



ОБЩИЕ ХАРАКТЕРИСТИКИ

Микроконтроллер **КР1878ВЕ1** предназначен для использования в системах управления, работающих в масштабе реального времени. Микроконтроллер отличается высокой производительностью, наличием энергонезависимой памяти данных, возможностью многократного перепрограммирования памяти команд, малым количеством внешних выводов и низким током потребления. Введением в память команд соответствующих прикладных программ осуществляется адаптация микроконтроллера к каждому конкретному применению.

ОСОБЕННОСТИ

- ✦ Память команд на электрически стираемом ППЗУ (ЭСППЗУ) – 1К · 16
- ✦ Память данных: – ОЗУ – 128 · 8
– ЭСППЗУ – 64 · 8
- ✦ Система команд – 52 команды
- ✦ Тактовая частота – 32 kHz , 8 MHz
- ✦ Время выполнения любой команды 2 такта (250 ns при частоте 8 MHz)
- ✦ Прерывания – 7 (начальный пуск, системная ошибка, сторожевой таймер, порт **A**, порт **B**, таймер, окончание записи в ЭСППЗУ)
- ✦ Время реакции на прерывание – 3 такта
- ✦ 12 линий вв/выв с индивидуальным управлением направлением и прерыванием от любой линии. Максимальный ток – 25 мА
- ✦ 16-разрядный таймер с 8-разрядным делителем счетной частоты
- ✦ Сторожевой таймер с автономным генератором
- ✦ Технология изготовления микросхемы – КМОП, два металла
- ✦ Ток потребления:
 - < 2 мА при $V_{CC} = 5 В$ и $f = 5 мГц$
 - 50 мкА при $V_{CC} = 5 В$ и $f = 32 кГц$
 - <1 мкА в режиме **STOP**

Микроконтроллер **КР1878ВЕ1** содержит функционально законченные устройства: центральный процессор, ЭСППЗУ команд, ОЗУ данных, ЭСППЗУ данных, сторожевой таймер, 2 порта ввода/вывода и таймер общего назначения. Обмен данными между центральным процессором, ОЗУ данных и периферийными устройствами производится по единой шине. По цоколевке **КР1878ВЕ1** совместим с подобными микроконтроллерами фирм Microchip, Zilog ...

Структурная схема микроконтроллера приведена на рис. 3 .

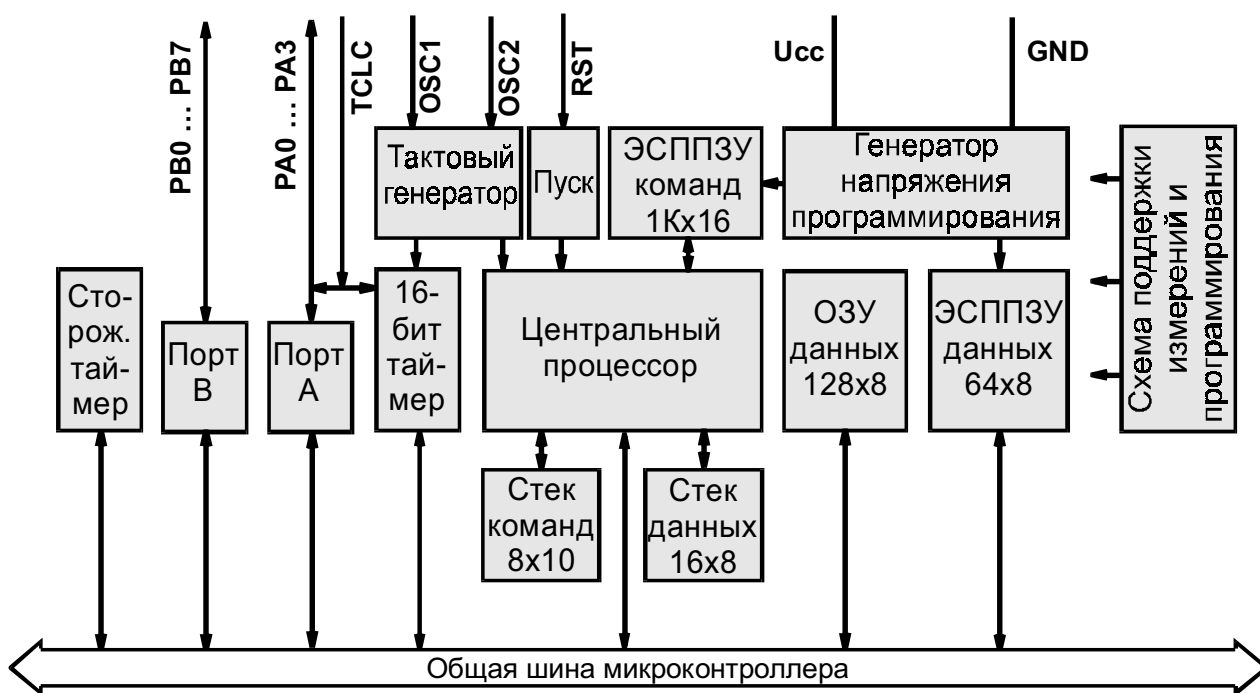


Рис. 3. Структурная схема микроконтроллера



КОНСТРУКЦИЯ

Микроконтроллер изготовлен по КМОП-технологии в пластмассовом 18-выводном корпусе DIP типа 2104.18-8 (рис. 4). Цоколевка и описание выводов приведены на рис. 5 и в табл. 2.

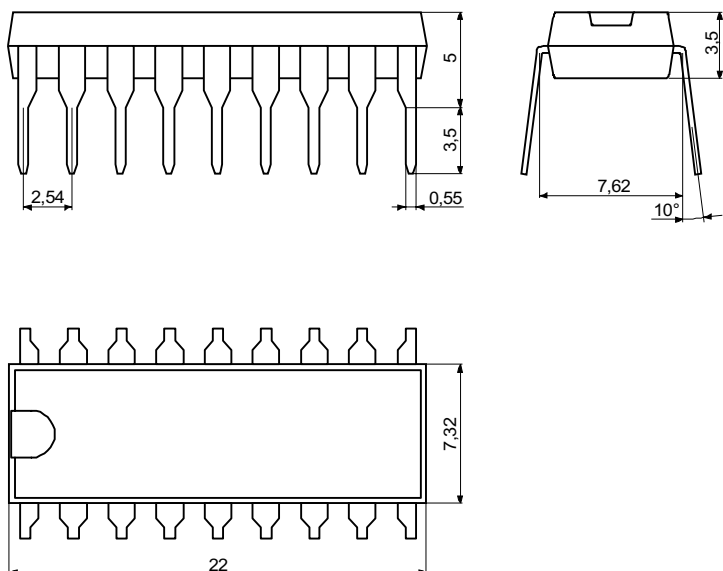


Рис. 4. Корпус 2104.18-8

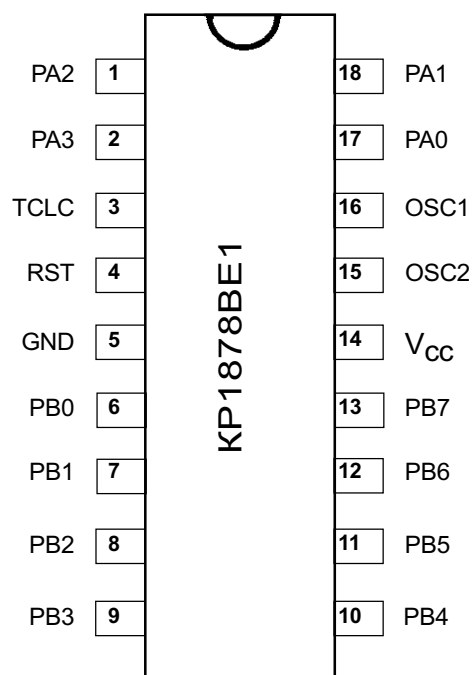


Рис. 5. Выводы

Таблица 2

Описание выводов

Номер вывода	Символ	Назначение
17, 18, 1, 2	PA0 ÷ PA3	Линии ввода / вывода порта А
6 ÷ 13	PB0 ÷ PB7	Линии ввода / вывода порта В
3	TCLC/PA4	Внешняя частота таймера или линия ввода / вывода порта А
15, 16	OSC1, OSC2	Задающие выводы генератора тактовой частоты
4	RST	Входной сигнал установки и программирования
14	V _{cc}	Напряжение питания
5	GND	Общий вывод

ЦЕНТРАЛЬНЫЙ ПРОЦЕССОР

Центральный процессор микроконтроллера **An15E03** предназначен для выполнения арифметических и логических преобразований 8-разрядных операндов, расположенных в памяти микроконтроллера. Архитектура процессора характеризуется разделенной памятью команд и данных. Это позволяет совместить процессы выборки команд и выборки операндов из памяти. Система команд процессора – симметричная, т.е. имеются двухоперандные команды, работающие одновременно с двумя операндами. Выборка операндов из памяти данных в

случае двухоперандных команд производится одновременно по двум различным шинам – данных **SRC** операнда и данных **DST** операнда, с отдельными шинами адресов. Запись результата в память производится также по отдельной шине записи данных. Адрес записи совмещен с адресом чтения операнда **DST**.

Команды процессора имеют размерность 16 разрядов. Для обеспечения механизмов перехода к подпрограммам и прерываний программ в процессоре существует отдельный аппаратный стек глубиной в восемь адресов, где хранятся адреса возврата из подпрограмм и прерываний. В процессоре имеются служебные регистры для поддержки адресации операндов. Для сохранения значений этих регистров и слова состояния процессора при прерываниях программ и переходов к подпрограммам введен отдельный аппаратный стек данных глубиной в 16 байт. Обращения к регистрам периферийных устройств микроконтроллера происходят по тем же шинам и точно так же, как и к оперативной памяти.

Любая из команд процессора выполняется за одинаковый период времени. Одновременно на разных стадиях выполняются три команды процессора. Условные и безусловные переходы происходят по абсолютным адресам памяти команд, что позволяет устранить потерю времени при выполнении переходов.

Принципы адресации операндов, система команд, механизм прерываний программ и временное представление процесса выполнения команд подробнее описаны ниже.

Структурная схема центрального процессора приведена на рисунке 6.

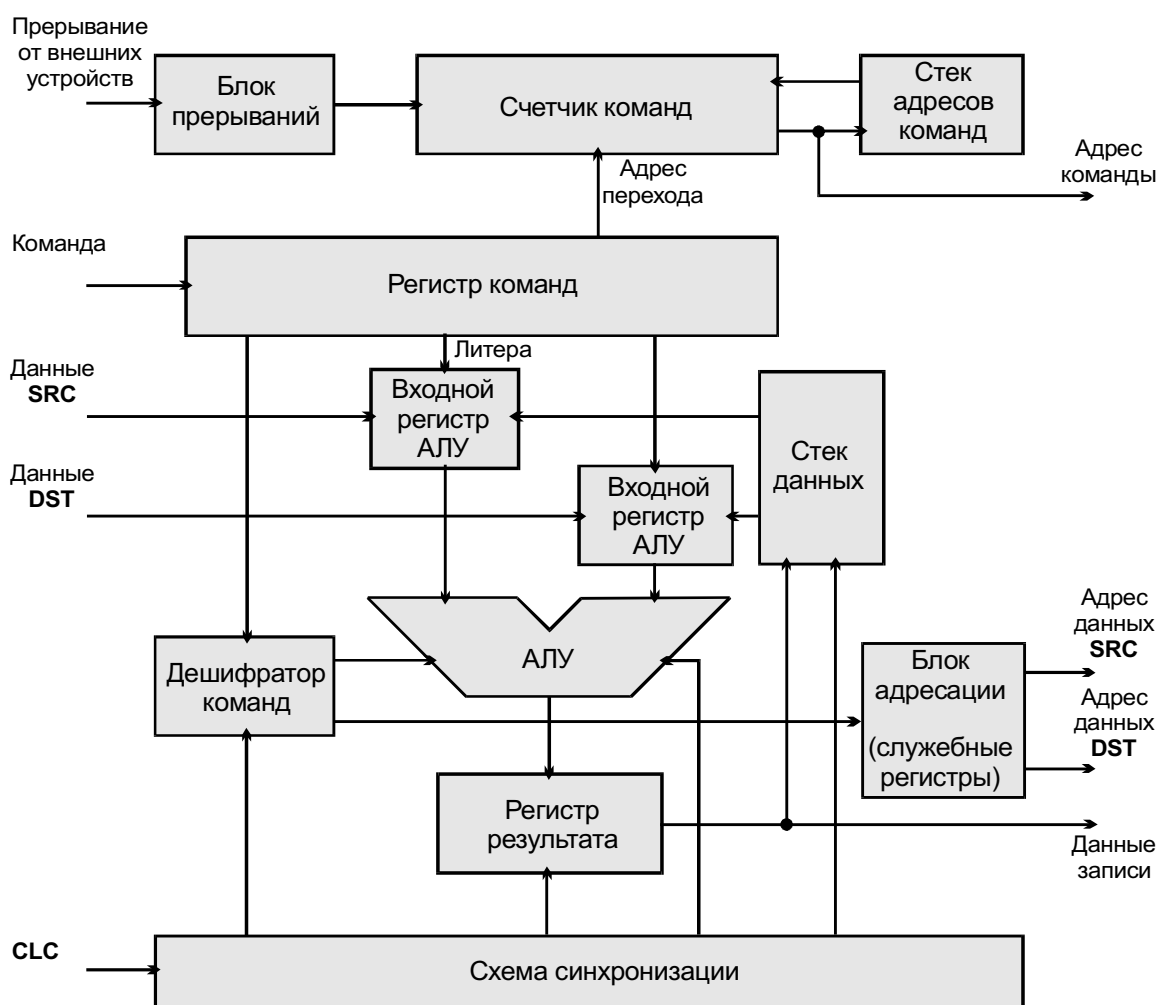


Рис. 6 . Структурная схема центрального процессора



ПРИНЦИПЫ АДРЕСАЦИИ ОПЕРАНДОВ

Все операнды команд (объекты программирования) находятся в едином адресном пространстве памяти данных. Размерность этого пространства в микроконтроллере **КР1878ВЕ1** – 256 байт. Распределение адресного пространства памяти данных приведено в табл. 3.

Таблица. 3

Адресное пространство памяти данных

Адрес ₁₆	Устройство	Адрес ₁₆	Устройство
0	Регистр состояния процессора	1B, 1C	Свободный адрес
1	Рабочий регистр порта А	1D	Регистр управления сторожевого таймера
2	Рабочий регистр порта В	1E ÷ 37	Свободные адреса
3	Свободный адрес	38	Регистр управления блока ЭСППЗУ
4	Регистр управления таймера	39	Регистр адреса блока ЭСППЗУ
5	Рабочий регистр таймера	3A ÷ 3E	Свободные адреса
6 ÷ 18	Свободные адреса	3F	Регистр данных блока ЭСППЗУ
19	Регистр конфигурации порта А	40 ÷ DF	Оперативная память
1A	Регистр конфигурации порта В	C0 ÷ FF	Свободные адреса

Команда имеет одновременный доступ к 32 байтам этого адресного пространства через четыре сегмента (окна) адресации (сегменты **A**, **B**, **C** и **D**) размером по 8 байт. Каждому сегменту соответствует регистр адреса сегмента, значение которого определяет местонахождение данного сегмента в пространстве адресации памяти данных (табл. 4.)

Таблица. 4

Сегменты адресации

Сегмент A	Сегмент B	Сегмент C	Сегмент D
A0	B0	C0	D0
A1	B1	C1	D1
A2	B2	C2	D2
A3	B3	C3	D3
A4	B4	C4	D4
A5	B5	C5	D5
A6	B6	C6	D6
A7	B7	C7	D7

Регистры адресов сегментов принадлежат к категории служебных регистров процессора **SR**. В процессоре имеется восемь служебных регистров (**SR0**, **SR7**), обращение к которым производится специальными командами процессора. Значения этих регистров можно сохранять в отдельном аппаратном стеке данных и восстанавливать из него с помощью соответствующих команд. Список служебных регистров приведен в табл. 5.

**Служебные регистры процессора**

№ регистра	Назначение
SR0	Регистр адреса сегмента A
SR1	Регистр адреса сегмента B
SR2	Регистр адреса сегмента C
SR3	Регистр адреса сегмента D
SR4	Регистр адреса 0-го косвенного регистра
SR5	Регистр режима 0-го косвенного регистра
SR6	Регистр адреса 1-го косвенного регистра
SR7	Регистр режима 1-го косвенного регистра

Формат служебных регистров

Первые четыре служебных регистра имеют одинаковый формат и одинаковым образом участвуют в образовании полного адреса операнда.

Разряды	7	6	5	4	3	2	1	0
Регистры SRO ÷ 3	x	x	x	Старшие 5 разрядов адреса				

Регистры адреса сегментов **A, B, C** и **D**

Примечание. Здесь и далее знаком "x" отмечены не используемые разряды регистров, находящиеся в состоянии "0".

Разряды	7	6	5	4	3	2	1	0
Регистр SR4	Полный адрес операнда в памяти данных							

Регистр адреса 0-го косвенного регистра **IR0**

Разряды	7	6	5	4	3	2	1	0
Регистр SR5	Режим IR0	x	x	x	x	x	x	x

Регистр режима 0-го косвенного регистра **IR0**

Разряды	7	6	5	4	3	2	1	0
Регистр SR6	8 разрядов адреса в памяти данных (команд)							

Регистр адреса 1-го косвенного регистра **IR1**

Разряды	7	6	5	4	3	2	1	0
Регистр SR7	Режим IR1	x	x	x	3 старших разряда адреса в памяти команд			

Регистр режима 1-го косвенного регистра **IR1**



Полный адрес операнда образуется слиянием сегментного индекса и текущего значения соответствующего данному сегменту регистра адреса сегмента.

Адрес операнда в памяти данных

Разряды

7	6	5	4	3	2	1	0
Значение регистра адреса сегмента				Индекс сегмента			
5 старших разрядов адреса				3 младших разряда адреса			

Таким образом, программист имеет доступ к четырем участкам памяти размером в восемь байт каждый. Перемещение каждого участка памяти в нужную область памяти данных производится записью адреса сегмента в соответствующий регистр адреса сегмента с помощью команды загрузки служебных регистров.

Два старших индекса сегмента **D** имеют несколько иной статус и используются для полной адресации операнда в адресном пространстве памяти данных в качестве регистров косвенной адресации **IR0** и **IR1**. Обращение команды к индексу **D6 (IR0)** транслируется по адресу, записанному в регистре адреса **0**-го косвенного регистра – **4**-м служебном регистре. Соответственно обращение к **D7 (IR1)** транслируется по адресу, записанному в регистре адреса **1**-го косвенного регистра – **5**-м служебном регистре.

Режим работы **IR0**

Значение битов 7 и 6	Режим работы IR0
00	Отключение регистра
01	Автоинкремент адреса
10	Автодекремент адреса
11	Чистая косвенность

Режим работы **IR1**

Значение битов 7 и 6	Режим работы IR1
00	Адресация памяти команд
01	Автоинкремент адреса
10	Автодекремент адреса
11	Чистая косвенность

Оба косвенных регистра обладают возможностями изменения адреса после обращения либо в сторону увеличения, либо в сторону уменьшения адреса на единицу в зависимости от режима работы косвенного регистра. Регистр **IR0** имеет режим отключения, при котором адрес обращения формируется из значения регистра адреса сегмента **D** и индекса **6**. Регистр **IR1** вместо режима отключения имеет режим обращения к памяти команд. В этом режиме адрес операнда, находящегося в памяти команд, формируется из полного значения шестого и трех младших разрядов седьмого служебных регистров. Адресация памяти команд через **IR1** производится с точностью до байта. Режим работы **IR1** при адресации памяти команд автоинкрементный. То есть после обращения к текущему байту памяти команд регистр **IR1** указывает на следующий байт памяти команд.

Смена режимов работы косвенных регистров и загрузка в них адреса производится специальной командой загрузки служебных регистров.

Регистры косвенной адресации удобно использовать при работе с упорядоченными структурами данных, используя автоинкрементный либо автодекрементный режимы.

КРАТКИЙ ПРИМЕР ИСПОЛЬЗОВАНИЯ АДРЕСАЦИИ:

Выполняется команда **MOV %d7,%b4** – переслать содержимое байта памяти данных, который адресуется через сегмент **B** и расположен по четвертому индексу в сегменте **B**, в байт памяти данных, расположенный в памяти данных по адресу, содержащемуся в косвенном регистре **IR1**.

Косвенный регистр работает в автоинкрементном режиме.



Содержимое служебных регистров до выполнения команды:

$$SR1 = 17_{16}; SR6 = 96_{16}; SR7 = 40_{16}.$$

Операнд источника, указанный в команде b4

0 1 1 0 0

Регистр адреса сегмента **SR1**

0 0 0 1 0 1 1 1

Реальный адрес операнда источника =

1 0 1 1 1 1 0 0

DC₁₆

Операнд приемника, указанный в команде

1 1 1 1 1

d7

Косвенный регистр **IR1**

SR7

0 1 0 0 0 0 0

режим автоинкремента

SR6

1 0 0 1 0 1 1 0

Адрес операнда приемника =

1 0 0 1 0 1 1 0 96₁₆

Содержимое служебных регистров после выполнения команды:

$$SR1 = 17_{16}; SR6 = 97_{16}; SR7 = 40_{16}.$$

РЕГИСТРЫ ПЕРИФЕРИЙНЫХ УСТРОЙСТВ

Регистры периферийных устройств (портов ввода/вывода, таймеров, ЭСППЗУ данных) размещены в едином адресном пространстве вместе с оперативной памятью данных в фиксированной области внешних устройств, занимающих начальные 64 байта. Обращение к этим регистрам можно осуществлять любой из команд процессора, точно так же как и к байтам оперативной памяти. Некоторые особенности обращения к регистрам периферийных устройств, проистекающие из организации вычислительного процесса, описаны в соответствующем разделе данной публикации.

Наличие четырех сегментов адресации вносит, с одной стороны, определенные неудобства, так как необходимо помнить, какое местоположение в данный момент занимает каждый из сегментов в адресном пространстве. С другой стороны, использование сегментной адресации позволяет производить быстрое переключение контекста при переходах к подпрограммам и при прерываниях программ.

Каждый из сегментов можно интерпретировать как восемь виртуальных 8-разрядных регистров. То есть программист имеет в своем распоряжении тридцать два 8-разрядных регистра, которые он может располагать в памяти по своему усмотрению. Рассмотрим пример использования сегментов адресации. Один из сегментов удобно размещать в 0-х адресах области регистров периферийных устройств. Другие один или два сегмента размещаются в области глобальных переменных программы. Четвертый сегмент размещен в области локальных переменных данной подпрограммы. В случае вызова подпрограммы или прерывания программы для того, чтобы вызванная подпрограмма могла использовать свои локальные переменные, не испортив переменные прерванной или вызывающей программы, достаточно сохранить в стеке данных командой **PUSH** текущее значение регистра адреса сегмента локальных переменных и загрузить новое требуемое значение этого регистра. По завершении работы вызываемой подпрограммы достаточно командой **POP** восстановить старое значение регистра адреса сегмента.



Адресное пространство памяти данных	Расположение сегментов до вызова программы и после возврата	Расположение сегментов после вызова программы
Регистры периферийных устройств	0 ₁₆ – 40 ₁₆ Сегмент А SR0 = 0	Сегмент А
Глобальные переменные	40 ₁₆ – 48 ₁₆ Сегмент В SR1 = 8 ₁₆	Сегмент В
Локальные переменные	48 ₁₆ – 50 ₁₆ Сегмент С SR2 = 9 ₁₆	Сегмент С
	50 ₁₆ – 60 ₁₆ Сегмент D SR3 = A ₁₆	
	60 ₁₆ – Сегмент D SR3 = C ₁₆	

Возможности сегментной адресации в сочетании с использованием регистров косвенной адресации, обеспечивают нормальную работу с памятью данных.

РЕГИСТР СОСТОЯНИЯ ПРОЦЕССОРА

Регистр состояния процессора содержит разряды, отражающие текущее состояние вычислительного процесса. Отдельные его разряды могут изменяться в процессе выполнения всех команд, некоторые – командами изменения регистра состояния. Установка разряда – это изменение состояния данного разряда в логическую **1**, сброс разряда – в логический **0**. При прерываниях содержимое регистра состояния процессора всегда сохраняется в стеке данных, а при выполнении команды возврата из прерывания – восстанавливается из стека.

Представление отрицательных чисел при выполнении арифметических команд дается в дополнительном коде. Старший разряд байтового операнда несет информацию о знаке числа. У положительных чисел знаковый разряд равен нулю, а у отрицательных – единице. Наибольшее число $3F_{16} = 128_{10}$, а наименьшее $-80_{16} = -128_{10}$. Перенос из старшего разряда происходит при сложении двух отрицательных чисел и при сложении большего по модулю положительного числа с меньшим по модулю числом или равным по модулю отрицательным числом. Арифметическое переполнение происходит при сложении двух одинаковых по знаку чисел, если результат по модулю превышает 128_{10} .

Формат регистра состояния процессора.

Разряды	7	6	5	4	3	2	1	0
Мнемоника	Резерв	DC	OF	IE	S	Z	C	

C-разряд – признак переноса, устанавливается в арифметических командах и используется в командах сдвига;

Z-разряд – признак равенства нулю, устанавливается в случае равенства нулю результата выполнения команды;

S-разряд – знаковый признак, устанавливается в случае отрицательного результата выполнения команды;

IE-разряд – разрешение прерывания, если разряд **IE** установлен, разрешается прерывание от периферийных устройств, при сброшенном **IE**-разряде запрос на прерывания игнорируются;



OF-разряд – разряд арифметического переполнения, устанавливается при арифметическом переполнении в предыдущей команде;

DC-разряд – разряд тетрадного переноса, устанавливается при возникновении тетрадного переноса при выполнении предыдущей команды;

7-й и 6-й разряды регистра состояния процессора не используются и зарезервированы для других моделей процессора с большим размером памяти команд.

Разряды **S**, **Z** и **C** изменяются в процессе выполнения команд и также могут быть установлены или сброшены командами установки и сброса регистра состояния. По факту установки или сброса этих разрядов могут быть осуществлены условные переходы. Разряд **IE** изменяется только командами установки и сброса регистра состояния. Разряды **OF** и **DC** устанавливаются по результатам выполнения арифметических команд и сбрасываются любой другой следующей командой. Наличие этих разрядов проверяется командами тестирования переполнения и тетрадного переноса. Регистр состояния доступен по чтению физического адреса **0** в области адресов периферийных устройств.

ЭЛЕКТРИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Основные электрические характеристики микроконтроллера КР1878ВЕ1 и предельные значения допустимых режимов эксплуатации приведены в таблицах 6 и 7 соответственно.

Таблица 6

Основные электрические характеристики

Параметр	Сим-вол	Единица	Мин	Тип	Макс	Условия
Напряжение питания	V_{CC}	V	4,0	-	6,0	-
Напряжение сохранения данных в ОЗУ	V_{DR}	V	1,5	-	-	CPU in STOP mode
Ток потребления	I_{CCA}	μA	-	60	400	$F_{OSC} = 32 \text{ KHz}, V_{CC} = 5,5 \text{ V}$ (WDT disabled)
		mA	-	1,5	3	$F_{OSC} = 1 \text{ MHz}, V_{CC} = 5,5 \text{ V}$
		mA	-	2	4	$F_{OSC} = 6 \text{ MHz}, V_{CC} = 5,5 \text{ V}$
Напряжение низкого уровня на - входах: - PA0 ÷ PA3, PB0 ÷ PB7 - TCLC, RST, OCS1 - выходах: - PA0÷PA3, PB0÷PB7	V_{IL}	V	0	C	0,8	$4,5 \leq V_{CC} \leq 5,5 \text{ V}$ $4 \leq V_{CC} \leq 6 \text{ V}$ $4 \leq V_{CC} \leq 6 \text{ V}$ $I_{OL} = 8,5 \text{ mA}, V_{CC} = 4,5 \text{ V}$
			0		$0,16 V_{CC}$	
			0		$0,2 V_{CC}$	
Напряжение высокого уровня на: - входах: - PA0 ÷ PA3, PB0 ÷ PB7 - TCLC, RST, OCS1 - выходах: - PA0÷PA3, PB0÷PB7	V_{IH}	V	$0,36 V_{CC}$	-	V_{CC}	$4,5 \leq V_{CC} \leq 5,5 \text{ V}$ $4 \leq V_{CC} \leq 6 \text{ V}$ $4 \leq V_{CC} \leq 6 \text{ V}$ $I_{OL} = -3 \text{ mA}, V_{CC} = 4,5 \text{ V}$
			$0,48 V_{CC}$		V_{CC}	
			$0,85 V_{CC}$		$0,2 V_{CC}$	
Ток утечки через нагрузочный транзистор на выходах портов	I_{PU}	μA	50	250	400	$V_{CC} = 5 \text{ V}$
Ток утечки для выводов: PA0 ÷ PA3, PB0 ÷ PB7; TCLC, RST, OCS1	I_{IL}	μA	-	-	± 1 ± 5	$GND \leq U \leq U_{CC}$ Вывод в 3-м состоянии $GND \leq V \leq V_{CC}$
Емкость выходов	C_{IO}	pF	-	-	50	-
Циклов перезаписи в ЭСППЗУ	N_{WR}	-	-	10^5	-	-
Время записи в ЭСППЗУ	T_{WR}	ms	-	5	-	-

**Предельные значения допустимых режимов эксплуатации**

Параметр	Единица	Значение
Температура хранения	°C	-65 ÷ +150
Напряжение между выводами V_{CC} и GND	V	0 ÷ +7,5
Напряжение между выводами RST и GND	V	0 ÷ +14
Напряжение между остальными выводами и GND	V	-0.6 ÷ ($V_{CC}+0,6$)
Максимальный ток через вывод V_{CC}	mA	100
Максимальный ток через вывод GND	mA	150
Максимальный ток через выводы портов А и Б	mA	80