Architektury systemów komputerowych

Lista 8

$x_8 = 7$ (minimum na bdb)

1. W tym zadaniu rozważamy procesor, w którym poszczególne fazy wykonywania rozkazów trwają odpowiednio: IF 300ps, ID 400ps, EX 250ps, MEM 500ps, WB 100ps. Jaka będzie długość cyklu zegarowego w przypadku procesora jednocyklowego, a jaka w przypadku procesora z przetwarzaniem potokowym? Załóżmy, że możemy rodzielić jedną z faz na dwie mniejsze (każdą o czasie działania równym połowie wyjściowej fazy). Którą fazę najlepiej wybrać? Jak takie rozdzielenie wpłynie na długość cyklu? To samo zadanie rozwiąż przy założeniu, że fazy trwają odpowiednio: IF 200ps, ID 150ps, EX 120ps, MEM 190ps, WB 140ps.

2. Wzorując się na rysunku ze slajdu 9, wykład 8, pokaż ścieżki forwardowania danych dla następującego kodu:

```
add $3, $4, $6
sub $5, $3, $2
lw  $7, 100($5)
add $8, $7, $2
```

3. Znajdź wszystkie zależności danych w poniższym kodzie. Które będą hazardami (nieusuwalnymi), a które zostaną rozwiązane przez forwarding?

```
add $3, $4, $2
sub $5, $3, $1
lw  $5, 100($3)
add $7, $3, $6
```

4. Rozważmy następujący ciąg rozkazów:

```
lw $1, 40($6)
add $6, $2, $2
sw $6, 50, ($1)
```

   Znajdź wszystkie zależności danych. Jaki będzie czas wykonania tego ciągu rozkazów na procesorze z przetwarzaniem potokowym, jeśli założymy, że:

   (a) nie ma forwardingu, a cykl zegarowy ma 200 ps,

   (b) jest forwarding ALU-ALU, ale nie ma forwardingu MEM-ALU, a cykl zegarowy ma 220ps,

   (c) jest pałny forwarding, a cykl zegarowy ma 250 ps?

   We wszystkich przypadkach zakładamy, że procesor ma układy zapewniające wstawienie opóźnień gwarantujących poprawne wykonywanie programów.

5. Rozważmy ciąg instrukcji `lw, add, lw, add, ...` długości 1000, w którym każda instrukcja zależy dokładnie od poprzedniej instrukcji (`add` ma dodać wartość z rejestru ładowanego przez poprzedni rozkaz `lw`, a `lw` ma bazę adresu w rejestrze, który jest wyliczny przez poprzedni rozkaz `add`. Ile cykli zabierze wykonanie tego ciągu procesorem ze sladju 33, wykład 8, część B? Ile zajęłoby, gdybyśmy nie używali forwardingu?

6. Rozważmy następujące dwa fragmenty kodu:

```
a)
lw  $1, 40, ($6)
add $2, $3, $1
add $1, $6, $4
sw  $2, 20($4)
and $1, $1, $4
```

```
b)
add $1, $5, $3
sw  $1, 0($2)
lw  $1, 4($2)
add $5, $5, $1
sw  $1, 0($2)
```

Dla każdego z nich:

(a) Załóżmy, że procesor nie ma układów forwardowania danych ani wykrywania hazardów. Dopisz dodatkowe instrukcje `nop` (nic nie rób, *no operation*), które zapewnią poprawne wykananie.

(b) Spróbuj usunąć hazardy za pomocą przestawiania/modyfikacji rozkazów, zmniejszając tym samym liczbę koniecznych rozkazów `nop`. Ewentualne modyfikacje rozkazów mogą polegać na zmianie numerów rejestrów. Możesz założyć, że rejestr $7 nie jest używany przez program.

7. Rozważmy fragmenty kodu z poprzedniego zadania.

(a) Załóżmy, że procesor ma zaimplemetowane układy forwardowania, ale nie ma układu wykrywania hazardów po `lw` (*hazard detection*). Co stanie się podczas wykanywania podanego kodu?

(b) Rozważmy teraz model procesora ze slajdu 33, wykład 9. Jakie sygnały sterujące wyprodukuje *hazard detection unit* podczas pierwszych pięciu cykli wykonywania kodu?

8. (2 pkt.) W tym i w następnym zadaniu rozważmy architekturę z następującymi rozkazami dwuargumentowymi: `add, addi, mul, move`.

- Rozkaz `add rd, rs` dodaje rejestry `rd` i `rs`, a wynik zapisuje w `rd`.
- Rozkaz `mul rd, rs` mnoży rejestr `rd` przez `rs`, a wynik zapisuje w `rd`.
- Rozkaz `addi rd, const` dodaje stałą `const` do rejestru `rd`.
- Rozkaz `move rd, rs` kopiuje zawartość rejestru `rs` do rejestru `rd`.

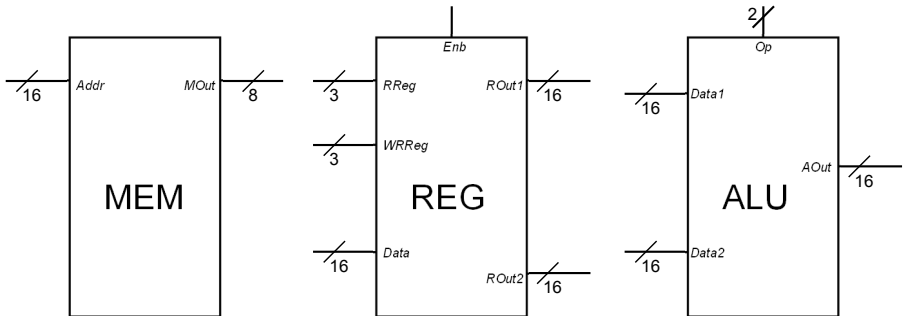
Projektować będziemy procesor realizujący tą listę rozkazów. Zakładamy, że procesor będzie dysponował plikiem ośmiu, 16-bitowych rejestrów ogólnego przeznaczenia. Mamy tylko pamięć rozkazów (programy operują jedynie na rejestrach; pamięć jest tylko do odczytu – zakłamy, że dostajemy ją razem z programem), adresowanie jest bajtowe, a adresy mają długość 16 bitów. Każdy rozkaz jest długości jednego bajtu. Wszystkie dane są 16-bitowe. Kodowanie rozkazów jest następujące:

- dwa najbardziej znaczące bity to kod operacji: `add` – 00, `addi` – 01, `mul` – 10, `move` – 11,
- kolejne trzy bity to numer rejestru `rd`,
- trzy najmniej znaczące bity to numer rejestru `rs` lub (trzybitowa) stała w reprezentacji uzupełnień do 2.

Do dyspozycji mamy następujące bloki (patrz również rysunek):

- układ pamięci `MEM`: jedno szesnastobitowe wejście adresowe *Addr*, jedno ośmiobitowe wyjście *MOut*; czas działania: 200 ps,
- plik rejestrów `REG`: dwa trzybitowe wejścia: *RReg* (nr rejestru do odczytu), *WRReg* (nr rejestru do odczytu/zapisu), dwa 16-bitowe wyjścia *ROut*1 – odczytany rejestr *RReg*, *ROut*2 – odczytany rejestr *WRReg*, 16-bitowe wejście *Data* z daną do zapisu, wejście zezwolenia na zapis *Enb*; czas odczytu: 120 ps, czas zapisu: 120 ps,

- jednostka arytmetyczno-logiczna `ALU`: dwa szesnastobitowe wejścia danych $Data1, Data2$, jedno szesnastobitowe wyjście $AOut$, jedno dwubitowe wejście operacji Op (00 – dodaj, 11 – pomnóż, 01 - odejmij); czas działania: 220 ps.
- Szesnastobitowy rejestr licznika rozkazów, `PC`, z wejściem zezwolenia na zapis $WEnb$



Możesz użyć też dodatkowych układów realizujących pewne proste czynności (np. dodawanie stałej do `PC`) – wyspecyfikuj ich działanie. Przydadzą się też na pewno multipleksery, może coś jeszcze... W podpunktach dotyczących wyliczania czasu działania możesz pominąć czas działania wszystkich dodatkowych układów.

(a) Przedstaw schematyczny rysunek procesora w wersji jednocyklowej. Zaznacz dokładnie, gdzie kierowane są poszczególne bity. Opisz precyzyjnie działanie jednostki sterującej (jakie sygnały sterujące są potrzebne, jak są generowane).

(b) Jaka powinna być optymalna długość cyklu zegarowego dla wersji jednocyklowej?

9. W tym zadaniu rozważamy potokową implementację procesora dla architektury opisanej w zadaniu poprzednim.

(a) Ile etapów potoku powinien mieć prosty procesor potokowy? Jakie to etapy? Jaka powinna być optymalna długość cyklu zegarowego?

(b) Z jakiego rodzaju hazardami możemy mieć do czynienia w naszym procesorze potokowym? Podaj przykład kodu powodującego hazardy.

(c) Opisz prosty i efektywny sposób eliminowanania hazardów z poprzedniego podpunktu.