

В качестве инструментального компьютера для ПК может быть использована ЭВМ СМ-1800, выполненная на том же микропроцессоре КР580ИК80А. Могут быть использованы и ЭВМ с другой системой команд, например СМ-4. Эта машина должна быть оснащена специальными программами — кроссобеcпечением, позволяющим транслировать программы в машинные коды микропроцессора КР580ИК80А.

Чтобы наглядно представить выигрыш, который дают инструментальные машины, отметим, что компилятор с языка Паскаль для ЭВМ СМ-1800 занимает на гибком магнитном диске более 200 Кбайт и требует для своей работы не менее 48 Кбайт ОЗУ машины, куда отдельные части компилятора загружаются по мере надобности. Таким образом, работать на языке Паскаль на ПК без инструментальной ЭВМ практически нельзя.

Полученные на инструментальных ЭВМ программы в машинных кодах могут работать на таких простых компьютерах, как, например, ПК или какие-либо микропроцессорные устройства управления, память которых рассчитана на выполнение только одной программы.

Несмотря на повсеместное распространение языков высокого уровня, по занимаемому объему памяти и скорости работы во многих случаях более выгодны программы, написанные на языке Ассемблера. Это машинно-ориентированный язык, программирование на нем, по сути, не отличается от программирования в машинных кодах. Но при этом язык Ассемблера позволяет работать программисту, используя мнемонику команд, символьные переменные и автоматическое распределение памяти, избавляет его от ручной трансляции исходных текстов программ в машинные коды.

С чем же связано то, что ассемблерные программы более эффективны? Поясним это на примере. Компилятор, встретив в исходном тексте программы какую-либо арифметическую операцию над целыми числами, включает в состав объектного (машинного) кода соответствующие программы для выполнения арифметических операций. Но предположим, что эти программы могут выполнять действия не только с целыми числами. Следовательно, в объектном коде появляются «лишние», неиспользуемые коды. Эта избыточность приводит к нерациональному увеличению объема программы после трансляции, увеличивается время ее выполнения. А программы на языке Ассемблера в принципе могут быть свободны от такой избыточности, так как сам язык очень близок к машинным кодам.

## ПРИЛОЖЕНИЕ

### 1. Микропроцессор КР580ИК80А

Микропроцессор КР580ИК80А конструктивно выполнен в пластмассовом корпусе с 40 выводами. Для микропроцессора необходимы три источника напряжения питания: +5, —5 и +12 В.

Выводы микропроцессора и назначение передаваемых через них сигналов:

$\Phi 1$ ,  $\Phi 2$  — входные периодические сигналы для тактирования микропроцессора, формируемые внешним генератором. Они должны иметь амплитуду +12 В,

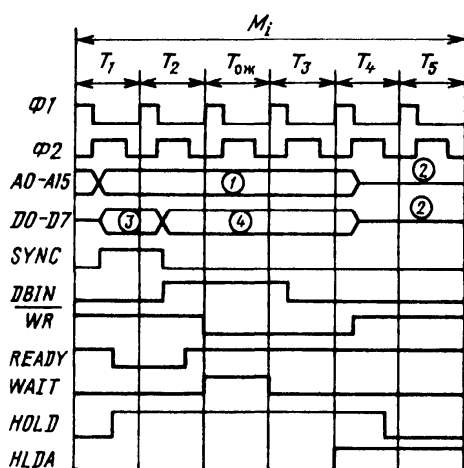


Рис. П.1. Временная диаграмма машинного цикла микропроцессора: 1 — адрес; 2 — высокий импеданс; 3 — состояние микропроцессора; 4 — данные

в то время как все другие сигналы микропроцессора имеют ТТЛ-уровни. Тактирующие импульсы определяют моменты всех действий микропроцессора. Выполнение каждой команды происходит за один или несколько (до 5) машинных циклов, в каждом из которых происходит обмен одним байтом с памятью или портами ввода-вывода. В свою очередь, машинный цикл разделяется на 3...5 тактов, длительность которых равна периоду следования тактирующих импульсов. Временная диаграмма выполнения машинного цикла приведена на рис. П. 1. В первом машинном цикле  $M_1$  любой команды микропроцессор в такте  $T_3$  считывает из памяти микроЭВМ код операции команды и, если не требуются дополни-

тельные действия по обмену данными с памятью или портами ввода-вывода, заканчивает ее выполнение в такте  $T_4$  (или  $T_4$  и  $T_5$ ). В противном случае для выполнения команды требуются дополнительные машинные циклы  $M_2$ — $M_5$ .

$A_0$ — $A_{15}$  — выходные сигналы шины адреса для адресации памяти и портов ввода-вывода. Причем при адресации портов ввода-вывода информация, передаваемая по линиям  $A_0$ — $A_7$  шины адресов, дублируется на линиях  $A_8$ — $A_{15}$ . Это позволяет более равномерно распределить нагрузку на шину адреса при использовании в микроЭВМ большого количества портов ввода-вывода. Код адреса появляется на шине адреса в такте  $T_1$  с задержкой 200...270 нс относительно фронта импульса  $\Phi_2$  и остается стабильным до момента возникновения фронта импульса  $\Phi_2$  в такте, следующем за тактом  $T_3$  текущего цикла. Линии шины адреса могут принимать высокоимпедансное состояние.

$D_0$ — $D_7$  — двунаправленные линии шины данных. Являются входными при вводе данных из памяти или портов ввода-вывода, выходными — при выводе данных из микропроцессора. Используются также для вывода в такте  $T_1$  по фронту  $\Phi_2$  дополнительной информации о характере операций обмена в текущем машинном цикле — байте состояния. Эта информация снимается с линий шины данных через 220...280 нс после прихода импульса  $\Phi_2$  в такте  $T_2$ . Линии шины данных могут принимать высокоимпедансное состояние.

$DBIN$  — выходной сигнал, инициирующий выдачу данных из ячейки памяти или порта на шину данных. Во время действия этого сигнала микропроцессор считывает байт с шины данных. Сигнал генерируется с задержкой 130...200 нс относительно фронта  $\Phi_2$  в такте  $T_2$  машинных циклов считывания данных в микропроцессор. Байт данных должен быть установлен на шине данных не менее чем за 30 нс до появления заднего фронта  $\Phi_1$  и оставаться стабильным не менее 130 нс после появления фронта  $\Phi_2$  такта  $T_3$ . Сигнал  $DBIN$  снимается по положительному фронту  $\Phi_2$  такта  $T_3$  с максимальной задержкой 200 нс.

$WR$  — выходной сигнал, сопровождающий выдачу микропроцессором байта данных на шину данных для записи в ячейку памяти или порт. Формируется в

тактах T3 или T<sub>ож</sub> (такт ожидания) с задержкой 70...120 нс относительно положительного фронта сигнала Ф1. Байт данных микропроцессор формирует на шине данных в такте T2 с задержкой 220...280 нс относительно фронта Ф2. Сигнал  $\overline{WR}$  снимается в такте, следующем за тактом T3 с задержкой около 100 нс относительно фронта Ф1. Следует заметить, что на рис. П.1 сигналы DBIN и  $\overline{WR}$  изображены активными условно. В течение реального машинного цикла активен только один из них. Эти сигналы являются общими как для модулей памяти, так и для портов ввода-вывода, что приемлемо только в простейших микроЭВМ. В более сложных микроЭВМ используют дополнительные сигналы управления, выдаваемые микропроцессором на шину данных в тактах T1/T2 в виде 8-разрядного кода — байта состояния, который запоминается (фиксируется) во внешнем регистре и определяет действие микропроцессора в следующих тактах текущего машинного цикла. Наличие единицы в разрядах D0, D2—D7 или нуля в разряде D1 байта состояния является признаком выполнения микропроцессором в текущем машинном цикле следующих действий:

- D0(INTA) — обслуживание запроса прерывания;
- D1(WO) — запись в память или вывод данных в порт;
- D2(STACK) — обращение к области памяти, используемой в качестве стека;
- D3(HLT) — останов микропроцессора по команде HLT;
- D4(OUT) — вывод данных в порт;
- D5(MI) — чтение кода операции команды;
- D6(INP) — ввод данных из порта;
- D7(MEMR) — чтение данных из памяти.

SYNC — выходной сигнал синхронизации, вырабатываемый по фронту Ф2 в такте T1 каждого машинного цикла и указывающий, что по шине данных передается байт состояния микропроцессора. Сигнал снимается через 130...200 нс после появления фронта Ф2 в такте T2. По сигналу синхронизации байт состояния фиксируется во внешнем регистре. Таким образом, через шину данных из микропроцессора поступают не только данные, а в отдельные моменты и управляющие сигналы. Такое использование шин для передачи в разное время различных сигналов называют временным мультиплексированием. Оно необходимо из-за ограниченного числа выводов микропроцессора. Увеличивать же число выводов у микропроцессора нецелесообразно, так как это повышает его стоимость и снижает надежность.

READY — входной сигнал от модулей памяти или портов ввода-вывода, указывающий на их готовность к обмену данными с микропроцессором. Если сигнал на этом входе имеет низкий уровень, то после такта T2 микропроцессором выполняются вспомогательные такты ожидания T<sub>ож</sub> (т. е. он переходит в состояние ожидания). Это позволяет использовать в микроЭВМ память и периферийное оборудование с малым быстродействием. Число вспомогательных тактов ожидания определяется длительностью поддержания входного сигнала на выводе READY в состоянии 0. Как только память или порт будут готовы к обмену, они установят на этом входе высокий уровень напряжения, что позволит микропроцессору перейти к выполнению такта T3. Контроль состояния сигнала на входе READY осуществляется в момент действия импульса Ф2 в тактах T2 и T<sub>ож</sub>. Для правильной работы микропроцессора необходимо, чтобы этот сигнал устанавливался и оставался стабильным минимум за 180 нс до оконча-

ния импульса  $\Phi 2$ . Необходимо отметить, что при выполнении микропроцессором тактов ожидания выходные сигналы на его выводах остаются неизменными.

*WAIT* — выходной сигнал, подтверждающий переход микропроцессора в состояние ожидания. Устанавливается и снимается по фронту  $\Phi 1$ .

*HOLD* — входной сигнал, инициирующий перевод шин адресов и данных микропроцессора в высокоимпедансное состояние. Это позволяет производить непосредственный обмен данными между ячейками памяти и портами микроЭВМ, минуя микропроцессор (режим ПДП). Контроль состояния сигнала на входе *HOLD* осуществляется в момент действия импульса  $\Phi 2$  в тактах  $T_2$ , аналогично как и при контроле сигнала на входе *READY*. Для правильной работы микропроцессора необходимо, чтобы этот сигнал на входе *HOLD* устанавливался и оставался стабильным минимум за 180 нс до окончания импульса  $\Phi 2$ . Рекомендуется синхронизировать сигнал *HOLD* сигналом  $\Phi 1$ . Так как перед выполнением циклов ПДП микропроцессор должен закончить обмен в текущем машинном цикле, то сигнал *READY* имеет более высокий приоритет, чем *HOLD*.

*HLDA* — выходной сигнал подтверждения перевода шин адресов и данных микропроцессора в высокоимпедансное состояние. Формируется при обнаружении на входе *HOLD* напряжения высокого уровня. Формирование сигнала происходит с задержкой в 70...120 нс относительно положительного фронта  $\Phi 1$  в такте  $T_3$  или следующем за ним такте ( $T_4$  или  $T_1$  следующего цикла) в зависимости от того, выполнялся ли цикл чтения или записи соответственно. Затем по положительному фронту  $\Phi 2$  с задержкой не более 200 нс линии шин адресов и данных переводятся в высокоимпедансное состояние.

*INT* — входной сигнал запроса прерывания. При его появлении микропроцессор, окончив выполнение текущей команды, устанавливает в первом такте цикла  $M1$  очередной команды на шине данных байт состояния, где вместо признака чтения данных из памяти *MEMR* присутствует признак подтверждения прерывания *INTA*. В результате этого микропроцессор вместо кода очередной команды текущей программы считывает с шины данных микроЭВМ код одной из команд вызова подпрограмм, формируемый дополнительной БИС — контроллером прерывания. Таким образом происходят прерывание выполнения текущей программы и переход к выполнению подпрограммы обработки запроса прерывания.

*INTE* — выходной сигнал разрешения прерывания выполнения текущей программы. Отражает состояние внутреннего триггера микропроцессора, разрешающего прохождение запроса прерывания. Он устанавливается, если ранее в программе была выполнена команда разрешения прерывания *EI*. При обработке запроса прерывания триггер разрешения прохождения запроса прерывания автоматически сбрасывается и снимает выходной сигнал *INTE*, что не позволяет микропроцессору в дальнейшем реагировать на новые запросы прерывания вплоть до выполнения следующей команды *EI*. Запретить прерывание текущей программы можно также командой запрета прерывания *DI* в любом месте выполняемой программы.

*RESET* — входной сигнал установки микропроцессора в исходное состояние. После снятия этого сигнала микропроцессор начинает выполнение программы с команды, содержащейся в ячейке памяти с адресом 0000H. При этом