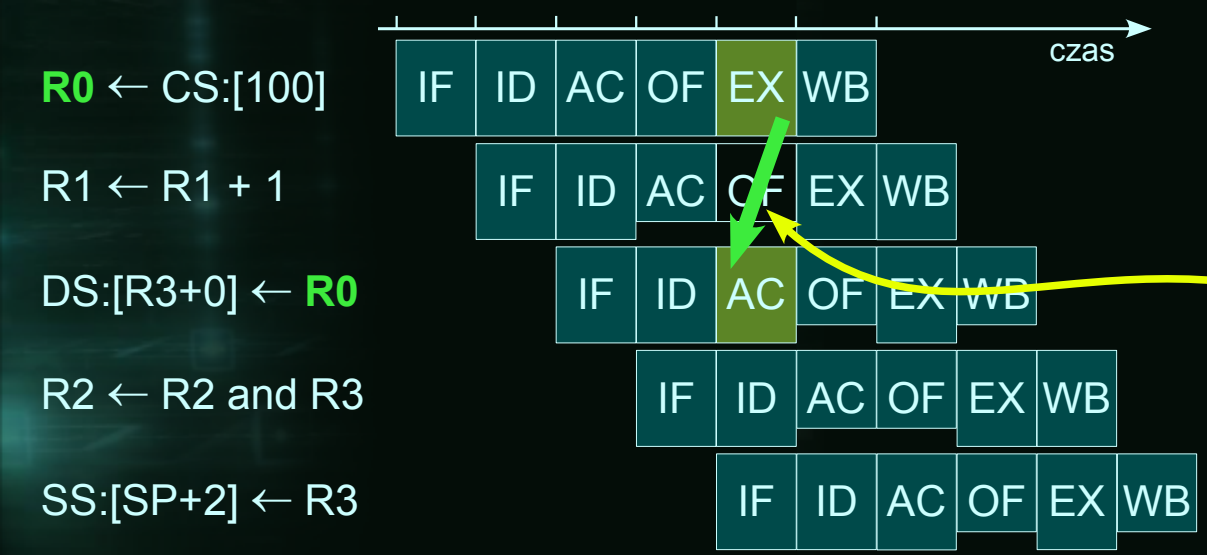
– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard danych występuje przy dostępnie do pamięci lub rejestrów.
Hazard występuje w odmianach: RAW, WAR i WAW.



Rozwiązania problemu:
- przekierowania – np. połączenie wyjścia ALU i wejścia układu adresowego,
- zmiana kolejności wykonania instrukcji,
- wprowadzenie opóźnień (stall / bubble),
- każde z powyższych.

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

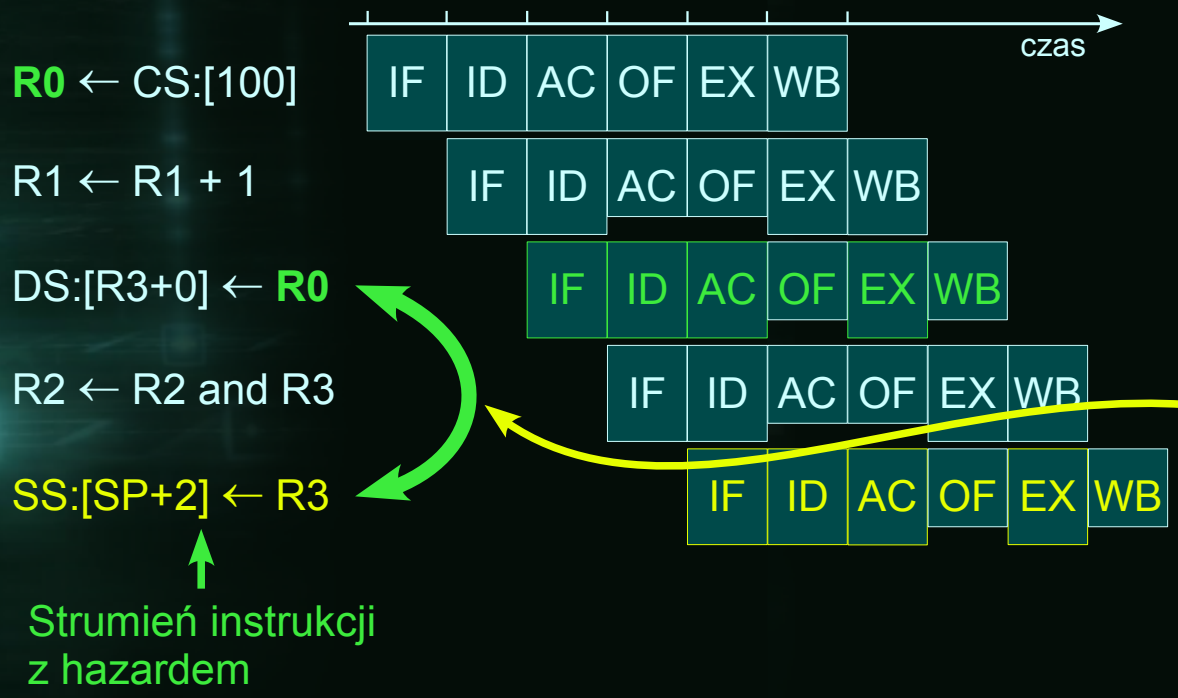
– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard danych występuje przy dostępnie do pamięci lub rejestrów.
Hazard występuje w odmianach: RAW, WAR i WAW.



Rozwiązania problemu:
- przekierowania – np. połączenie wyjścia ALU i wejścia układu adresowego,
- zmiana kolejności wykonania instrukcji,
- wprowadzenie opóźnień (stall / bubble),
- każde z powyższych.

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

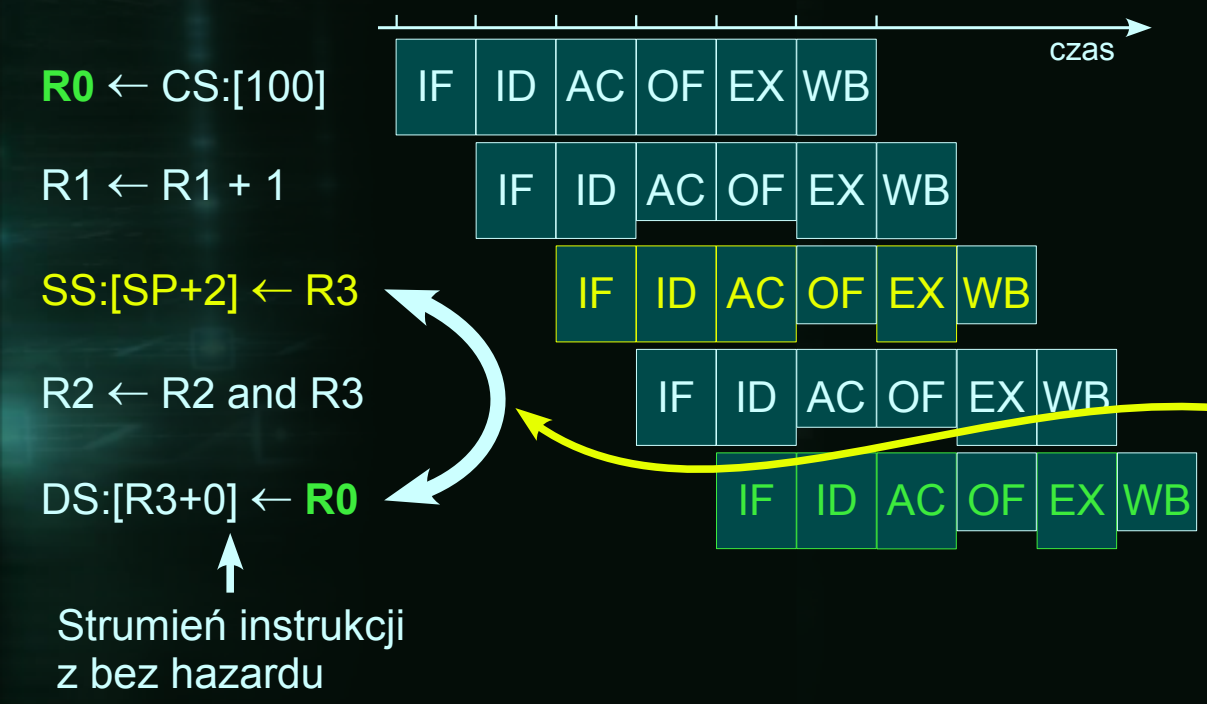
– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard danych występuje przy dostępnie do pamięci lub rejestrów.
Hazard występuje w odmianach: RAW, WAR i WAW.



Rozwiązania problemu:
- przekierowania – np. połączenie wyjścia ALU i wejścia układu adresowego,
- zmiana kolejności wykonania instrukcji,
- wprowadzenie opóźnień (stall / bubble),
- każde z powyższych.

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

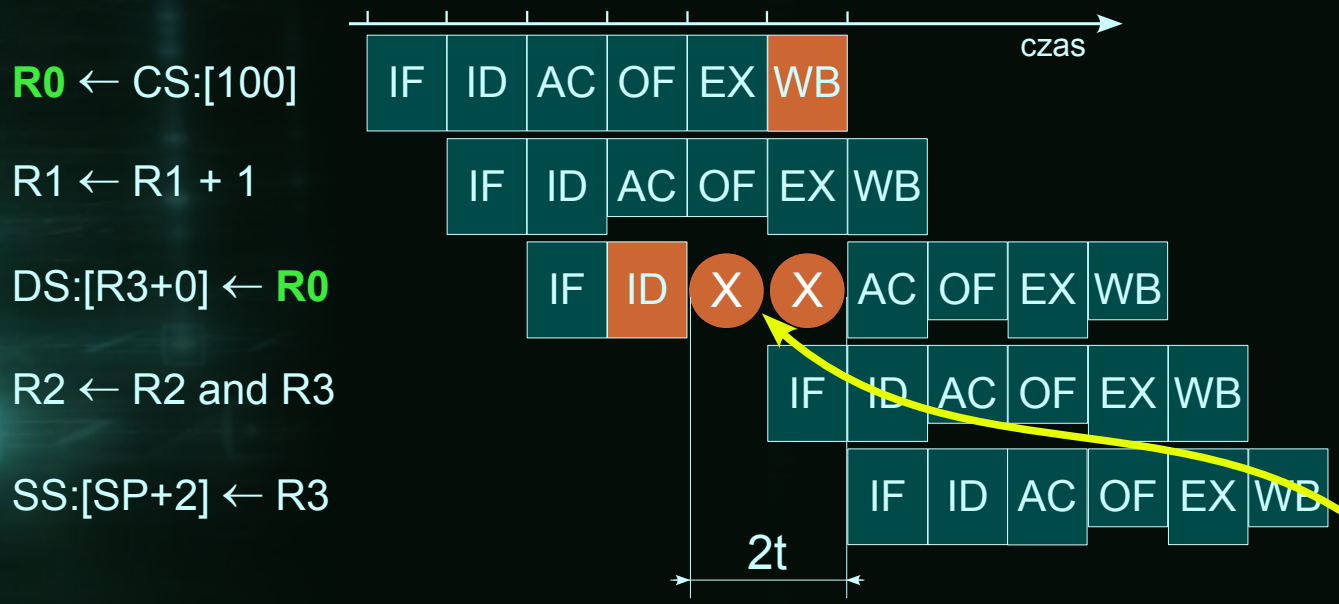
– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard danych występuje przy dostępnie do pamięci lub rejestrów.
Hazard występuje w odmianach: RAW, WAR i WAW.



Rozwiązania problemu:
- przekierowania – np. połączenie wyjścia ALU i wejścia układu adresowego,
- zmiana kolejności wykonania instrukcji,
- wprowadzenie opóźnień (stall / bubble),
- każde z powyższych.

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

- wstęp
- problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard sterowania występuje w sytuacji, gdy nie jest możliwe określenie następnej instrukcji jaką powinien wykonać procesor. Problem dotyczy przede wszystkim skoków warunkowych (ang. conditional branch).

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

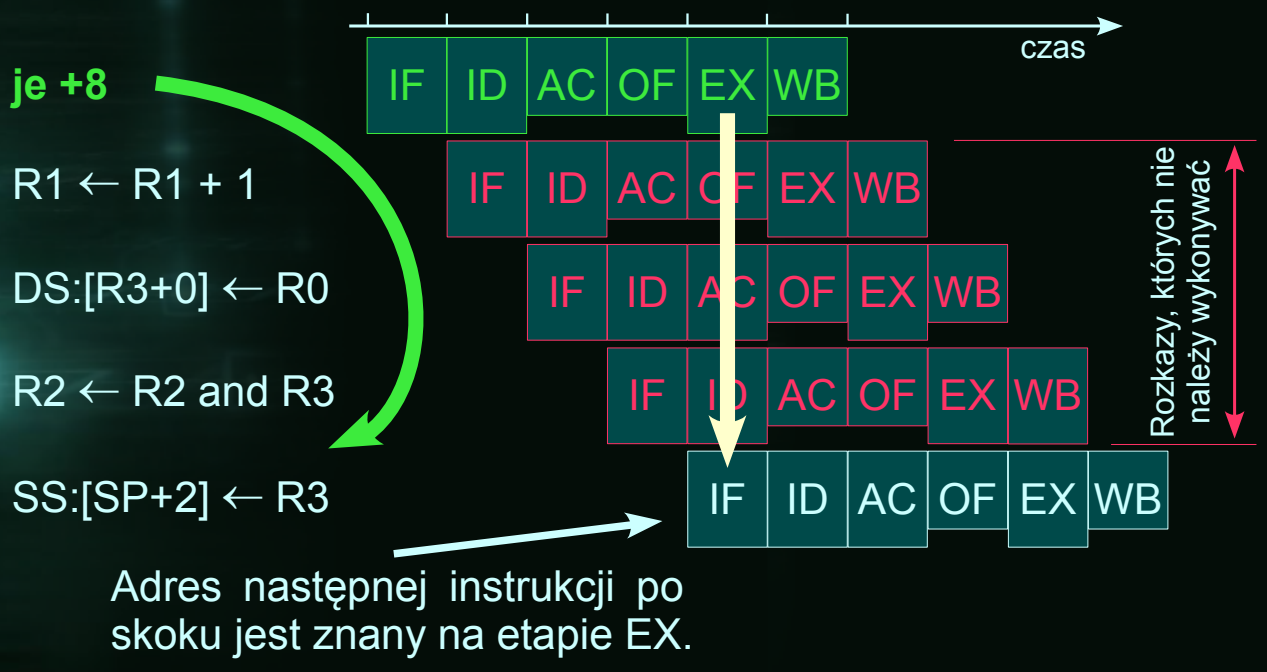
– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard sterowania występuje w sytuacji, gdy nie jest możliwe określenie następnej instrukcji jaką powinien wykonać procesor. Problem dotyczy przede wszystkim skoków warunkowych (ang. conditional branch).



Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard sterowania występuje w sytuacji, gdy nie jest możliwe określenie następnej instrukcji jaką powinien wykonać procesor. Problem dotyczy przede wszystkim skoków warunkowych (ang. conditional branch).

Rozwiązania problemu:
- blokowanie potoku do czasu osiągnięcia właściwej instrukcji,
- przewidywanie skoków:
  - statyczne,
  - dynamiczne (heurystyczne).
- opóźnienie wykonania skoku.

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

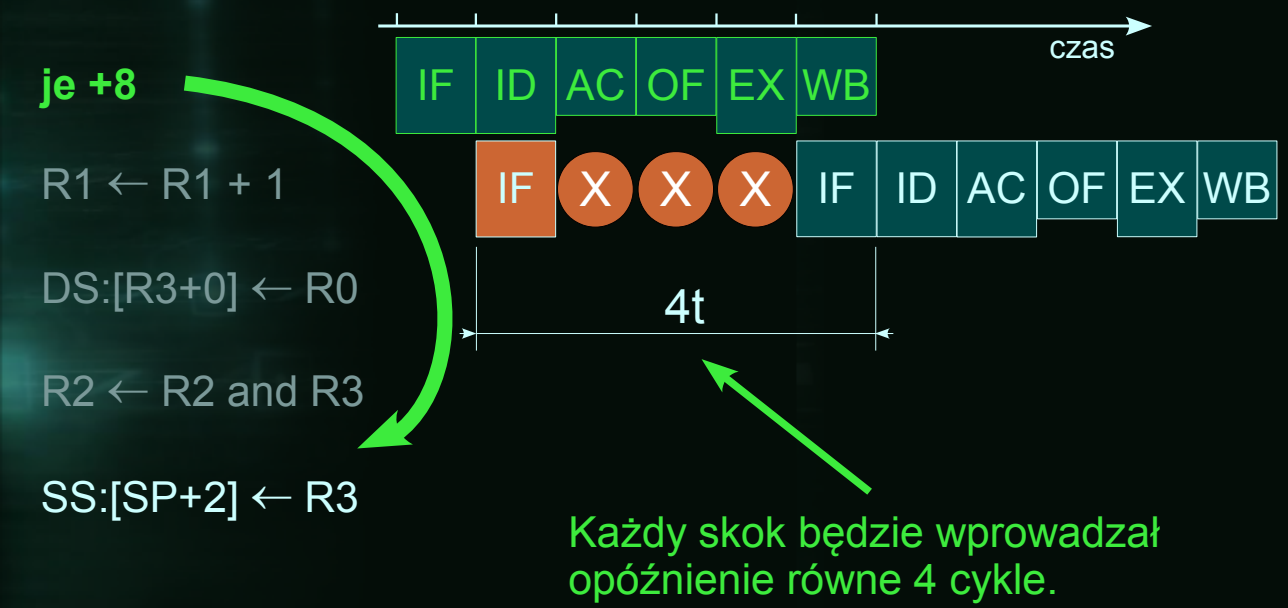
– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard sterowania występuje w sytuacji, gdy nie jest możliwe określenie następnej instrukcji jaką powinien wykonać procesor. Problem dotyczy przede wszystkim skoków warunkowych (ang. conditional branch).



Rozwiązania problemu:
- blokowanie potoku do czasu osiągnięcia właściwej instrukcji,
- przewidywanie skoków:
  - statyczne,
  - dynamiczne (heurystyczne).
- opóźnienie wykonania skoku.

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

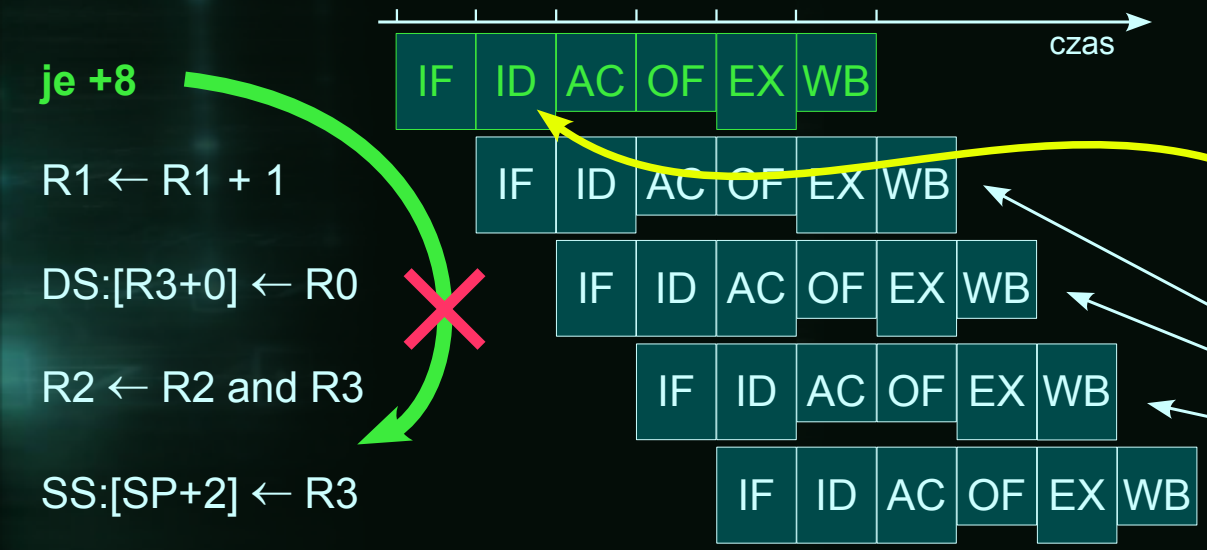
– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard sterowania występuje w sytuacji, gdy nie jest możliwe określenie następnej instrukcji jaką powinien wykonać procesor. Problem dotyczy przede wszystkim skoków warunkowych (ang. conditional branch).



Rozwiązania problemu:
- blokowanie potoku do czasu osiągnięcia właściwej instrukcji,
- przewidywanie skoków:
  - statyczne,
  - dynamiczne (heurystyczne).
- opóźnienie wykonania skoku.

Predykcje:
- uznanie, że skok NIE zostanie wykonany,
- inne strategie.

Instrukcje za skokiem zostaną wykonane – brak opóźnienia.

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

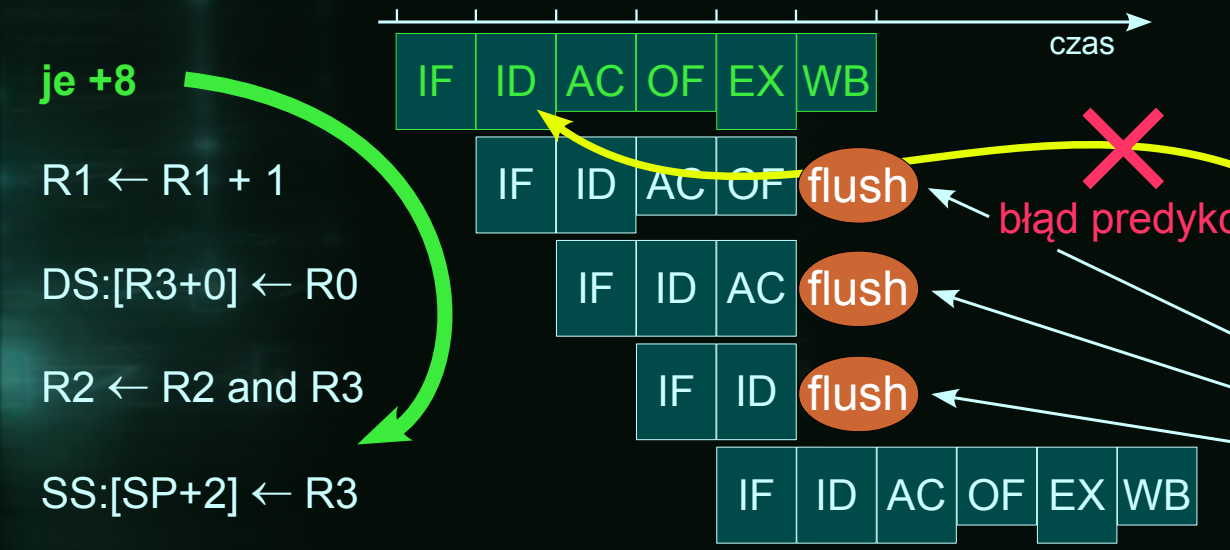
– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard sterowania występuje w sytuacji, gdy nie jest możliwe określenie następnej instrukcji jaką powinien wykonać procesor. Problem dotyczy przede wszystkim skoków warunkowych (ang. conditional branch).



Rozwiązania problemu:
- blokowanie potoku do czasu osiągnięcia właściwej instrukcji,
- przewidywanie skoków:
  - statyczne,
  - dynamiczne (heurystyczne).
- opóźnienie wykonania skoku.

Predykcje:
- uznanie, że skok NIE zostanie wykonany,
- inne strategie.

Instrukcje w potoku nie wykonają zapisu pamięci / rejestrów

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard sterowania występuje w sytuacji, gdy nie jest możliwe określenie następnej instrukcji jaką powinien wykonać procesor. Problem dotyczy przede wszystkim skoków warunkowych (ang. conditional branch).

Rozwiązania problemu:
- blokowanie potoku do czasu osiągnięcia właściwej instrukcji,
- przewidywanie skoków:
  - statyczne,
  - dynamiczne (heurystyczne).
- opóźnienie wykonania skoku.

Strategie statyczne przewidywania skoków:
- uznanie skoków powrotnych (malejących adresów), jako zawsze wykonywanych, a skoków do adresów rosnących jako nigdy nie wykonywanych:
  - adresy powrotów zawierają pętle programowe, zazwyczaj wykonywane wielokrotnie,
  - konstrukcja „if-then-else“ często wykonuje instrukcje po słowie „then“ - brak skoku.
- kompilator może przewidywać fakt wykonania skoku i ustawiać w instrukcji bit „predict taken / not taken“.

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

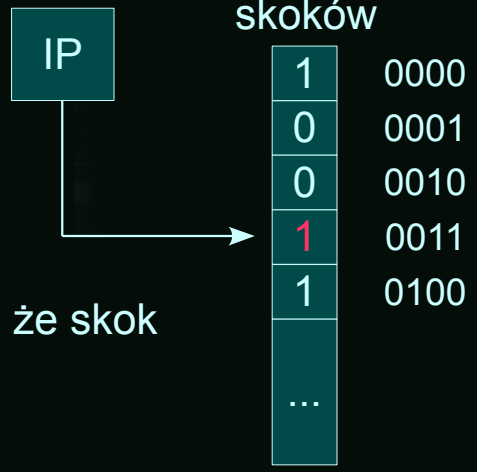
Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard sterowania występuje w sytuacji, gdy nie jest możliwe określenie następnej instrukcji jaką powinien wykonać procesor. Problem dotyczy przede wszystkim skoków warunkowych (ang. conditional branch).

Rozwiązania problemu:
- blokowanie potoku do czasu osiągnięcia właściwej instrukcji,
- przewidywanie skoków:
  - statyczne,
  - dynamiczne (heurystyczne).
- opóźnienie wykonania skoku.

Wartość „1” w tablicy oznacza, że ostatnim razem gdy rejestr licznika programu (IP) zawierał wartość kończącą się na 0011, to skok został wykonany.

Kontroler CPU będzie przypuszczał, że skok zostanie wykonany ponownie.




```
for(i=0, i < 33; i++)
{
   …
}
```

→

```
for_loop:
...
R3 ← R3 + 1
cmp R3, 32
jne for_loop
```

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

– wstęp

– problem hazardu

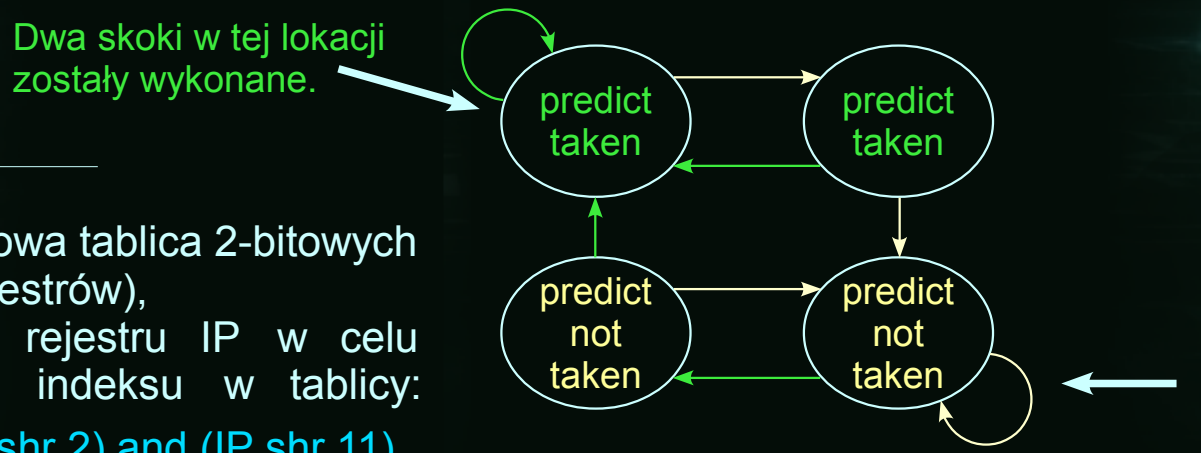
Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard sterowania występuje w sytuacji, gdy nie jest możliwe określenie następnej instrukcji jaką powinien wykonać procesor. Problem dotyczy przede wszystkim skoków warunkowych (ang. conditional branch).

Rozwiązania problemu:
- blokowanie potoku do czasu osiągnięcia właściwej instrukcji,
- przewidywanie skoków:
  - statyczne,
  - dynamiczne (heurystyczne).
- opóźnienie wykonania skoku.

Predykcja dynamiczna dwubitowa z nasycaniem:



Implementacja:
- 512-elementowa tablica 2-bitowych liczników (rejestrów),
- hash'owanie rejestru IP w celu wyznaczenia indeksu w tablicy:

index = (IP shr 2) and (IP shr 11)

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

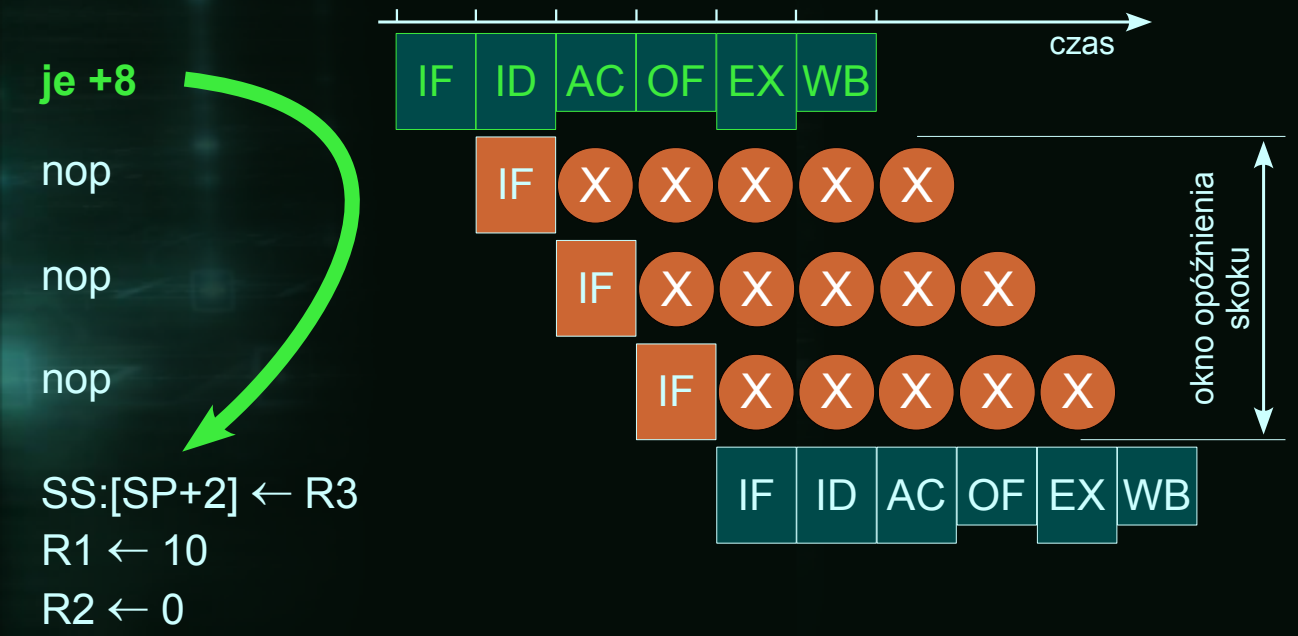
– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard sterowania występuje w sytuacji, gdy nie jest możliwe określenie następnej instrukcji jaką powinien wykonać procesor. Problem dotyczy przede wszystkim skoków warunkowych (ang. conditional branch).



Rozwiązania problemu:
- blokowanie potoku do czasu osiągnięcia właściwej instrukcji,
- przewidywanie skoków:
  - statyczne,
  - dynamiczne (heurystyczne).
- opóźnienie wykonania skoku.

1. Szerokość okna opóźnienia skoku można zmniejszyć przez wyprowadzenie adresu skoku ze wcześniejszego stopnia potoku.
2. Skok warunkowy nie musi zostać wykonany natychmiastowo.

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

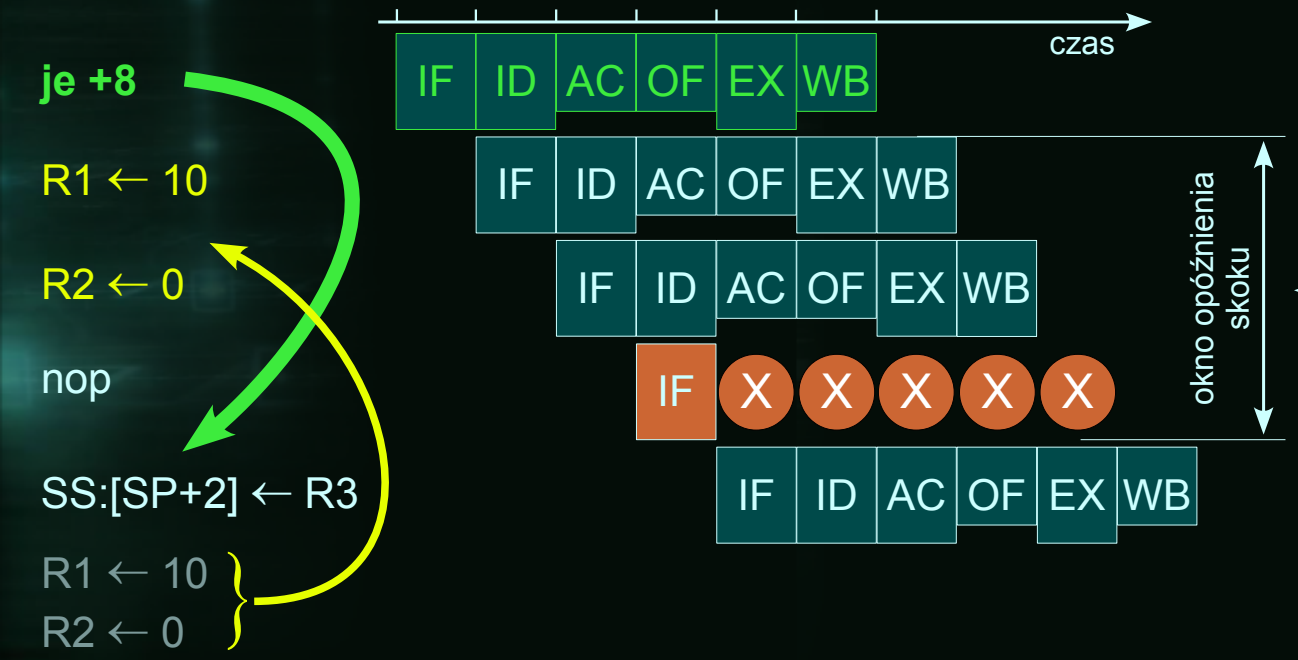
– wstęp

– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard sterowania występuje w sytuacji, gdy nie jest możliwe określenie następnej instrukcji jaką powinien wykonać procesor. Problem dotyczy przede wszystkim skoków warunkowych (ang. conditional branch).



Rozwiązania problemu:
- blokowanie potoku do czasu osiągnięcia właściwej instrukcji,
- przewidywanie skoków:
  - statyczne,
  - dynamiczne (heurystyczne).
- opóźnienie wykonania skoku.

Kompilator może wykorzystać okno opóźnienia skoku i umieścić w nim instrukcje, które nie są zależne od segmentu instrukcji związanych ze skokiem.

Zjawisko hazardu

omówienie problemów związanych z pracą potokową w CPU

– wstęp

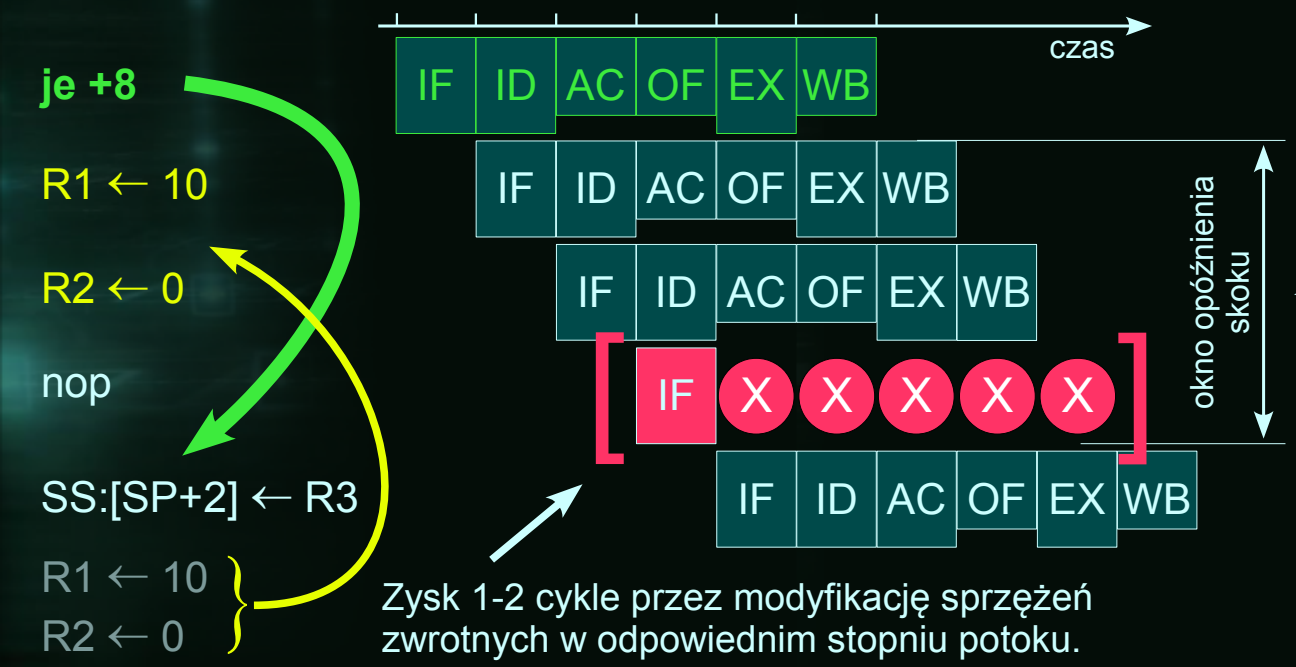
– problem hazardu

Wpływ hazardów na potok można niwelować stosując pewne rozwiązania sprzętowe oraz programowe.

Hazard w potoku:
- strukturalny
- danych
- sterowania

Hazard sterowania występuje w sytuacji, gdy nie jest możliwe określenie następnej instrukcji jaką powinien wykonać procesor. Problem dotyczy przede wszystkim skoków warunkowych (ang. conditional branch).

Rozwiązania problemu:
- blokowanie potoku do czasu osiągnięcia właściwej instrukcji,
- przewidywanie skoków:
  - statyczne,
  - dynamiczne (heurystyczne).
- opóźnienie wykonania skoku.



Zysk 1-2 cykle przez modyfikację sprzężeń zwrotnych w odpowiednim stopniu potoku.

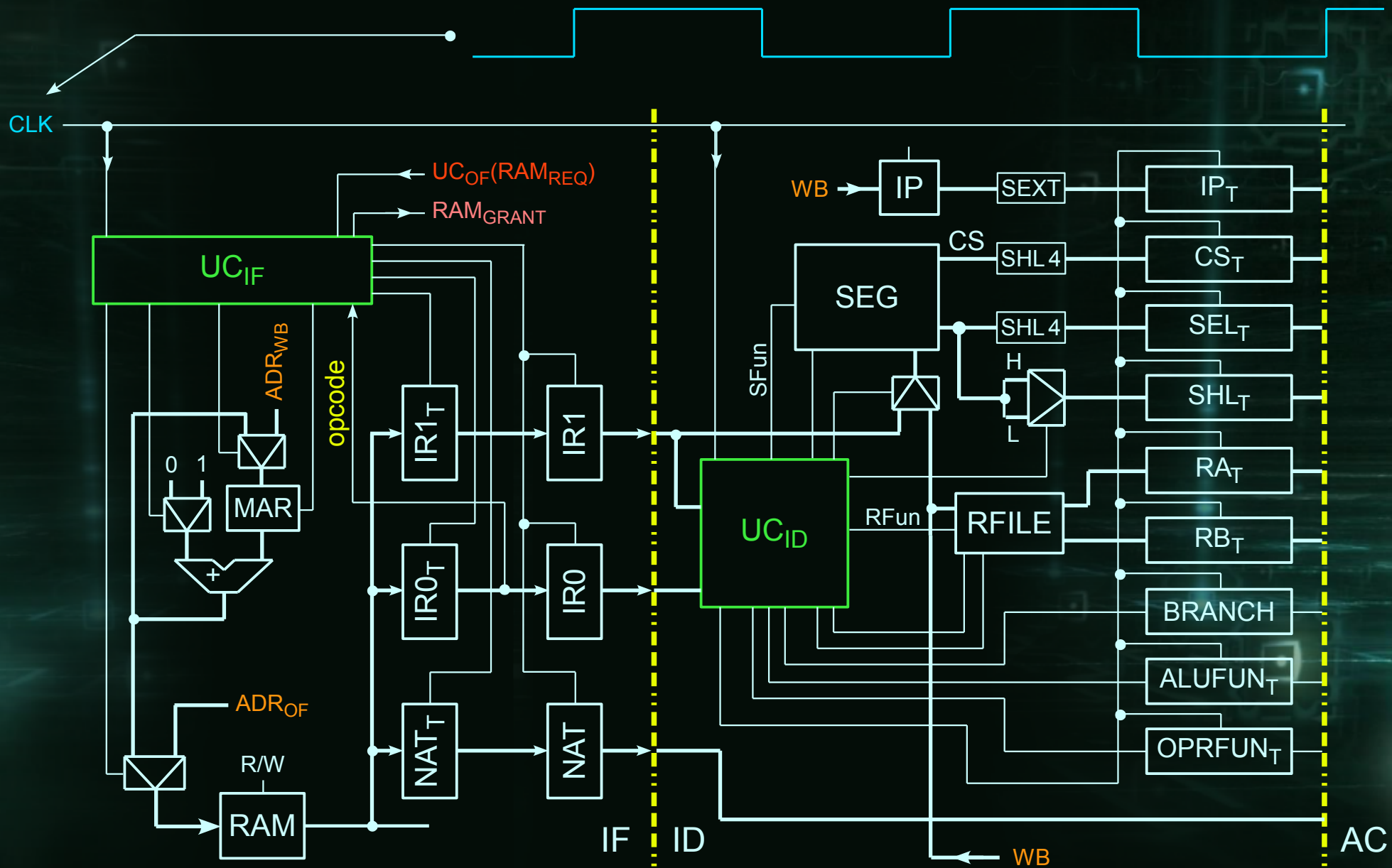
Kompilator może wykorzystać okno opóźnienia skoku i umieścić w nim instrukcje, które nie są zależne od segmentu instrukcji związanych ze skokiem.

Projekt μP – kontroler w potoku

Projekt µP – kontroler w potoku

sterowanie potokiem – pierwsze 2 stopnie

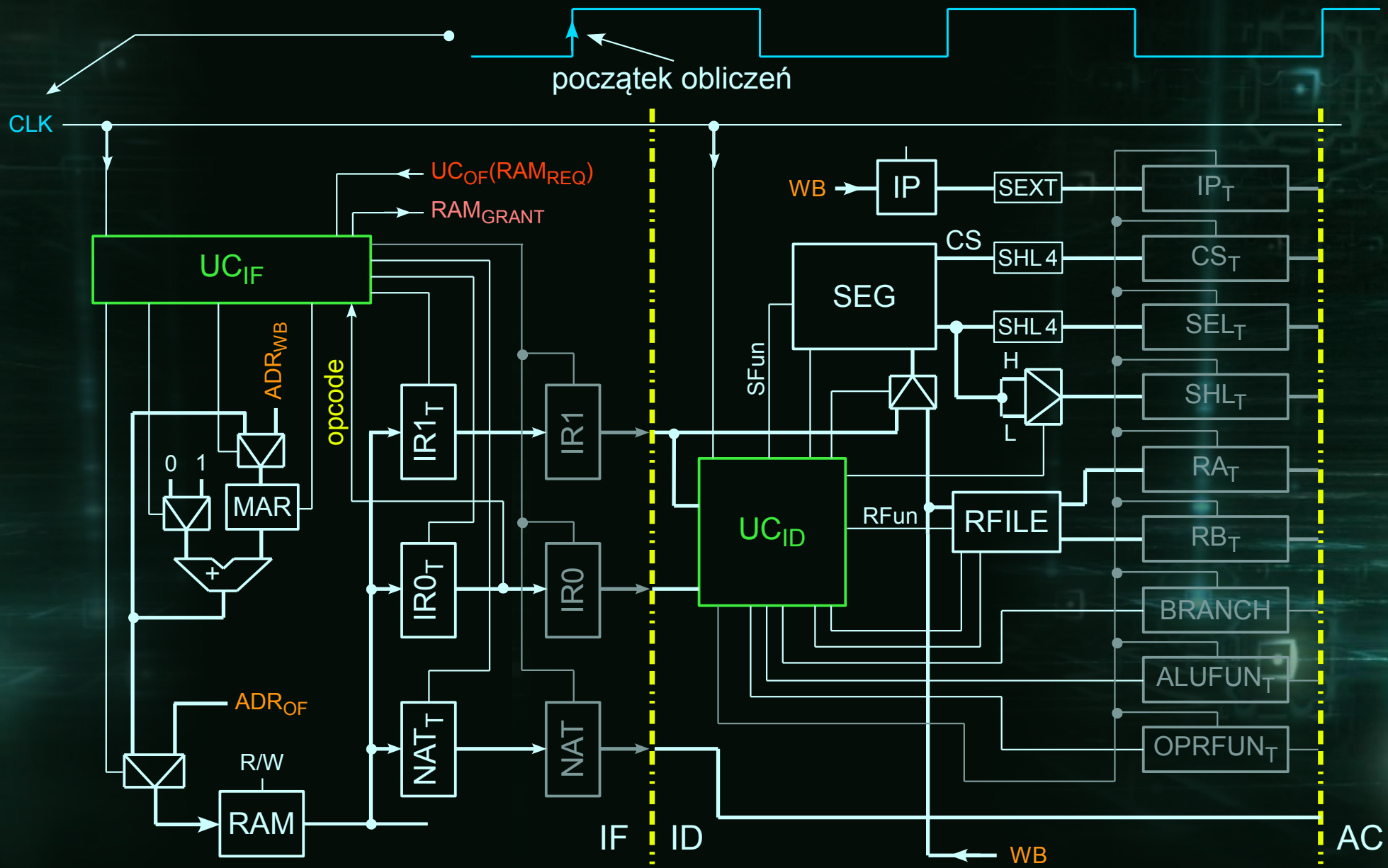
Stopnie potoku: IF oraz ID



Projekt µP – kontroler w potoku

sterowanie potokiem – pierwsze 2 stopnie

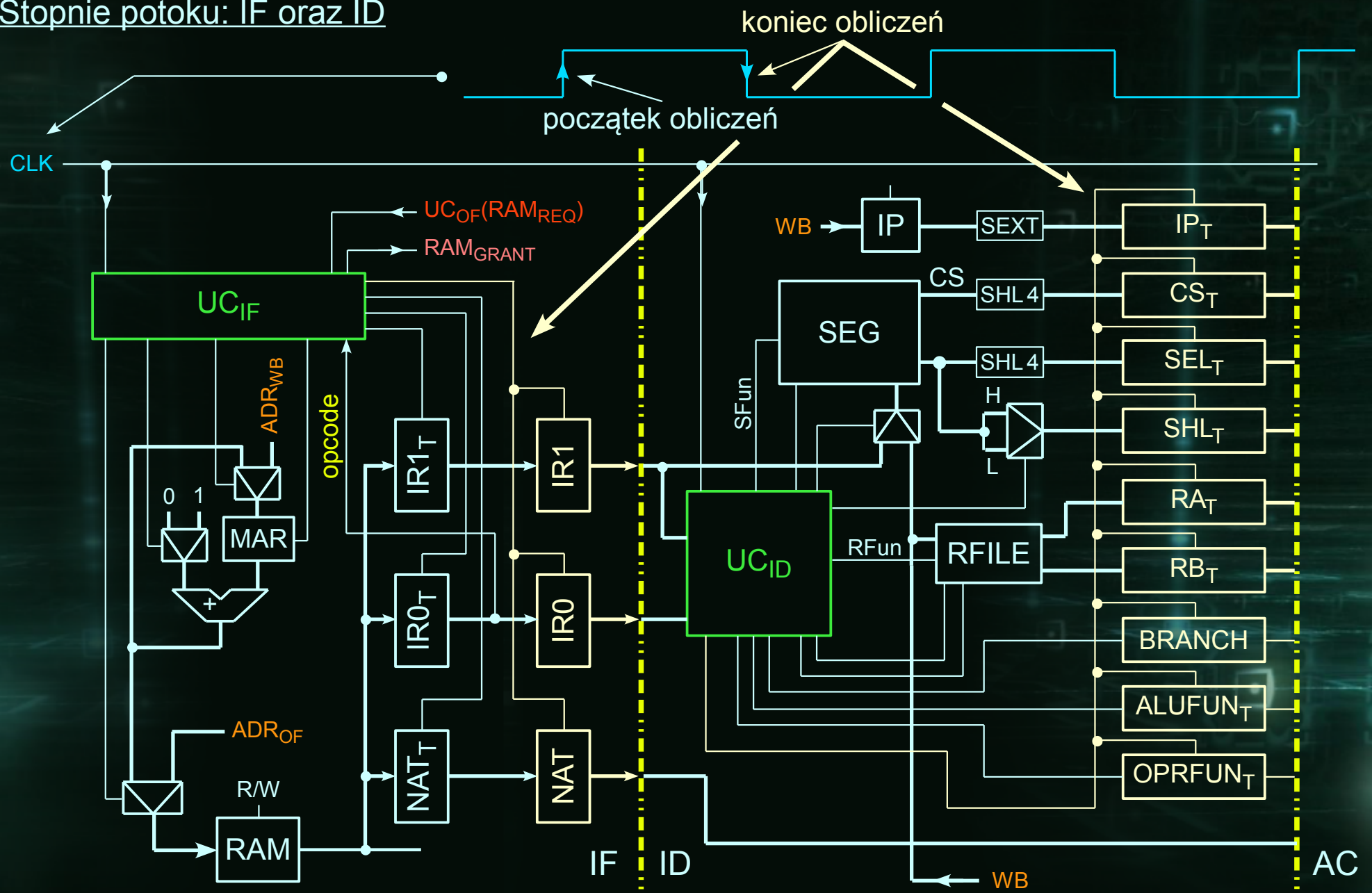
Stopnie potoku: IF oraz ID



Projekt µP – kontroler w potoku

sterowanie potokiem – pierwsze 2 stopnie

Stopnie potoku: IF oraz ID



Projekt µP – kontroler w potoku

sterowanie potokiem – pierwsze 2 stopnie

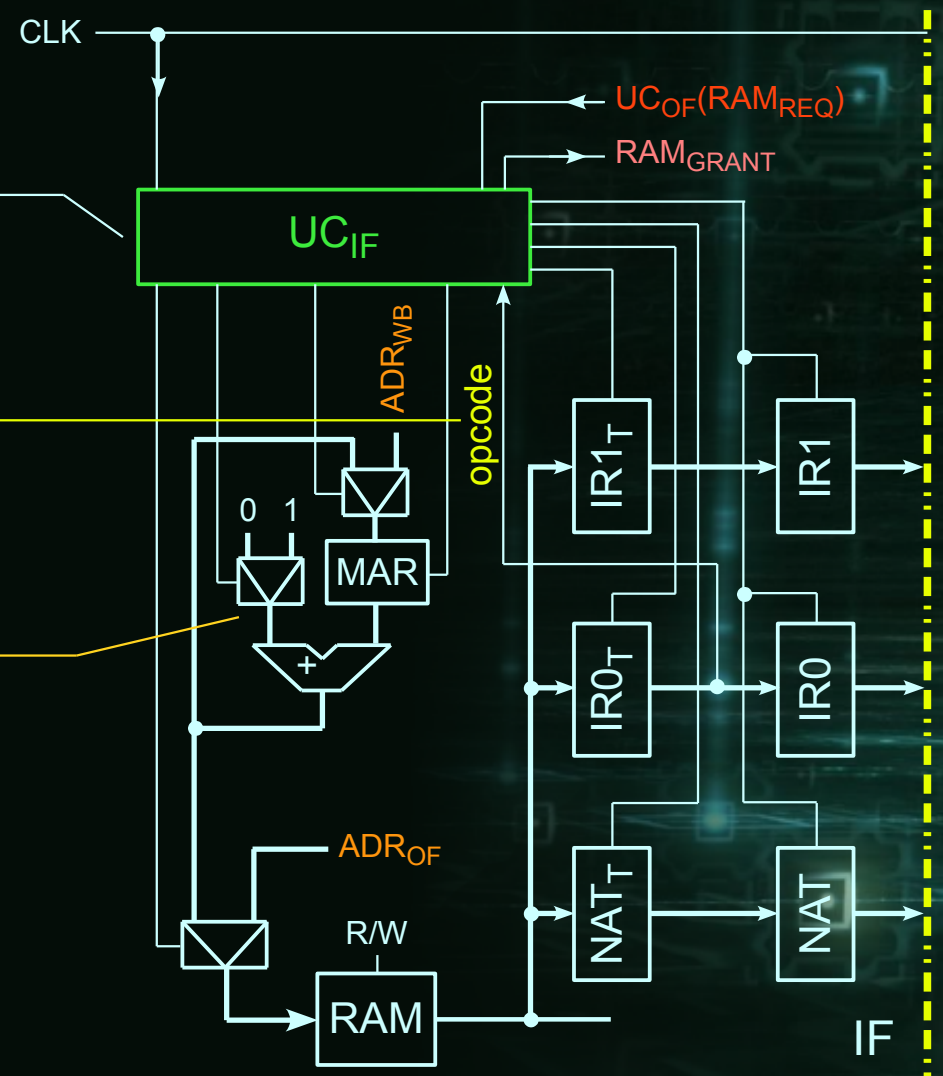
Stopień potoku: IF – pobranie rozkazu (początek obliczeń)

START:

1. STATE = OPR, I_{SIZE} = 0, I_{TARGET} = 0,
   $ADR_{MUX+} \leftarrow 0$, $ADR_{MUXRAM} \leftarrow 0$,
   $ADR_{WB} \leftarrow 1$
2. MAR $\leftarrow ADR_{RAM}$
3. IR[I_{TARGET}] $\leftarrow$ RAM[MAR]
4. jeśli I_{TARGET} == 0 to
   4.1. pre-decode IR[0]
   4.2. jeśli I == typ 1 to I_{SIZE} = 1
   4.3. jeśli I == typ 2 to I_{SIZE} = 2
   4.4. $ADR_{WB} \leftarrow 0$, $ADR_{MUX+} \leftarrow 1$
5. I_{TARGET} = I_{TARGET} + 1
6. jeśli $I_{SIZE} \neq 0$ to
   6.1. I_{SIZE} = I_{SIZE} – 1
   6.2. skok do etapu 2

KONIEC:

7. STATE = IDLE



W projekcie CPU układy kontrolerów będą układami asynchronicznymi (mikroprogramowanymi lub nie).

Projekt µP – kontroler w potoku

sterowanie potokiem – pierwsze 2 stopnie

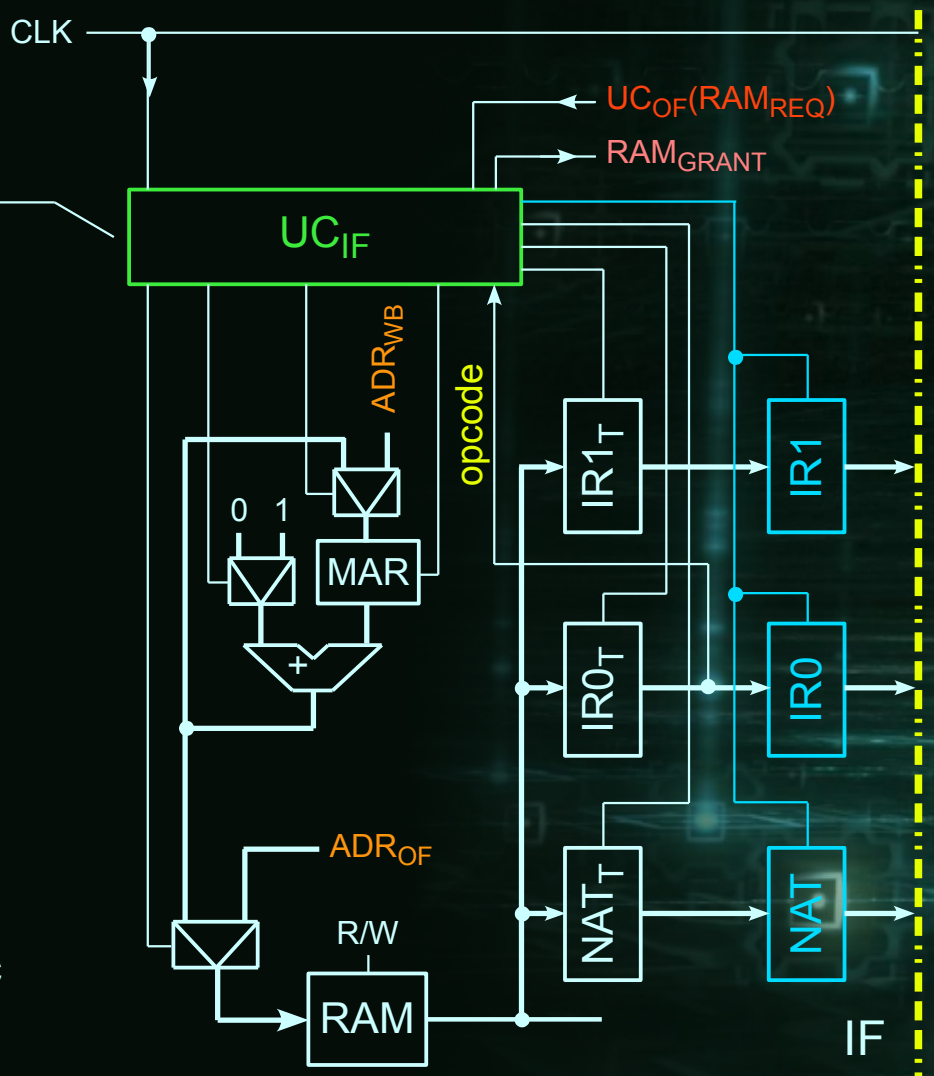
Stopień potoku: IF – pobranie rozkazu (koniec obliczeń)

START:

1. $REG_{WR} \leftarrow 0$
2. jeśli STATE == IDLE to
   - 2.1. STATE = REGWR
   - 2.2. $REG_{WR} \leftarrow 1$
   - 2.3. STATE = OPR, $REG_{WR} \leftarrow 0$
   - 2.4. skok do etapu 2
3. jeśli STATE ≠ IDLE to
   - 3.1. skok do etapu 2

KONIEC:

zmienna stanu



W projektowanym CPU, poza pobieraniem rozkazów kontroler stopnia IF powinien dodatkowo obsługiwać:

- arbitraż dostępu do pamięci RAM,
- obsługę przerwań.

Ponadto każdy z kontrolerów powinien w razie potrzeby informować pozostałe układu (kontrolery) o zakończeniu wykonywanych operacji, pozwalając zaplanować dostęp do wspólnych zasobów.

Projekt µP – kontroler w potoku

sterowanie potokiem – pierwsze 2 stopnie

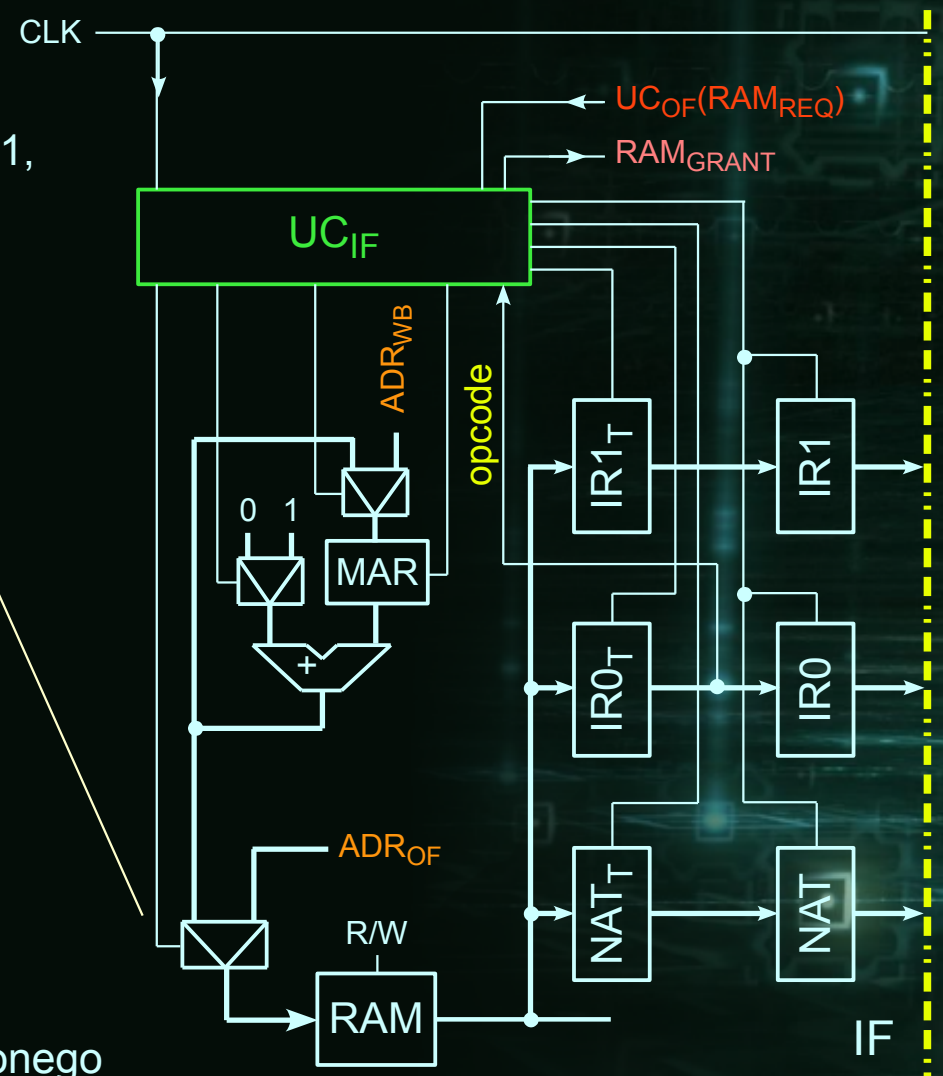
Stopień potoku: IF – arbitraż dostępu do RAM

START:

1. STATE = OPR, I_{SIZE} = 0, I_{TARGET} = 0,
   $ADR_{MUX+} \leftarrow 0$, $ADR_{MUXRAM} \leftarrow 0$, $ADR_{WB} \leftarrow 1$,
   $RAM_{GRANT} \leftarrow 0$
2. jeśli $UC_{OF}(RAM_{REQ})$ == 0 to
   2.1. $MAR \leftarrow ADR_{RAM}$, $ADR_{MUXRAM} \leftarrow 0$,
   $RAM_{GRANT} \leftarrow 0$
   2.2 skok do etapu 4
3. jeśli $UC_{OF}(RAM_{REQ})$ == 1 to
   3.1. $ADR_{MUXRAM} \leftarrow 1$
   3.2. $RAM_{GRANT} \leftarrow 1$
   3.3. skok do etapu 2
4. $IR[I_{TARGET}] \leftarrow RAM[MAR]$
5. jeśli I_{TARGET} == 0 to
   ...

KONIEC:

W projektowanym CPU nie będzie jawnie wyróżnionego układu sterującego RAM, dlatego odpowiednie kontrolery powinny generować sygnały sterujące dla pamięci.



Potoku w rodzinie x86

Potok w rodzinie x86

cechy przetwarzania potokowego w procesorach rodziny x86

- rodzina x86
- wpływ na wydajność

Wydajność CPU z przetwarzaniem potokowym zależy również od konstrukcji potoku. Przykład rodziny x86 pokazuje zagrożenia związane tą technologią.

Pojęcia:
- czas pracy CPU
- potok w Pentium III
- potok w Pentium IV

Potok w rodzinie x86

cechy przetwarzania potokowego w procesorach rodziny x86

- rodzina x86
- wpływ na wydajność

Wydajność CPU z przetwarzaniem potokowym zależy również od konstrukcji potoku. Przykład rodziny x86 pokazuje zagrożenia związane tą technologią.

Pojęcia:
- czas pracy CPU
- potok w Pentium III
- potok w Pentium IV

Jest to wielkość określana przez:
- liczbę wykonanych instrukcji,
- liczbę cykli zegara na instrukcję (CPI),
- czas trwania jednego cyklu zegara.

Odpowiednio:

T_{CPU} = (wykonane rozkazy) $*$ CPI $*$ T_C

można zmienić sprzętowo

Potok w rodzinie x86

cechy przetwarzania potokowego w procesorach rodziny x86

– rodzina x86

– wpływ na wydajność

Wydajność CPU z przetwarzaniem potokowym zależy również od konstrukcji potoku. Przykład rodziny x86 pokazuje zagrożenia związane tą technologią.

Pojęcia:
- czas pracy CPU
- potok w Pentium III
- potok w Pentium IV

Jest to wielkość określana przez:
- liczbę wykonanych instrukcji,
- liczbę cykli zegara na instrukcję (CPI),
- czas trwania jednego cyklu zegara.

Odpowiednio:

T_{CPU} = (wykonane rozkazy) $*$ CPI $*$ T_C

można zmienić sprzętowo

Potok w rodzinie P6 składa się z 10 stopni, a jego konstrukcja jest dostosowana do obsługi przewidywania rozgałęzień.

Potok procesora Pentium III									
IF	IF	ID	ID	ID	RN	ROB	Rdy/ Sch	DIS	EX

Stopnie:
- RN – register rename,
- ROB – reorder buffer,
- Rdy/Sch – instruction scheduler,
- DIS – dispatcher.

Potok w rodzinie x86

cechy przetwarzania potokowego w procesorach rodziny x86

– rodzina x86

– wpływ na wydajność

Wydajność CPU z przetwarzaniem potokowym zależy również od konstrukcji potoku. Przykład rodziny x86 pokazuje zagrożenia związane tą technologią.

Pojęcia:
- czas pracy CPU
- potok w Pentium III
- potok w Pentium IV

Jest to wielkość określana przez:
- liczbę wykonanych instrukcji,
- liczbę cykli zegara na instrukcję (CPI),
- czas trwania jednego cyklu zegara.

Odpowiednio:

$$T_{CPU} = (\text{wykonane rozkazy}) * CPI * T_C$$

można zmienić sprzętowo

Struktura potoku w procesorze Pentium IV (netburst):

TC next IP	– trace cache next instruction pointer,
TC Fetch	– trace cache fetch,
Drive	– signals drive,
Alloc	– resources allocation,
Rename	– register rename,
Queue	– μopr queue,
Schedule	– μopr schedule,
Dispatch	– μopr dispach,
Reg file	– internal register read,
Execute	– μopr execute,
Flags	– CPU flag update,
Bra check	– check branch predicted,
Drive	– signals drive.

Stopnie potoku wprowadzają opóźnienie potrzebne dla propagacji sygnałów między modułami CPU.

Potok w rodzinie P6 składa się z 10 stopni, a jego konstrukcja jest dostosowana do obsługi przewidywania rozgałęzień.

Potok procesora Pentium III									
IF	IF	ID	ID	ID	RN	ROB	Rdy/ Sch	DIS	EX

Stopnie:
- RN – register rename,
- ROB – reorder buffer,
- Rdy/Sch – instruction scheduler,
- DIS – dispatcher.

Potok w rodzinie x86

cechy przetwarzania potokowego w procesorach rodziny x86

- rodzina x86
- wpływ na wydajność

Zwiększanie liczby stopni pokoku może mieć negatywne konsekwencje, głównie spowodowane przez znajwisko hazardu, które wymaga podjęcia dodatkowych czynności związanych z ich obsługą.

Koszt zwiększania liczby stopni potoku:

CPU	Rok	Taktowanie (w MHz)	Głębokość potoku
i486	1989	25	5
Pentium	1993	66	5
Pentium Pro	1997	200	10
P4 Willmette	2001	2000	22
P4 Prescott	2004	3600	31
Core 2 Conroe	2006	2930	14
Core 2 Yorkfield	2008	2930	16
Core i7 Gulftown	2010	3460	16

Jaki jest powód ?

Potok w rodzinie x86

cechy przetwarzania potokowego w procesorach rodziny x86

– rodzina x86

– wpływ na wydajność

Zwiększanie liczby stopni pokoku może mieć negatywne konsekwencje, głównie spowodowane przez znajwisko hazardu, które wymaga podjęcia dodatkowych czynności związanych z ich obsługą.

Koszt zwiększania liczby stopni potoku:

CPU	Rok	Taktowanie (w MHz)	Głębokość potoku	Pobór mocy (w W)
i486	1989	25	5	5
Pentium	1993	66	5	10
Pentium Pro	1997	200	10	29
P4 Willmette	2001	2000	22	75
P4 Prescott	2004	3600	31	103
Core 2 Conroe	2006	2930	14	75
Core 2 Yorkfield	2008	2930	16	95
Core i7 Gulftown	2010	3460	16	130

Koszt obsługi potoku z dużą liczbą stopni. Ponadto wydajność znacząco maleje.

Zagrożenia związane z pracą potokową:

- spowodowane hazardami – konieczność blokowania (stall) lub opróżniania potoku (flush),
- zbyt długi potok wprowadza dłuższe opóźnienie,
- procesory z zaawansowanymi potokami wykonują wiele rozkazów spekulatywnie. Jeśli wyniki obliczeń nie zostają zatwierdzone to ma to negatywny wpływ wydajność,
- zazwyczaj obsługa potoku wymaga zastosowania złożonych układów kontrolno-sterujących, np. układu predykcji skoków, kolejek rozkazów, układów reorganizacji instrukcji, itp.

Koniec wykładu