Зертханалық жұмыс №11

Пән: Компьютер архитектурасы және операциялар келісімділігі

Тақырыбы: MIPS ассемблерінде конвейрлеу: нұсқауларды бірнеше кезеңге бөліп орындау

Мақсаты:
- MIPS архитектурасындағы конвейрлеу принципін түсіну;
- Нұсқауларды бірнеше кезеңге бөліп орындау (IF, ID, EX, MEM, WB) сатыларын зерттеу;
- Конвейрлеуде пайда болатын қателерді (hazards) талдау: деректер қақтығысы (data hazard), басқару қақтығысы (control hazard);
- Конвейрді тиімді пайдалану үшін nop-инструкция және қайта жоспарлау (instruction scheduling) тәсілдерін қолдану.

Қысқаша теория
MIPS процессорларының маңызды ерекшеліктерінің бірі — конвейрлеу (pipelining). Бұл әдіс бірнеше нұсқауды бір мезгілде әртүрлі кезеңде орындауға мүмкіндік береді. Жоғары деңгейлі тілдерде біз бір команданы орындап болған соң ғана келесісіне көшеміз. Ал процессорда әр нұсқау бірнеше кезеңнен тұрады.

MIPS архитектурасында нұсқауларды орындау келесі 5 кезеңге бөлінеді:
1. IF (Instruction Fetch) — нұсқауды жадыдан оқу.
2. ID (Instruction Decode) — нұсқауды талдау және регистрлерді оқу.
3. EX (Execute) — арифметикалық/логикалық операцияны орындау немесе адресті есептеу.
4. MEM (Memory) — жадыға қатынау (оқу/жазу).
5. WB (Write Back) — нәтижені регистрге жазу.

Әр нұсқау осы кезеңдерден өтеді. Конвейрдің басты идеясы — бір нұсқау EX кезеңінде тұрғанда, екіншісі ID кезеңінде, үшіншісі IF

кезеңінде орындалуы мүмкін. Осылайша процессор өнімділігі артады.

Мәселелер (Hazards):
- Data hazard — бір нұсқау алдыңғы нұсқаудың нәтижесін күткенде пайда болады.
- Control hazard — шартты көшу нұсқауларында (branch) пайда болады. Процессор қай нұсқауды орындау керегін білмей қалады.
- Structural hazard — аппараттық ресурстар жетпей қалғанда болады.

Шешу тәсілдері:
- nop (no operation) қосу арқылы конвейрді күтуге мәжбүрлеу.
- Instruction scheduling — командаларды орын ауыстырып жазу.
- Forwarding — нәтижені регистрге жазбай тұрып келесі командаға беру.

MIPS конвейрі — компьютер архитектурасын түсінудің маңызды бөлігі. Ол студенттерге төменгі деңгейде параллельділікті үйретеді.

Мысалдар

Мысал 1. nop арқылы қақтығысты шешу

```
.text
main:
   li $t0, 5
   li $t1, 10
   add $t2, $t0, $t1   # t2 = t0 + t1
   nop                 # деректер қақтығысын болдырмау
   sub $t3, $t2, $t0   # t3 = t2 - t0
```

Мысал 2. Instruction scheduling

```
.text
main:
   li $t0, 5
   li $t1, 10
```

```
add $t2, $t0, $t1   # t2 = 15

li $t4, 20          # тәуелсіз команда
sub $t3, $t2, $t0   # t3 = 10
```

Зертханалық жұмыс №11 тапсырмалары

1. Қарапайым бағдарлама жазыңыз: екі санды қосып, нәтижесін пайдаланып, тағы бір арифметикалық амал орындаңыз. Конвейр қақтығысын болдырмау үшін nop қосыңыз.
2. Массивтің үш элементін қосатын бағдарлама құрыңыз. Data hazard пайда болмас үшін instruction scheduling қолданыңыз.
3. Шартты көшу (beq) бар бағдарлама құрып, control hazard мәселесін түсіндіріңіз.
4. lw және sw командаларын қолданып, жадыдан оқу/жазу кезінде болатын hazard-ты көрсетіңіз.
5. Forwarding қолдануға болатын мысал келтіріңіз (кодта комментарий жазыңыз).