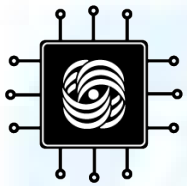
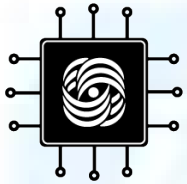


# АРХИТЕКТУРА КОМПЬЮТЕРНЫХ СИСТЕМ

## Лекция 6:

*Уровень микроархитектуры*



# План лекции

- Организация АЛУ
- Основные оптимизации быстродействия
- Примеры микроархитектур

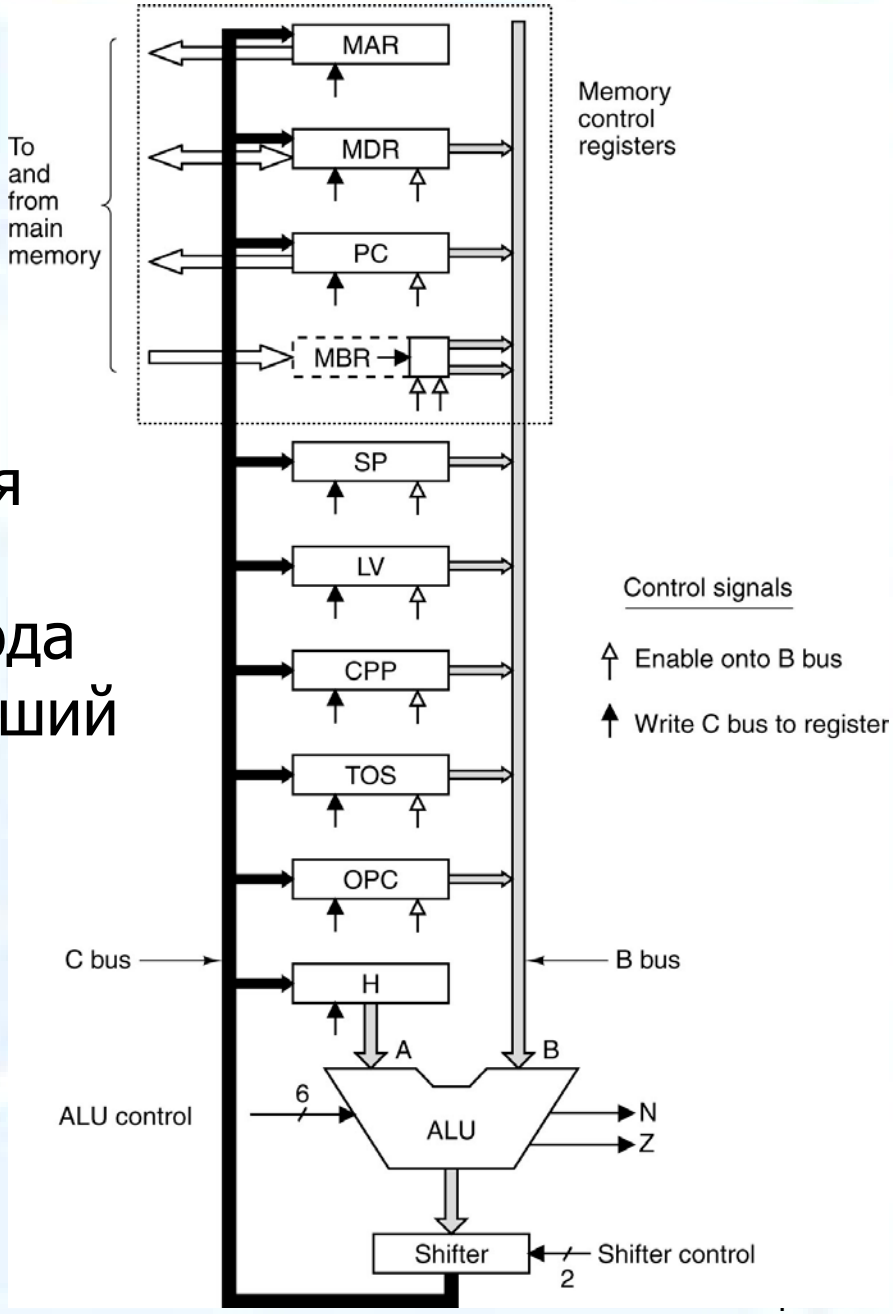
# Уровни архитектуры

- Цифровой логический уровень
- **Уровень микроархитектуры**
- Уровень архитектуры набора команд
- Уровень операционной системы
- Уровень ассемблера

# Тракт данных

**Входные сигналы**

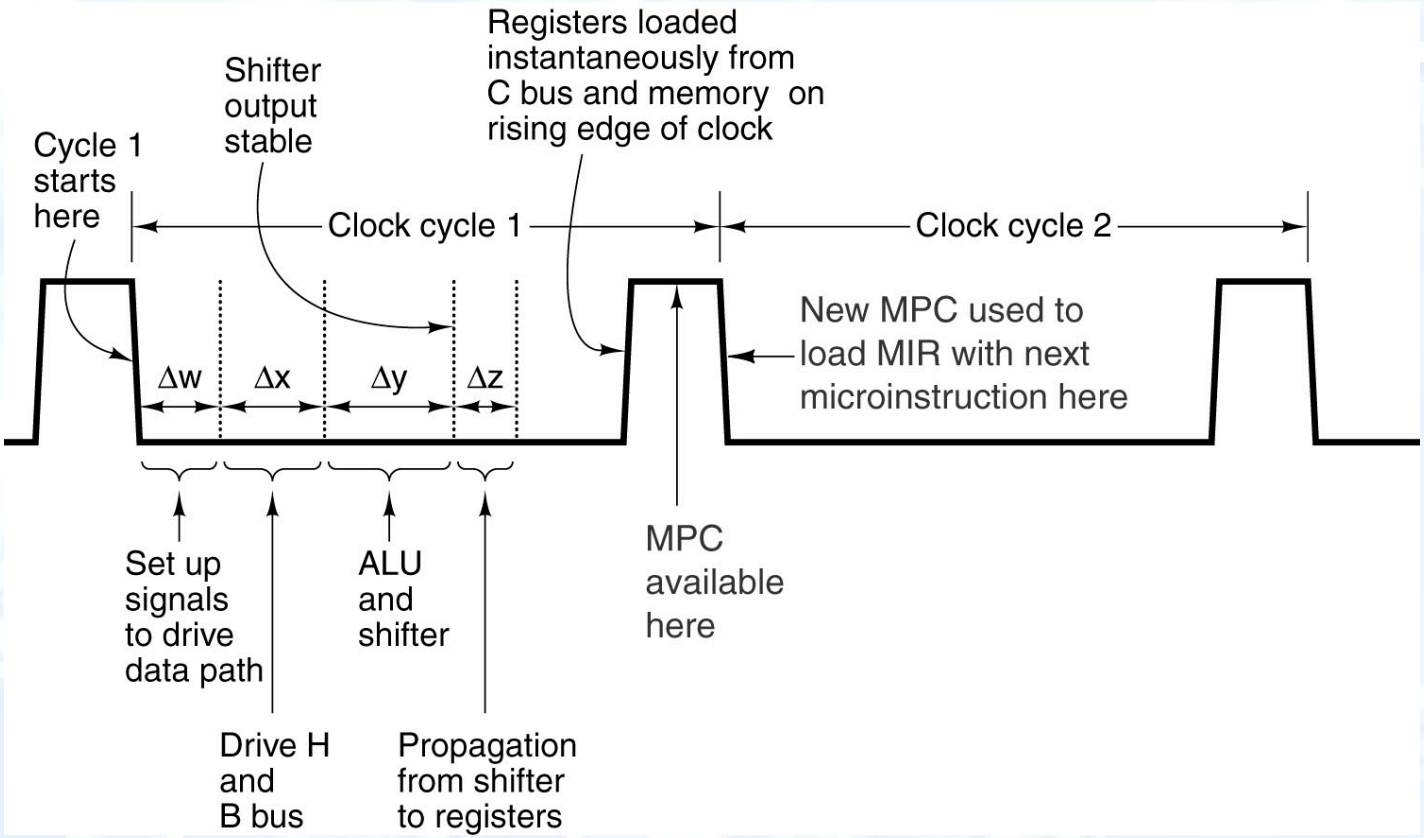
- **F**0 и **F**1 задание операции;
- **ENA** и **ENB** для разрешения входных сигналов
- **INVA –** инверсия левого хода
- **INC –** перенос бита в младший разряд



# Примеры сигналов

| $F_0$ | $F_1$ | ENA | ENB | INVA | INC | Function |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 0 | A |
| 0 | 1 | 0 | 1 | 0 | 0 | B |
| 0 | 1 | 1 | 0 | 1 | 0 | $\overline{A}$ |
| 1 | 0 | 1 | 1 | 0 | 0 | $\overline{B}$ |
| 1 | 1 | 1 | 1 | 0 | 0 | A + B |
| 1 | 1 | 1 | 1 | 0 | 1 | A + B + 1 |
| 1 | 1 | 1 | 0 | 0 | 1 | A + 1 |
| 1 | 1 | 0 | 1 | 0 | 1 | B + 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | B – A |
| 1 | 1 | 0 | 1 | 1 | 0 | B – 1 |
| 1 | 1 | 1 | 0 | 1 | 1 | –A |
| 0 | 0 | 1 | 1 | 0 | 0 | A AND B |
| 0 | 1 | 1 | 1 | 0 | 0 | A OR B |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | –1 |

# Синхронизация тракта данных



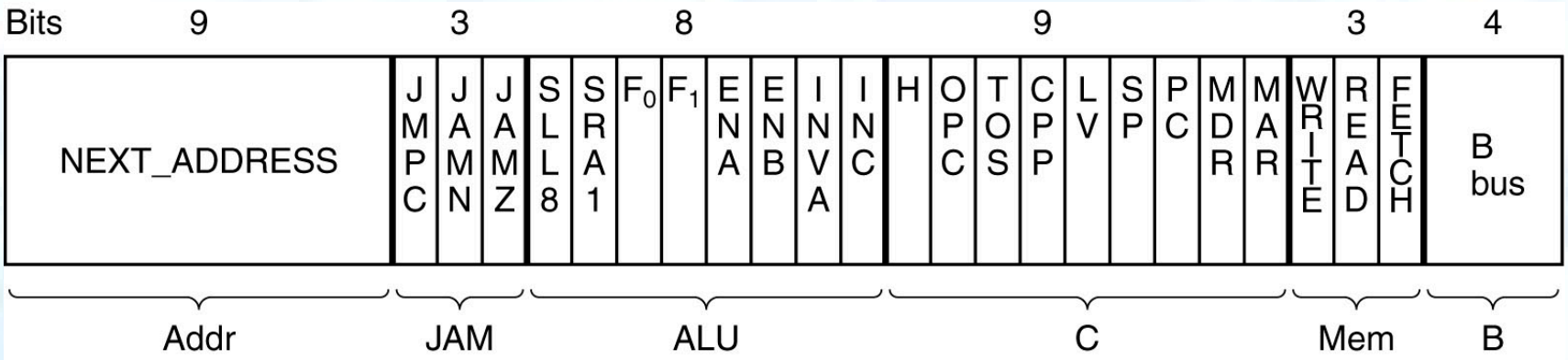
# Сигналы управления

- 9 сигналов для записи данных с шины C в регистры;
- 9 сигналов для разрешения передачи регистров на шину B и в АЛУ;
- 8 сигналов для управления АЛУ и схемой сдвига;
- 2 сигнала, кот. указывают, что нужно осуществить запись или чтение через MAR/MDR
- 1 сигнал, кот. указывает, что нужно осуществить вызов из памяти через регистры PC/MBR

# Цикл тракта данных

- Передача зн-ий рег-ров на шину B
- Прохождение этих сигналов через АЛУ и схему сдвига
- Передача полученных рез-тов на шину C
- Запись их в нужные регистры
- (*) считывание данных из памяти.

# Микрокоманда

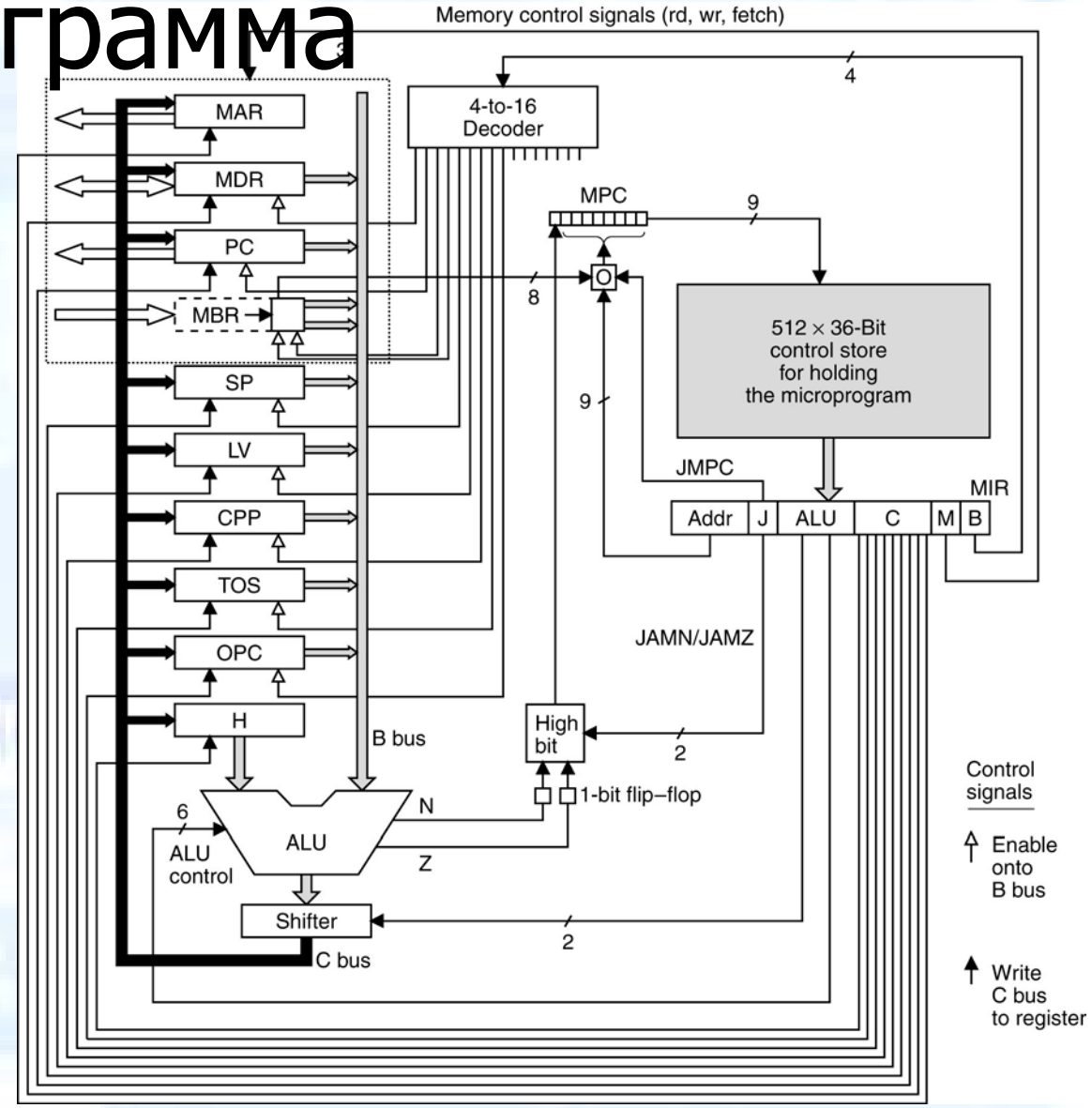



- Addr – адрес след. команды;
- JAM – определение того, как выбирается след. команда;
- ALU – ф-ции АЛУ и схемы сдвига;
- C – выбор регистров, кот. записываются с шины C;
- Mem – ф-ции памяти;
- B – выбор источника для шины B

B bus registers

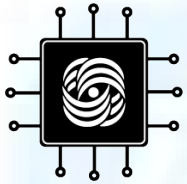
| | |
|---|---|
| 0 = MDR | 5 = LV |
| 1 = PC | 6 = CPP |
| 2 = MBR | 7 = TOS |
| 3 = MBRU | 8 = OPC |
| 4 = SP | 9-15 none |

# Полная диаграмма The Mic-1



# Основные оптимизации (1)

- Снижение количества микрокоманд
- 3-шинная архитектура
- Блок выборки команд
- Упреждающая выборка команд
- Конвейер



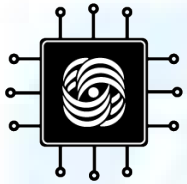
# Основные оптимизации (2)

- Переупорядочивание микрокоманд
- Подмена регистров
- Прогнозирование ветвлений
- Спекулятивное выполнение

# Снижение количества микрокоманд

| Label | Operations | Comments |
|---|---|---|
| pop1 | MAR = SP = SP – 1; rd | Read in next-to-top word on stack |
| pop2 | | Wait for new TOS to be read from memory |
| pop3 | TOS = MDR; goto Main1 | Copy new word to TOS |
| Main1 | PC = PC + 1; fetch; goto (MBR) | MBR holds opcode; get next byte; dispatch |

| Label | Operations | Comments |
|---|---|---|
| pop1 | MAR = SP = SP – 1; rd | Read in next-to-top word on stack |
| Main1.pop | PC = PC + 1; fetch | MBR holds opcode; fetch next byte |
| pop3 | TOS = MDR; goto (MBR) | Copy new word to TOS; dispatch on opcode |

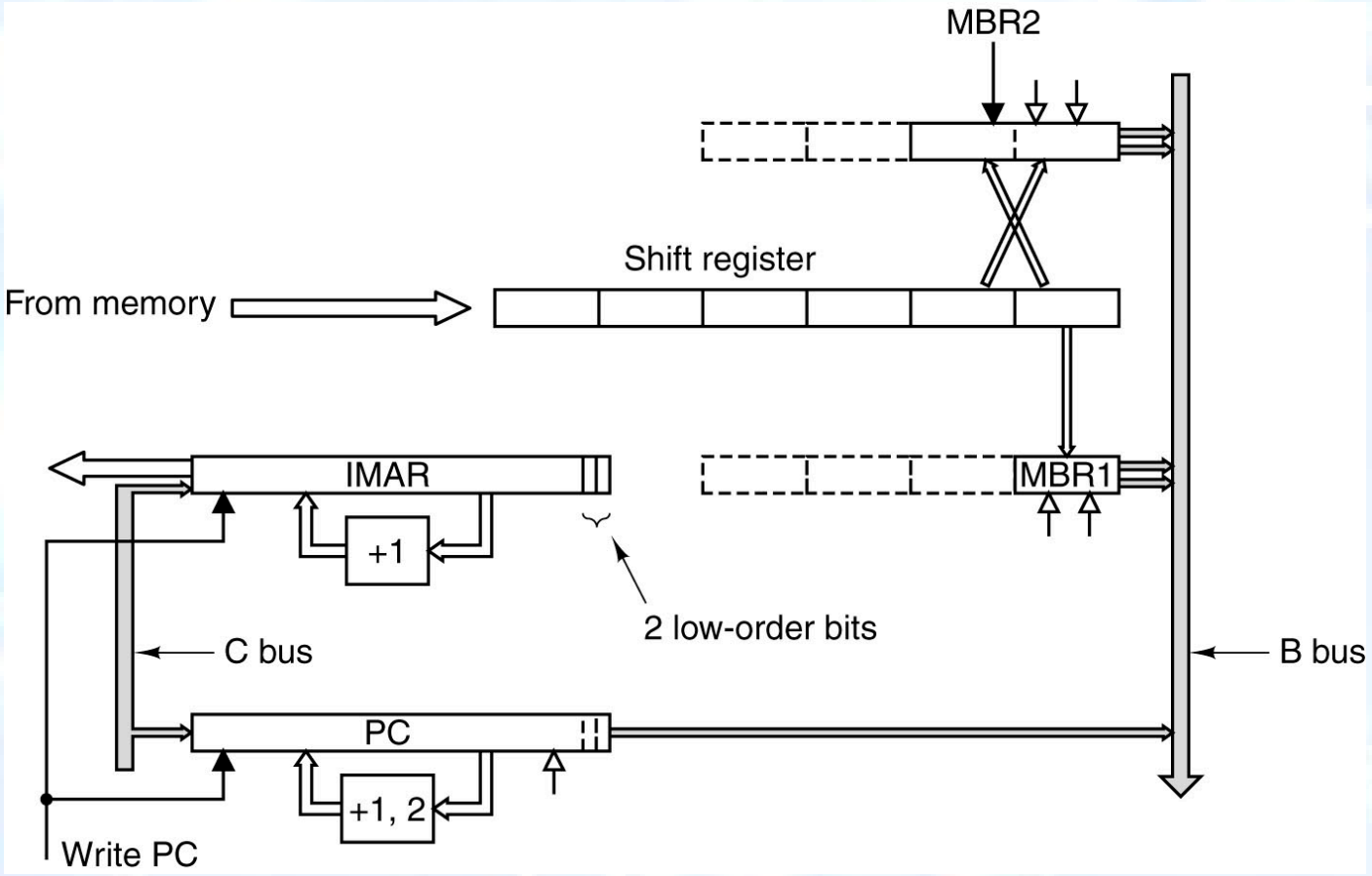


# 3-шинная архитектура

| Label | Operations | Comments |
|---|---|---|
| iload1 | H = LV | MBR contains index; Copy LV to H |
| iload2 | MAR = MBRU + H; rd | MAR = address of local variable to push |
| iload3 | MAR = SP = SP + 1 | SP points to new top of stack; prepare write |
| iload4 | PC = PC + 1; fetch; wr | Inc PC; get next opcode; write top of stack |
| iload5 | TOS = MDR; goto Main1 | Update TOS |
| Main1 | PC = PC + 1; fetch; goto (MBR) | MBR holds opcode; get next byte; dispatch |

| Label | Operations | Comments |
|---|---|---|
| iload1 | MAR = MBRU + LV; rd | MAR = address of local variable to push |
| iload2 | MAR = SP = SP + 1 | SP points to new top of stack; prepare write |
| iload3 | PC = PC + 1; fetch; wr | Inc PC; get next opcode; write top of stack |
| iload4 | TOS = MDR | Update TOS |
| iload5 | PC = PC + 1; fetch; goto (MBR) | MBR already holds opcode; fetch index byte |

# Блок выборки команд



# Проблемы конвейера

- RAW- взаимосвязи
- WAR- взаимосвязи
- WAW- взаимосвязи

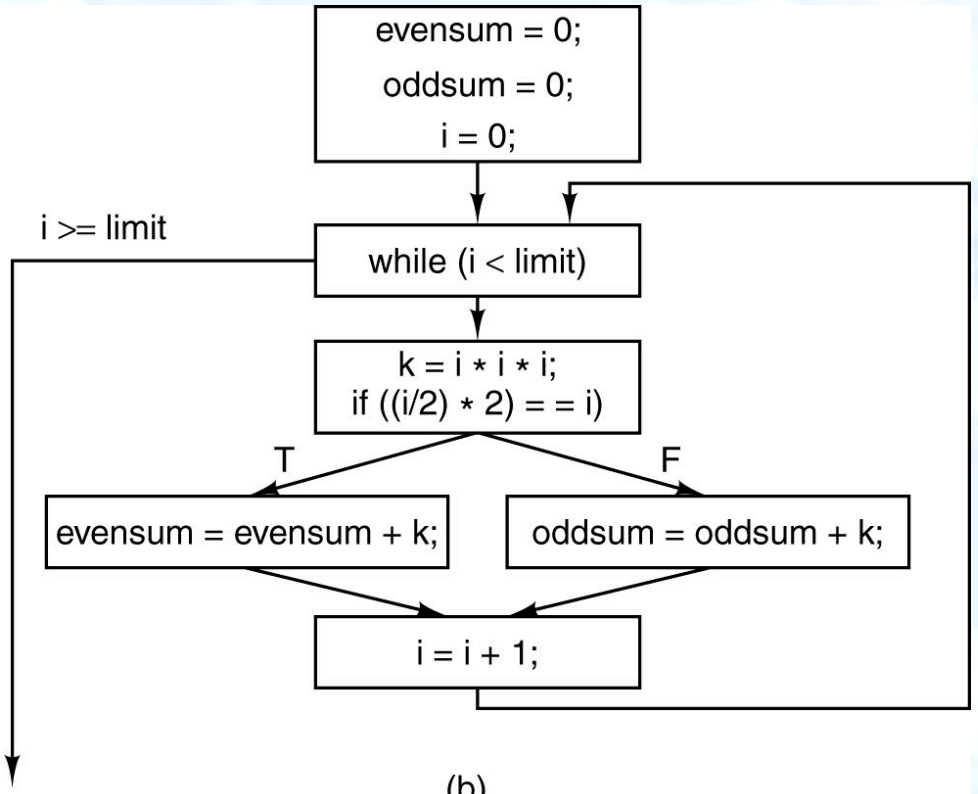
# Переименование регистров и переупорядочивание микрокоманд

| | | | | | Registers being read | | | | | | | | Registers being written | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Cy | # | Decoded | Iss | Ret | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | 1 | R3=R0*R1 | 1 | | 1 | 1 | | | | | | | | | | 1 | | | | |
| | 2 | R4=R0+R2 | 2 | | 2 | 1 | 1 | | | | | | | | | 1 | 1 | | | |
| 2 | 3 | R5=R0+R1 | 3 | | 3 | 2 | 1 | | | | | | | | | 1 | 1 | 1 | | |
| | 4 | R6=R1+R4 | – | | 3 | 2 | 1 | | | | | | | | | 1 | 1 | 1 | | |
| 3 | 5 | R7=R1*R2 | 5 | | 3 | 3 | 2 | | | | | | | | | 1 | 1 | 1 | | 1 |
| | 6 | S1=R0–R2 | 6 | | 4 | 3 | 3 | | | | | | | | | 1 | 1 | 1 | | 1 |
| | | | | 2 | 3 | 3 | 2 | | | | | | | | | 1 | | 1 | | 1 |
| 4 | | | 4 | | 3 | 4 | 2 | | 1 | | | | | | | 1 | | 1 | 1 | 1 |
| | 7 | R3=R3*S1 | – | | 3 | 4 | 2 | | 1 | | | | | | | 1 | | 1 | 1 | 1 |
| | 8 | S2=R4+R4 | 8 | | 3 | 4 | 2 | | 3 | | | | | | | 1 | | 1 | 1 | 1 |
| | | | | 1 | 2 | 3 | 2 | | 3 | | | | | | | | | 1 | 1 | 1 |
| | | | | 3 | 1 | 2 | 2 | | 3 | | | | | | | | | | 1 | 1 |
| 5 | | | | 6 | | 2 | 1 | | 3 | | | | | 1 | | | | | 1 | 1 |
| 6 | | | 7 | | | 2 | 1 | 1 | 3 | | | | | 1 | | 1 | | | 1 | 1 |
| | | | | 4 | | 1 | 1 | 1 | 2 | | | | | 1 | | 1 | | | | 1 |
| | | | | 5 | | | | 1 | 2 | | | | | 1 | | 1 | | | | |
| | | | | 8 | | | | 1 | | | | | | | | 1 | | | | |
| 7 | | | | | | | | 1 | | | | | | | | 1 | | | | |
| 8 | | | | | | | | 1 | | | | | | | | 1 | | | | |
| 9 | | | | 7 | | | | | | | | | | | | | | | | |

# Спеклятивное Выполнение

```
evensum = 0;
oddsum = 0;
i = 0;
while (i < limit) {
    k = i * i * i;
    if (((i/2) * 2) == i)
        evensum = evensum + k;
    else
        oddsum = oddsum + k;
    i = i + 1;
}
```

(a)

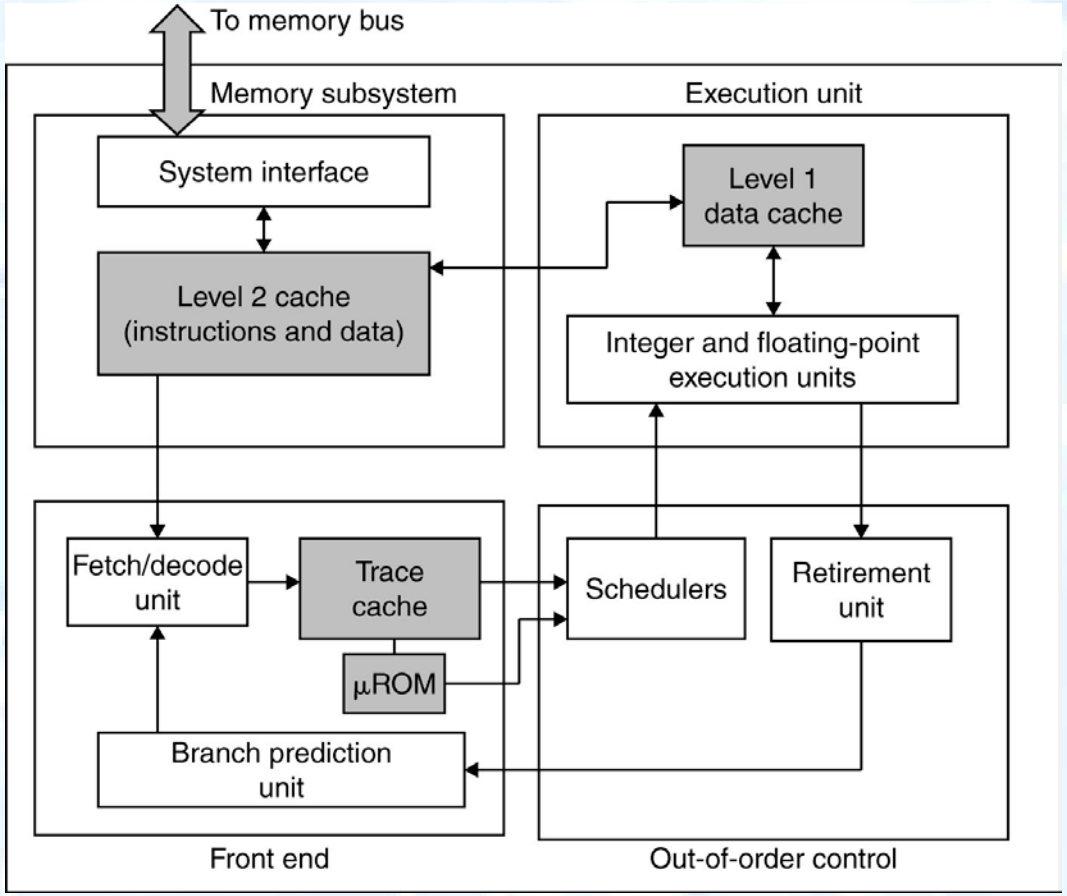



(b)

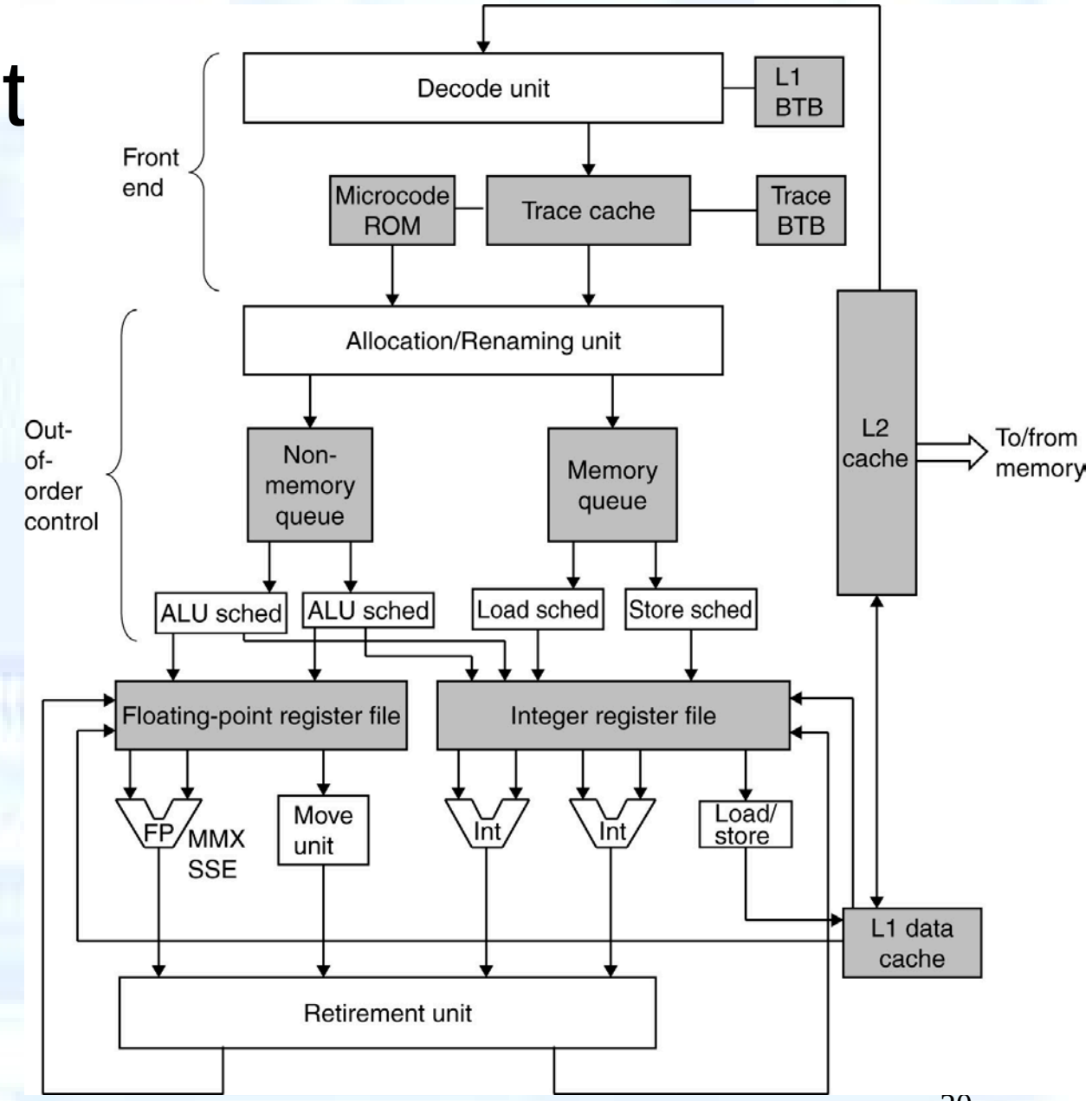
(a) Фрагмент программы.

(b) Блок схема.

# NetBurst Микроархитектура



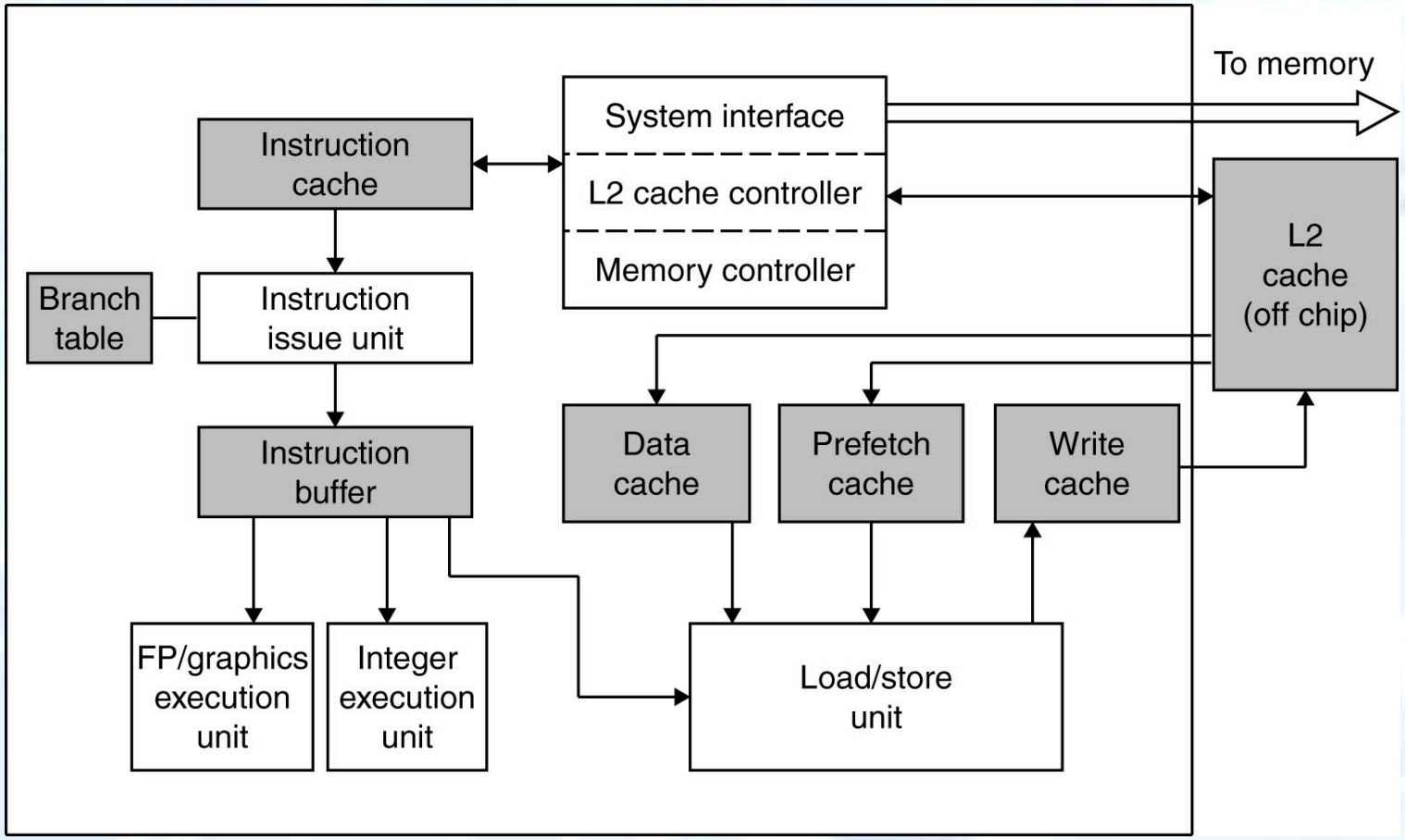
# The NetBurst Конвейер



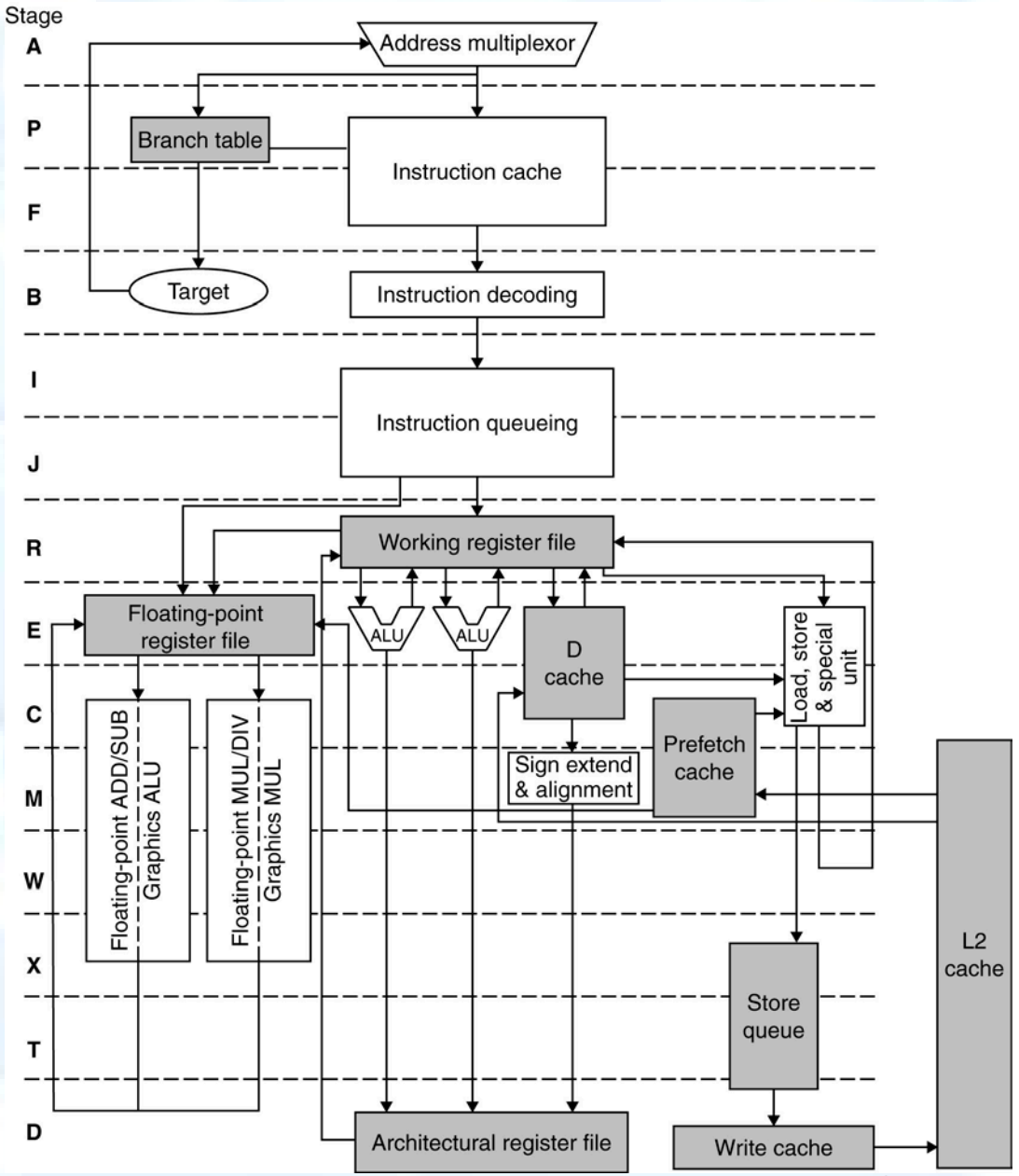
# Планировщики АЛУ

- АЛУ1 и блок смещения оп-ций с пл. точкой
- АЛУ2 и блок исполнения оп-ций с пл. точкой
- Команды загрузки
- Команды сохранения

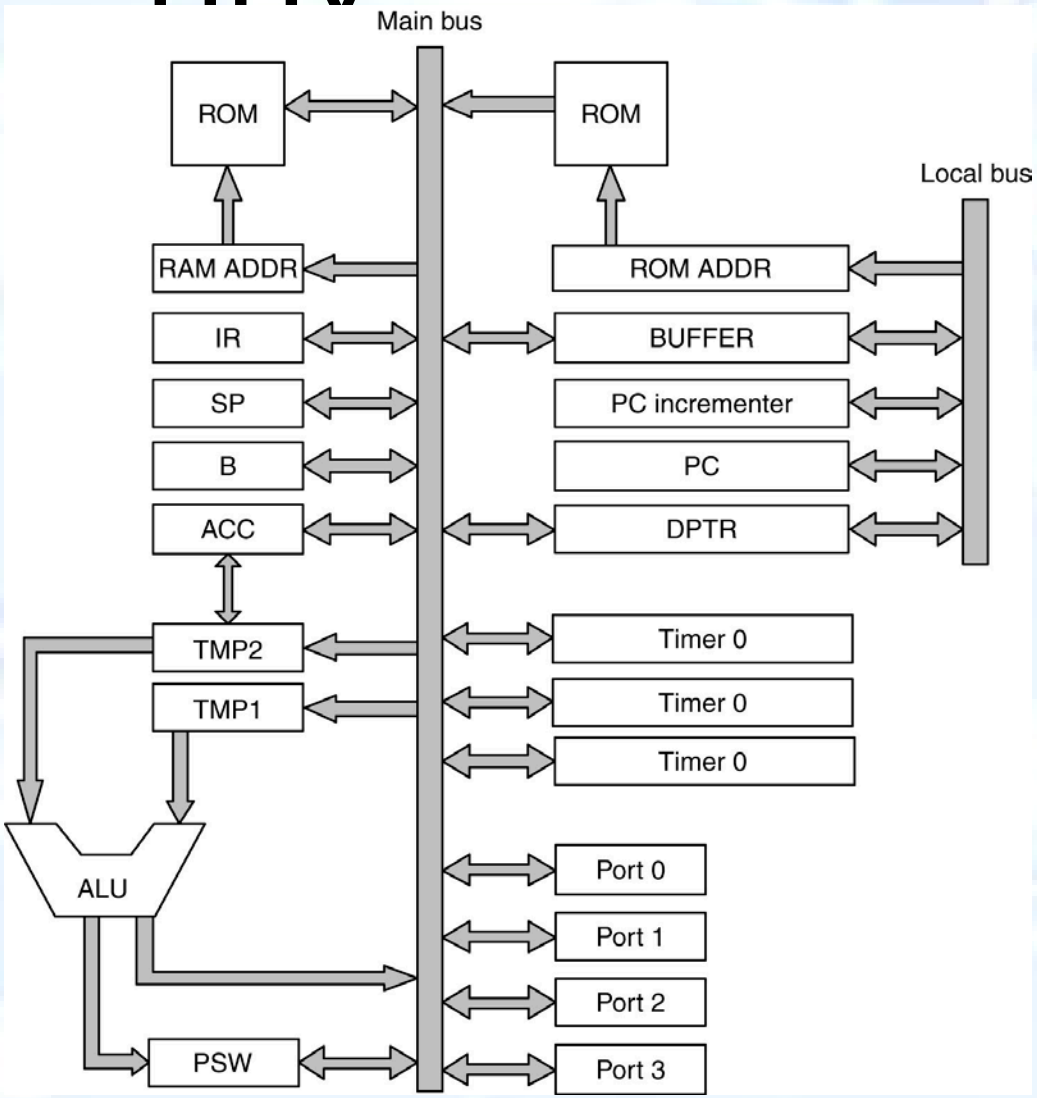
# Микроархитектура UltraSPARC III Cu

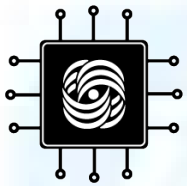


# UltraSPARC III Cu конвейер



# Микроархитектура 8051 ПРУ





# Спасибо за внимание!