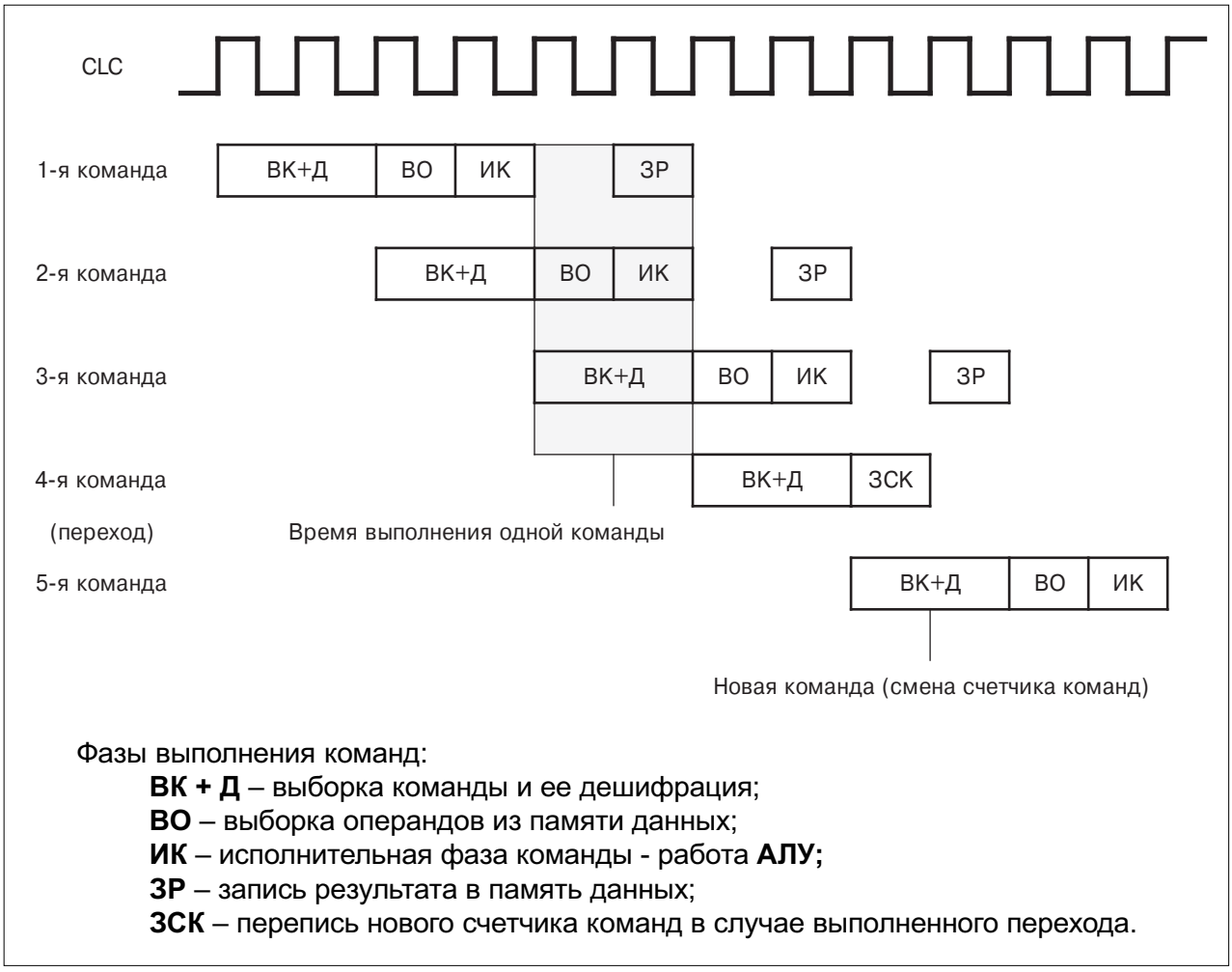
ОРГАНИЗАЦИЯ ВЫЧИСЛИТЕЛЬНОГО ПРОЦЕССА

Для повышения производительности процессора производится совмещение в потоке команд отдельных операций, выполняемых внутренними устройствами микроконтроллера. На рис. 7 показан принцип совмещения по времени различных фаз выполнения команды на примере трех арифметических команд и одной команды перехода.



Фазы выполнения команд:

ВК + Д – выборка команды и ее дешифрация;
ВО – выборка операндов из памяти данных;
ИК – исполнительная фаза команды - работа **АЛУ;**
ЗР – запись результата в память данных;
ЗСК – перепись нового счетчика команд в случае выполненного перехода.

Рис. 7. Временные диаграммы выполнения команд и совмещение по времени различных фаз выполнения команды

Выборка операндов для текущей команды и ее исполнение совмещаются с выборкой и дешифрацией последующей команды. Запись результата в память данных отложена на один такт для обеспечения нормальной загрузки информационных трактов памяти данных. Когда результат операции используется в качестве операнда для последующей команды, операнд берется из регистра результата на выходе **АЛУ**.

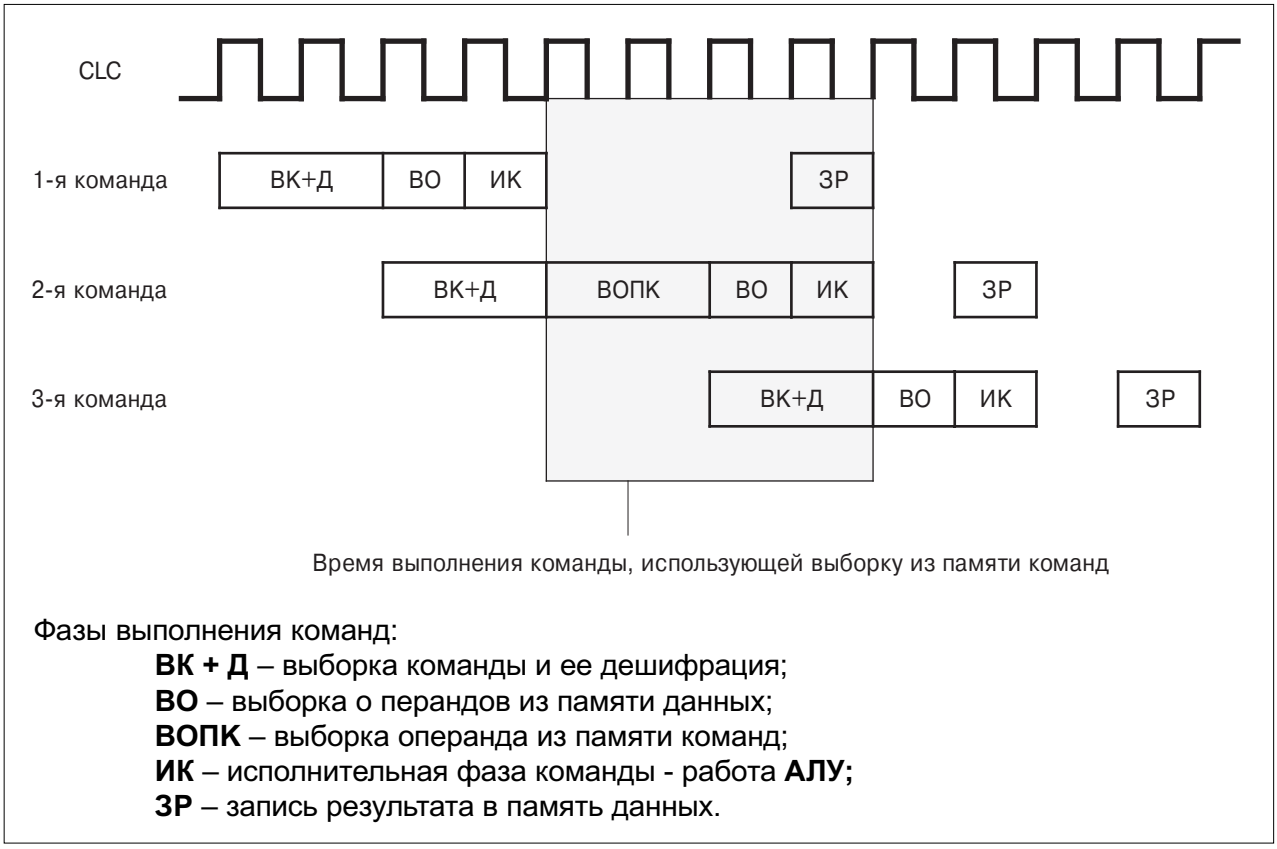
Таким образом, происходит совмещение по времени выборки и дешифрации (**n+1**)-й команды с выборкой операндов и исполнительной фазой **n**-й команды и записью результата (**n-1**)-й команды.

При обработке ситуаций, когда изменяется счетчик команд, после дешифрации команды перехода и соблюдения условия перехода происходит перепись нового счетчика команд из регистра команды в счетчик команд.

Обращения к операндам, размещенным в оперативной памяти и регистрах периферийных устройств, происходят единообразно по времени. Особым случаем является ситуация, когда

операнд размещен в памяти команд. Для этого необходимо установить регистр косвенной адресации **IR1** в режим адресации к памяти команд. Использовать **IR1** в этом режиме можно только для чтения операнда, запись в **IR1** не будет иметь смысла. Для выборки операнда из памяти команд приходится прерывать процесс выборки и дешифрации команд и производить эту выборку из блока памяти команд, что приводит к дополнительной потере времени.

На рис. 8 приведены временные диаграммы процесса выполнения трех команд, две из которых: 1-я и 3-я, используют операнды из памяти данных, а 2-я – операнд из памяти команд. Из временных диаграмм видно, что обращение к памяти команд приводит к задержке при выполнении 2-й команды на два такта тактовой частоты процессора.



Фазы выполнения команд:

ВК + Д – выборка команды и ее дешифрация;
ВО – выборка о перандов из памяти данных;
ВОПК – выборка операнда из памяти команд;
ИК – исполнительная фаза команды - работа **АЛУ;**
ЗР – запись результата в память данных.

Рис. 8 . Временные диаграммы выполнения команды,

использующей выборку операнда из памяти команд

Особенности совмещения по времени выполнения команд накладывают определенные ограничения на работу с регистрами внутренних периферийных устройств.

Если операнд **dst n-**й команды выступает в качестве операнда для чтения **(n + 1)-**й команды, то в силу того, что запись результата в память еще не произошла, операнд берется не из памяти, а из регистра результата команды на выходе **АЛУ**. Для оперативной памяти это не существенно, так как запись в нее произойдет в следующем такте. Но если в качестве этого операнда выступает регистр периферийного устройства, то отсутствие реального чтения этого регистра может привести к неприятным последствиям. Поэтому, желательно в этом случае вставлять между **n**-й и **(n + 1)**-й командами команду **NOP.**

Проиллюстрируем эту ситуацию примером. Выполняется отрезок программы, в котором производится взаимодействие с портом ввода / вывода **Б**, младшие четыре разряда которого сконфигурированы как входы, а старшие четыре разряда – как выходы. Вначале необходимо сформировать высокий уровень сигнала на линии **PB7**, а затем проконтролировать линию **PB2** на наличие на ней высокого уровня сигнала.

```
        BISH %a2,#8     ; PB7 = 1, подача высокого уровня на линию PB7
        BTTL %a2,#4     ; проверка линии PB2
        JNZ  high       ; если на PB2 высокий уровень, идти на участок программы high
low:    MOV  %b5,%a2    ; если на PB2 низкий уровень, переслать содержимое порта в память
```

В данном примере при выполнении команды **BTTL** вместо рабочего регистра порта **B** будет читаться результат предыдущей команды **BISH**, у которого младшие четыре разряда имеют всегда нулевое значение. Команда проверки разрядов не работает. Данный отрезок программы необходимо переписать следующим образом:

```
        BISH %a2,#8     ; PB7 = 1, подача высокого уровня на линию PB7
        NOP             ; пропуск 2 тактов
        BTTL %a2,#4     ; проверка линии PB2
        JNZ  high       ; если на PB2 высокий уровень, идти на участок программы high
low:    MOV  %b5,%a2    ; если на PB2 низкий уровень, переслать содержимое порта в память
```

Теперь при выполнении команды **BTTL** будет читаться рабочий регистр порта **B.**

При программировании периферийных устройств следует также учесть, что все команды, кроме команд **MOV** и **MOVL**, работают с операндом **dst** через чтение – модификацию – запись. То есть операнд **dst** считывается, над ним проводится преобразование, а затем результат преобразования записывается в память. Если все разряды регистра периферийного устройства доступны по чтению / записи, проблем не возникает. Если же отдельные разряды доступны только по записи, то установка их будет производиться только командами **MOV , MOVL** или **BIS** с литерой, непосредственно в этом разряде.

ПРЕРЫВАНИЯ ПРОГРАММЫ

Центральный процессор микроконтроллера **КР1878ВЕ1** обрабатывает прерывания от периферийных внутренних устройств. Каждое прерывание имеет свой приоритет и свой вектор программы обработки прерывания. Векторы прерывания – фиксированные адреса в памяти команд, на которые передается управление во время обслуживания прерывания. Для каждого прерывания отведен свой вектор. В случае одновременного возникновения прерываний первым будет обслуживаться то прерывание, вектор которого меньше. Вектора прерываний приведены в табл. 9.

Таблица 9

Векторы прерываний в микроконтроллере

Вектор прерывания	Источник прерывания	Состояние разряда IE
0	Начальный пуск	-
1	Сторожевой таймер	-
2	Переполнение стека команд или данных	-
3	Интервальный таймер	IE = 1
4, 5	Резерв	-
6	Порт A	IE = 1
7	Порт B	IE = 1
$8 \div E_{16}$	Резерв	-
F_{16}	Завершение записи в блок ЭСППЗУ	IE = 1

Прерывания по начальному пуску, от сторожевого таймера и по ошибке работы со стеками

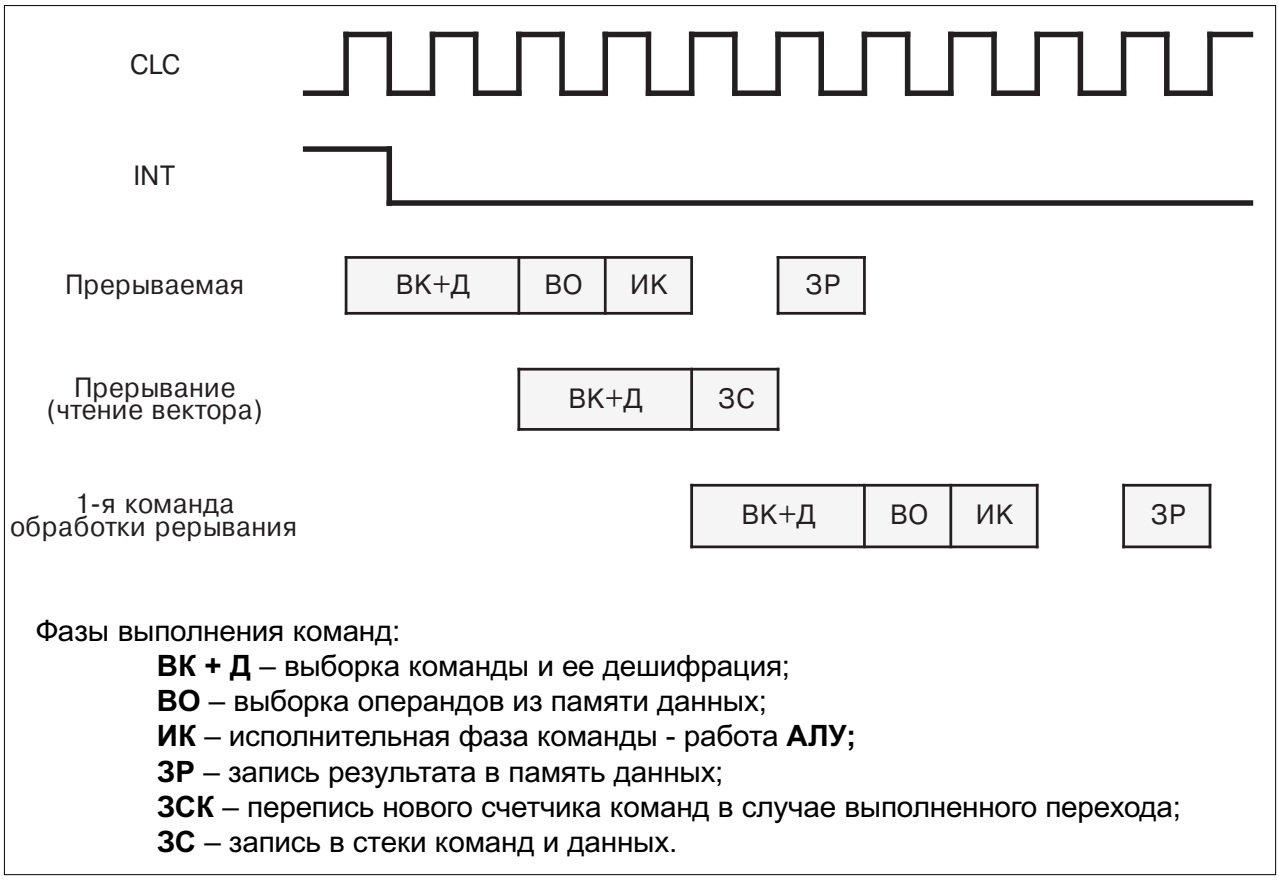
происходят вне зависимости от состояния разряда разрешения прерывания в регистре состояния процессора. Для всех остальных прерываний необходимо, чтобы этот разряд был установлен.

Механизм прерываний осуществляется единым образом и происходит в следующей последовательности:

- В регистре состояния процессора должен быть взведен разряд разрешения прерывания **IE** . Прерывания по начальному пуску, от сторожевого таймера и при переполнении стеков будут возникать независимо от значения разряда **IE** регистра состояния.
- По возникновении причины прерывания вырабатывается радиальный сигнал, подсвечивающий факт возникновения прерывания. Центральный процессор реагирует на фронт сигнала прерывания во время второй четверти фазы выборки и дешифрации команды.
- Центральный процессор, отреагировав на прерывание, отменяет выполнение команды, выборка которой была уже осуществлена, и приступает к обработке прерывания.
- В стек команд переписывается значение счетчика команд, указывающее на текущую команду. Указатель стека команд **ISP** переводится на одну позицию в сторону увеличения адреса. При прерывании по начальному пуску записи в стек команд происходить не будет.
- В стек данных переписывается значение регистра состояния процессора на момент выполнения текущей команды. Указатель стека данных **DSP** переводится на одну позицию в сторону увеличения адреса. При прерывании по начальному пуску записи в стек данных происходить не будет.
- В счетчик команд загружается значение, соответствующее вектору данного прерывания. Целесообразно в память команд по адресу вектора прерываний записывать команду безусловного перехода на программу обработки данного прерывания.
- Выполняется команда, записанная по адресу вектора прерывания. Если это был безусловный переход, выполняется переход на программу обработки прерывания.
- В программе обработки прерывания следует позаботиться о снятии сигнала прерывания. В случае обработки прерывания по переполнению стеков рекомендуется подать команду **RESET**. Для снятия прерывания от других устройств необходимо осуществить действия, предназначенные для снятия сигнала прерывания в данном устройстве.
- Если есть необходимость сохранения контекста, следует использовать команду **PUSH** для сохранения в стеке данных помимо регистра состояния значений необходимых служебных регистров.

Временные диаграммы процедуры прерывания приведены на рис. 9. Из рисунка видно, что реакция на прерывание составляет три такта тактовой частоты процессора.

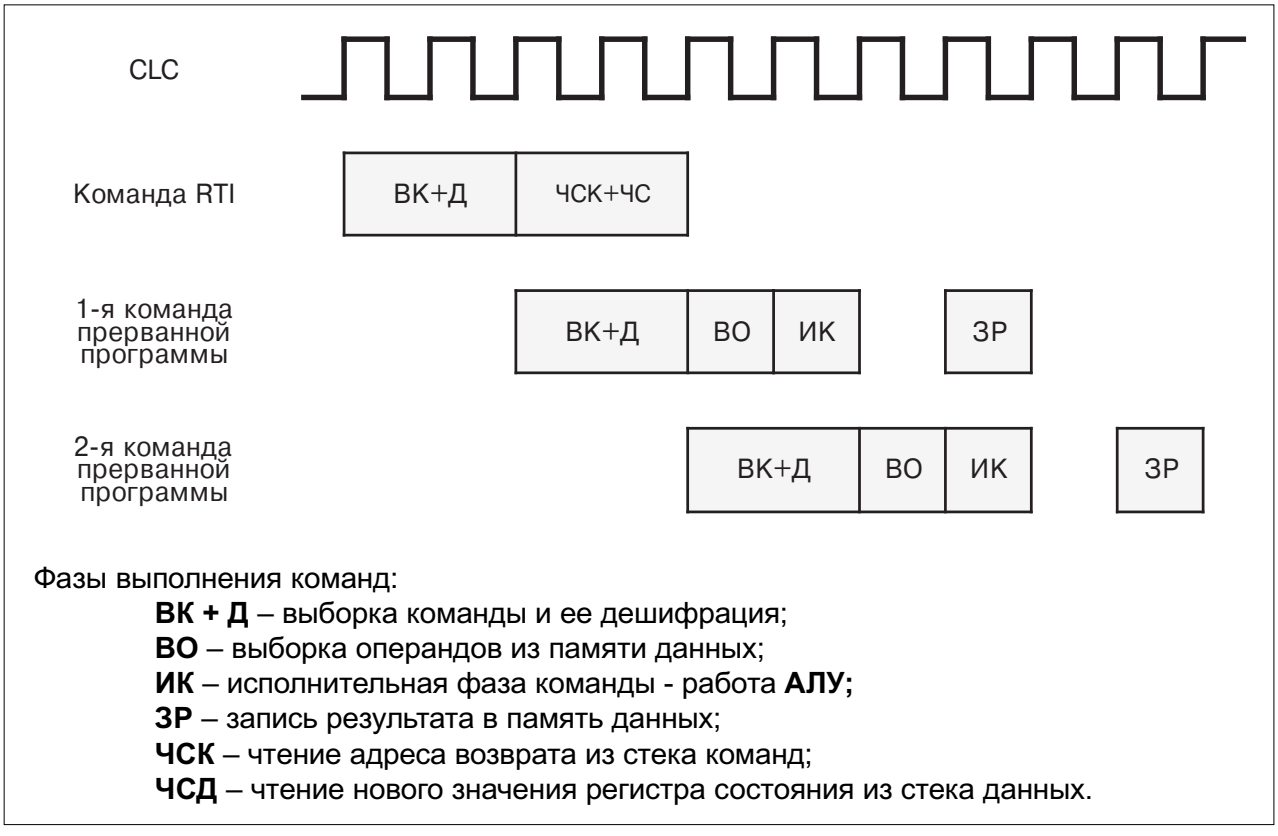
Возврат из прерывания осуществляется по команде **RTI**. По команде возврата из прерывания происходит восстановление из стека данных значения регистра состояний и из стека команд сохраненного счетчика команд. Процессор переходит к выполнению программы, которая была перервана по приходу сигнала прерывания. Временные диаграммы выполнения команды возврата из прерывания приведены на рис. 10.



Фазы выполнения команд:

ВК + Д – выборка команды и ее дешифрация;
ВО – выборка операндов из памяти данных;
ИК – исполнительная фаза команды - работа **АЛУ;**
ЗР – запись результата в память данных;
ЗСК – перепись нового счетчика команд в случае выполненного перехода;
ЗС – запись в стеки команд и данных.

Рис. 9. Диаграммы процедуры прерывания команд



Фазы выполнения команд:

ВК + Д – выборка команды и ее дешифрация;
ВО – выборка операндов из памяти данных;
ИК – исполнительная фаза команды - работа **АЛУ;**
ЗР – запись результата в память данных;
ЧСК – чтение адреса возврата из стека команд;
ЧСД – чтение нового значения регистра состояния из стека данных.

Рис. 10. Диаграммы выполнения команды возврата из прерывания