МИКРОСХЕМЫ ИНТЕГРАЛЬНЫЕ

Л1874ВЕ36, 1874ВЕ36, Л1874ВЕ36А, 1874ВЕ36А

Техническое описание

КФДЛ.431295.008 ТО

2012

Содержание

Введение

В настоящем техническом описании КФДЛ.431295.008 ТО приведено подробное описание архитектуры, функционального построения, системы команд и особенностей применения ИС серии 1874: Л1874ВЕ36, 1874ВЕ36, Л1874ВЕ36А, 1874ВЕ36А, которые представляют собой СБИС однокристальных 16-разрядных микроконтроллеров с внутренней программной памятью объемом 8 Кбайт.

Микроконтроллеры имеют высокую производительность, развитую систему встроенных функциональных блоков и достаточно мощную систему команд и предназначены для применения во встроенных системах для реализации функций управления и вычисления.

Изделия Л1874ВЕ36, 1874ВЕ36, Л1874ВЕ36А, 1874ВЕ36А выпускаются и поставляются по техническим условиям АЕЯР.431280.169ТУ: ИС Л1874ВЕ36, Л1874ВЕ36А – в корпусах 6108.68-1; ИС 1874ВЕ36, 1874ВЕ36А – в корпусах 4235.88-1 .

Микросхемы Л1874ВЕ36, 1874ВЕ36, Л1874ВЕ36А, 1874ВЕ36А могут поставляться как без встроенной памяти программ, так и с внутренним масочным ПЗУ программ.

Функциональным аналогом ИС является микроконтроллер 83С196КВ12 фирмы Intel.

Для разработки и отладки систем на основе микроконтроллеров разработаны и могут поставляться программно-аппаратные средства отладки:

- внутрисхемный эмулятор PICE-196 с соответствующим сменным адаптером (POD для ИС 1874ВЕ36, 1874ВЕ36А) и эмуляционной головкой;
- программный Отладчик-Симулятор PDS-96;
- кросс-компилятор Си МСС-96 и макроассемблер МСА-96.

Все указанные средства поддержки поставляются АО "Фитон", г. Москва.

Настоящее техническое описание может служить практическим руководством по применению микроконтроллеров для разработчиков систем на основе ИС Л1874ВЕ36, 1874ВЕ36, Л1874ВЕ36А, 1874ВЕ36А.

При обнаружении в техническом описании неточностей и опечаток просьба сообщать разработчикам:

- Тел.: (473) 226-20-35
- Факс: (473) 226-98-95
- E-mail: niiet@niiet.ru
- http://www.niiet.ru

1 Назначение и основные технические характеристики микросхем

Микросхемы представляют собой СБИС однокристальных 16-разрядных микроконтроллеров с внутренней программной памятью объемом 8 Кбайт. Микроконтроллеры предназначены для применения в системах управления и диагностики автомобильных двигателей. В области промышленного производства микроконтроллеры могут быть использованы для управления робототехническими комплексами, в системах автоматизации технологических процессов, в системах автоматизированного управления электроприводом, оргтехнике, вычислительной технике, телекоммуникационной технике и т. п.

1.1 Архитектурные характеристики микроконтроллеров:

- количество 8-разрядных портов ввода-вывода – 5;
- разрядность шины адресов-данных, бит – 8, 16;
- емкость регистрового блока, байт – 232;
- разрядность РАЛУ, бит – 16;
- объем адресуемой памяти, байт – 64К;
- объем внутренней памяти, байт – 8К;
- количество типов команд – 105;
- разрядность команды, байт – 1, …, 8;
- количество 16-разрядных таймеров – 6 (в том числе 4 программных);
- количество источников прерываний/векторов – 28/16;
- полностью дуплексный последовательный порт;
- высокоскоростная подсистема ввода-вывода;
- блок формирования сигналов ШИМ;
- 8-канальный 10-разрядный АЦП;
- режим поддержки мультипроцессорного протокола (HOLD#/HLDA#);
- режимы пониженного потребления мощности и хранения;
- 16-разрядный дежурный таймер.

1.2 Конструктивные характеристики микросхем

Кристалл микросхем ПАКД.757644.231 выполнен по КМОП технологии с поликремниевыми затворами, одним уровнем металлизации и n-карманом.

Размер кристалла (7,84 × 6,74) мм^2.

Микросхемы выполняются в двух вариантах:

1) 68-выводном металлокерамическом корпусе с матричным расположением выводов (PGA) типа 6108.68-1 ("Марево-68") – ИС Л1874ВЕ36, Л1874ВЕ36А;

2) 88-выводном металлокерамическом корпусе с планарным расположением выводов (LCC) типа 4235.88-1 ("Монополия-88") – ИС 1874ВЕ36, 1874ВЕ36А.

1.3 Электрические характеристики микросхем:

- номинальное значение напряжения питания, В – 5,0;
- допустимые отклонения напряжения питания от номинального значения, В – ±0,5;
- потребляемая мощность, мВт – 500;
- частота тактового сигнала, МГц – (3,5 – 20);
- диапазон рабочих температур – от минус 60 до 85 °С.

Примечание – Все значения электрических параметров ИС приведены в технических условиях АЕЯР.431280.169ТУ. Значения параметров, приведенные в ТО, являются справочными.

Электрические параметры микросхем при приемке и поставке приведены в таблице 1.3.1.

Значения предельно допустимых и предельных режимов эксплуатации микросхем приведены в таблице 1.3.2.

Условное графическое обозначение микросхем приведено на рисунках 1.3.1 и 1.3.2. Функциональное назначение выводов приведено в таблице 1.3.3.

Таблица 1.3.1 – Электрические характеристики микросхем при приёмке и поставке (справочные данные)

Наименование параметра, единица измерения	Буквенное обозначение параметра	Норма		Режим измерения	Примечание
		не менее	не более		
1 Выходное напряжение низкого уровня, В	U_{OL}	–	0,45	I_{OL} = 3,2 мА	1
2 Выходное напряжение высокого уровня, В	U_{OH1}	3,8	–	I_{OH} =–0,03 мА	2
3 Выходное напряжение высокого уровня, В	U_{OH2}	3,8	–	I_{OH} = –3,2 мкА	3
4 Входной ток высокого уровня, мкА	I_{IH}	–650	–	U_{IH} = 2,0 В	8
5 Входной ток низкого уровня, мкА	I_{IL}	–50	–	U_{IL} = 0,45 В	9
6 Ток утечки на входе, мкА	I_{ILH1} I_{ILL1}	–10	10	U_{IH} = 5,2 В U_{IH} = 5,5 В	4 5
7 Ток утечки на входе, мкА	I_{ILH2} I_{ILL2}	–3	3	U_{IH} = 5,5 В	6 7
8 Ток потребления первого источника питания, мА	I_{OCC1}	–	100	U_{CC1} = 5,5 В U_{CC2} =5,5 В	10
9 Ток потребления второго источника питания, мА	I_{OCC2}	–	10	U_{CC1} = 5,5 В U_{CC2} = 5,5 В	11
10 Время задержки распространения сигнала от входа XTAL1 до выхода CLCOUT, не более, нс	t_p (C,LH-CO,H)	–	85	U_{CC1} = 5,0 В U_{CC2} = 5,0 В U_{IH} = 3,0 В	
11 Функциональный контроль	ФК1 ФК2				12 13

Примечания

1 Для выходов AD.0 – AD.15, RD#, WR#, BHE#, INST, HSO.0 – HSO.5, PWM, CLCOUT, P2.0/TXD, RXD (в режиме 0), P1.0 – P1.7, P2.6, P2.7.

2 Для выходов P1.0 – P1.7, P2.6, P2.7.

3 Для выходов AD.0 – AD.15, RD#, WR#, BHE#, INST, HSO.0 – HSO.5, PWM, CLCOUT, P2.0/TXD, RXD (в режиме 0), ALE.

4 Для входов HSI.0 – HSI.3, EA#, READY, BW, NMI, P2.1/RXD, P2.2/EXTINT, P2.3/T2CLK, P2.4/T2RST. На измеряемый вывод подается U_{IH}, а на остальные U_{IL}.

5 Для входов HSI.0 – HSI.3, EA#, READY, BW, NMI, P2.1/RXD, P2.2/ EXTINT, P2.3/T2CLK, P2.4/ T2RST. На измеряемый вывод подается U_{IL}, а на остальные U_{IH}.

6 Для входов P0.0 – P0.7. На измеряемый вход подается U_{IH}, а на остальные U_{IL}.

7 Для входов P0.0 – P0.7. На измеряемый вход подается U_{IL}, а на остальные U_{IH}.

Окончание таблицы 1.3.1

8 Для входов/выходов P1.0 – P1.7, P2.6 , P2.7. На измеряемый вывод подается U_{IH}, а на остальные U_{IL}.

9 Для входов/выходов P1.0 – P1.7, P2.6 , P2.7. На измеряемый вывод подается U_{IL}, а на остальные U_{IH}.

10 Измерение тока потребления первого источника питания при U_{CC1} = 5,5 В, f_C =12/20 МГц*.

11 Измерение тока потребления аналого-цифрового преобразователя при f_C =12/20 МГц*.

12 Функциональный контроль ФК1 проводится при:

- f_C = (3,5 – 20) МГц*;
- U_{IL} по входам: P0.0 – 0,2 В, P0.1 – 0,16 В, P0.2 – 0,32 В, P0.3 – 0,64 В;
- U_{IH} по входам: P0.4 – 2,8 В, P0.5 – 3,2 В, P0.6 – 3,8 В, P0.7 – 4,2 В;
- по остальным входам: уровни U_{IH} = 3 В, U_{IL} = 0 В.

13 Функциональный контроль ФК2 проводится при f_C = 3,5 МГц.

* 20 МГц – для ИС Л1874ВЕ36, 1874ВЕ36; 12 МГц – для Л1874ВЕ36А, 1874ВЕ36А.

Таблица 1.3.2 – Предельно допустимые и предельные режимы эксплуатации микросхем

Наименование параметра режима, единица измерения	Буквенное обозначение параметра	Предельно допустимый режим		Предельный режим
		не менее	не более	
1 Напряжение 1-го источника питания, В	U_{CC1}	4,5	5,5	7,0
2 Напряжение 2-го источника питания, В	U_{CC2}	4,0	5,5	7,0
3 Частота следования импульсов тактового сигнала, МГц	f_C	3,5	12/20*	
4 Входное напряжение низкого уровня, В	U_{IL}	0	0,8	– 0,5
5 Входное напряжение высокого уровня, В	U_{IH}	$0,2U_{CC1}$+1,0	U_{CC1}+0,5	7,0 при U_{CC1}= 7 В
6 Входное напряжение высокого уровня тактового сигнала, В	U_{IHCI}	$0,7U_{CC1}$	U_{CC1}+0,5	7,0 при U_{CC1}= 7 В
7 Входное напряжение высокого уровня сигнала "сброс", В	$U_{IHRESET}$	2,6	U_{CC1}+0,5	7,0 при U_{CC1}= 7 В
8 Выходной ток высокого уровня по выводам RD#, WR#, ALE, BHE#, INST, HSO, P2.5/PWM, CLCOUT, P3.0 – P3.7, P4.0 – P4.7, P2.0/TXD, P2.1/RXD, мА	I_{OH}		–3,2	–10
9 Выходной ток низкого уровня, мА	I_{OL}		3,2	10
10 Емкость нагрузки, пФ	C_L		80	200
11 Длительность фронтов входных сигналов,нс	t_{LH}, t_{HL}	–	7	25

Примечание – Время действия предельных режимов по пунктам 1, 2, 4 – 7 не более 10 мс.

* Показатель в числителе для ИС Л1874ВЕ36А и 1874ВЕ36А, показатель в знаменателе для ИС Л1874ВЕ36 и 1874ВЕ36.

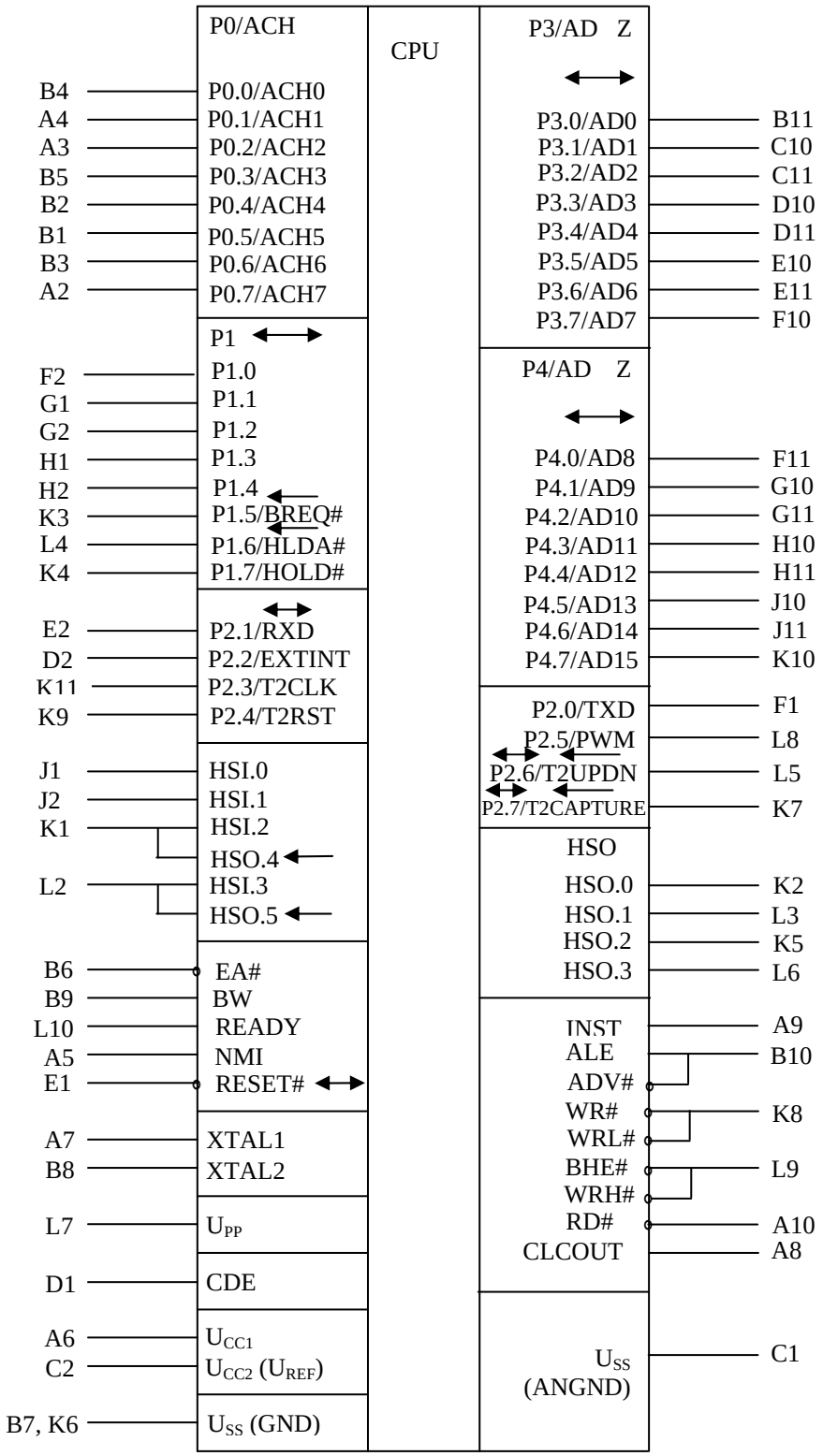


Рисунок 1.3.1 – Условное графическое обозначение микросхем Л1874ВЕ36 Л1874ВЕ36А в корпусе 6108.68-1 ("Марево-68")

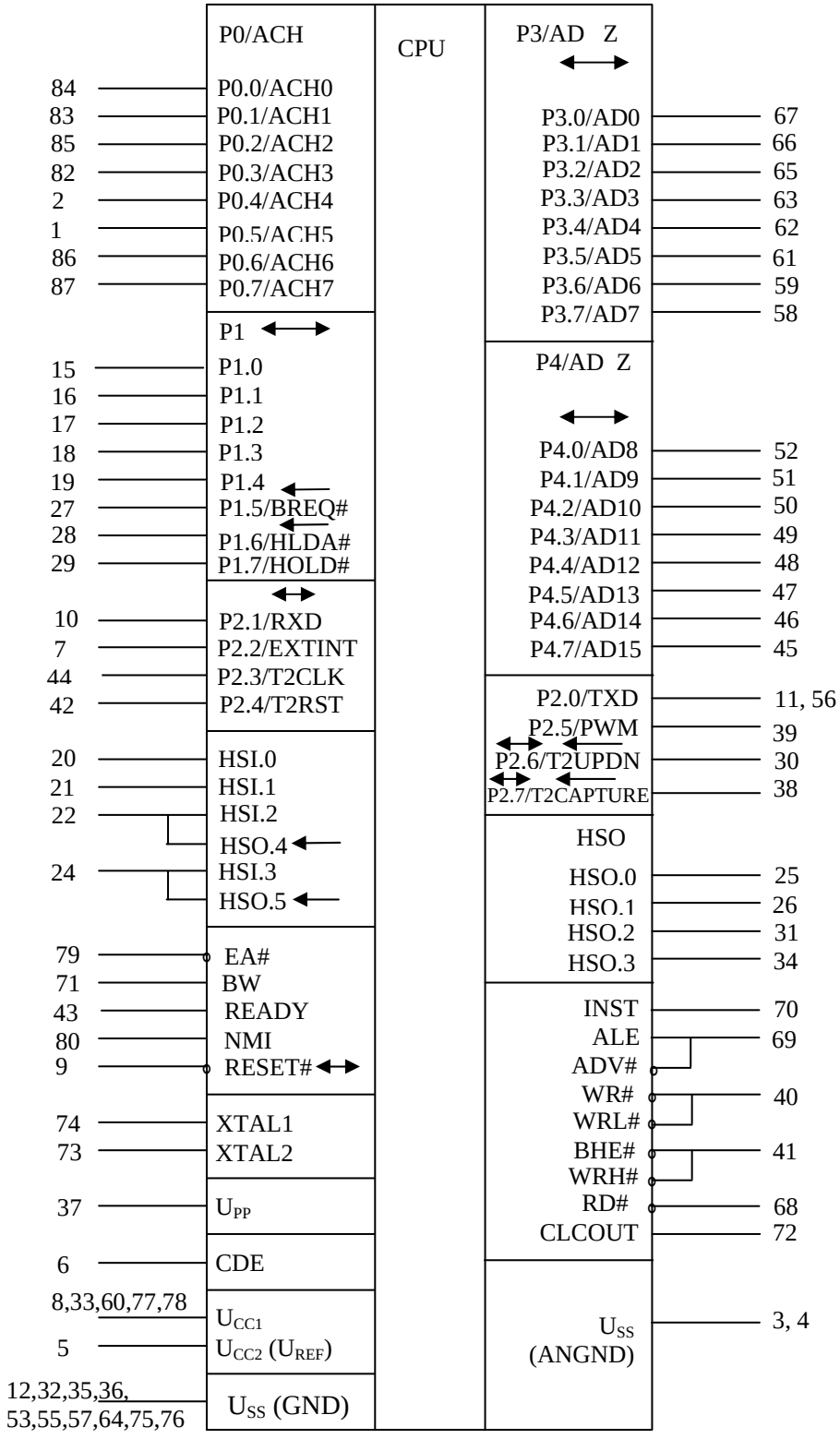


Рисунок 1.3.2 – Условное графическое обозначение микросхем 1874ВЕ36, 1874ВЕ36А в корпусе 4235.88-1 ("Монополия-88")

Таблица 1.3.3 – Функциональное назначение выводов микросхем

Обозначение вывода	Номер вывода, Тип корпуса 6108.68-1	Номер вывода, Тип корпуса 4235.88-1	Функциональное назначение	Тип вывода	Обозначение альтернативной функции вывода
1	2	3	4	5	6
P0.0	B4	84	Вход "порт 0, 0 разряд" Вход АЦП, канал 0	I I	 ACH.0
P0.1	A4	83	Вход "порт 0, 1 разряд" Вход АЦП, канал 1	I I	 ACH.1
P0.2	A3	85	Вход "порт 0, 2 разряд" Вход АЦП, канал 2	I I	 ACH.2
P0.3	B5	82	Вход "порт 0, 3 разряд" Вход АЦП, канал 3	I I	 ACH.3
P0.4	B2	2	Вход "порт 0, 4 разряд" Вход АЦП, канал 4	I I	 ACH.4
P0.5	B1	1	Вход "порт 0, 5 разряд" Вход АЦП, канал 5	I I	 ACH.5
P0.6	B3	86	Вход "порт 0, 6 разряд" Вход АЦП, канал 6	I I	 ACH.6
P0.7	A2	87	Вход "порт 0, 7 разряд" Вход АЦП, канал 7	I I	 ACH.7
P1.0	F2	15	Вход/выход "порт 1, 0 разряд"	I/O	
P1.1	G1	16	Вход/выход "порт 1, 1 разряд"	I/O	
P1.2	G2	17	Вход/выход "порт 1, 2 разряд"	I/O	
P1.3	H1	18	Вход/выход "порт 1, 3 разряд"	I/O	
P1.4	H2	19	Вход/выход "порт 1, 4 разряд"	I/O	
P1.5	K3	27	Вход/выход "порт 1, 5 разряд" Выход "запрос шины"	I/O O	 BREQ#
P1.6	L4	28	Вход/выход "порт 1, 6 разряд" Выход "подтверждение захвата шины"	I/O O	 HLDA#
P1.7	K4	29	Вход/выход "порт 1, 7 разряд" Вход "требование захвата шины"	I/O I	 HOLD#
P2.0	F1	11, 56	Выход "порт 2, 0 разряд" Выход последовательных данных	O O	 TXD
P2.1	E2	10	Вход "порт 2, 1 разряд" Вход/выход последовательных данных	I I/O	 RXD
P2.2	D2	7	Вход "порт 2, 2 разряд" Вход внешнего прерывания	I I	 EXTINT
P2.3	K11	44	Вход "порт 2, 3 разряд" Вход "синхронизация таймера 2"	I I	 T2CLK
P2.4	K9	42	Вход "порт 2, 4 разряд" Вход "сброс таймера 2"	I I	 T2RST
P2.5	L8	39	Выход "порт 2, 5 разряд" Выход ШИМ	O O	 PWM
P2.6	L5	30	Вход/выход "порт 2, 6 разряд" Вход "выбор режима таймера 2"	I/O I	 T2UPDN

Продолжение таблицы 1.3.3

1	2	3	4	5	6
P2.7	K7	38	Вход/выход "порт 2, 7 разряд" Вход "выборка данных таймера 2"	I/O I	 T2CAPTURE
P3.0	B11	67	Вход/выход "порт 3, 0 разряд" Вход/выход "адрес-данные, 0 разряд"	I/O I/O/Z	 AD0
P3.1	C10	66	Вход/выход "порт 3, 1 разряд" Вход/выход "адрес-данные, 1 разряд"	I/O I/O/Z	 AD1
P3.2	C11	65	Вход/выход "порт 3, 2 разряд" Вход/выход "адрес-данные, 2 разряд"	I/O I/O/Z	 AD2
P3.3	D10	63	Вход/выход "порт 3, 3 разряд" Вход/выход "адрес-данные, 3 разряд"	I/O I/O/Z	 AD3
P3.4	D11	62	Вход/выход "порт 3, 4 разряд" Вход/выход "адрес-данные, 4 разряд"	I/O I/O/Z	 AD4
P3.5	E10	61	Вход/выход "порт 3, 5 разряд" Вход/выход "адрес-данные, 5 разряд"	I/O I/O/Z	 AD5
P3.6	E11	59	Вход/выход "порт 3, 6 разряд" Вход/выход "адрес-данные, 6 разряд"	I/O I/O/Z	 AD6
P3.7	F10	58	Вход/выход "порт 3, 7 разряд" Вход/выход "адрес-данные, 7 разряд"	I/O I/O/Z	 AD7
P4.0	F11	52	Вход/выход "порт 4, 0 разряд" Вход/выход "адрес-данные, 8 разряд"	I/O I/O/Z	 AD8
P4.1	G10	51	Вход/выход "порт 4, 1 разряд" Вход/выход "адрес-данные, 9 разряд"	I/O I/O/Z	 AD9
P4.2	G11	50	Вход/выход "порт 4, 2 разряд" Вход/выход "адрес-данные, 10 разряд"	I/O I/O/Z	 AD10
P4.3	H10	49	Вход/выход "порт 4, 3 разряд" Вход/выход "адрес-данные, 11 разряд"	I/O I/O/Z	 AD11
P4.4	H11	48	Вход/выход "порт 4, 4 разряд" Вход/выход "адрес-данные, 12 разряд"	I/O I/O/Z	 AD12
P4.5	J10	47	Вход/выход "порт 4, 5 разряд" Вход/выход "адрес-данные, 13 разряд"	I/O I/O/Z	 AD13
P4.6	J11	46	Вход/выход "порт 4, 6 разряд" Вход/выход "адрес-данные, 14 разряд"	I/O I/O/Z	 AD14
P4.7	K10	45	Вход/выход "порт 4, 7 разряд" Вход/выход "адрес-данные, 15 разряд"	I/O I/O/Z	 AD15
HSI.0	J1	20	Вход "Быстрый ввод, канал 0"	I	
HSI.1	J2	21	Вход "Быстрый ввод, канал 1"	I	
HSI.2	K1	22	Вход "Быстрый ввод, канал 2" Выход "Быстрый вывод, канал 4"	I O	 HSO.4
HSI.3	L2	24	Вход "Быстрый ввод, канал 3" Выход "Быстрый вывод, канал 5"	I O	 HSO.5
HSO.0	K2	25	Выход "Быстрый вывод, канал 0"	O	
HSO.1	L3	26	Выход "Быстрый вывод, канал 1"	O	
HSO.2	K5	31	Выход "Быстрый вывод, канал 2"	O	
HSO.3	L6	34	Выход "Быстрый вывод, канал 3"	O	
EA#	B6	79	Вход "Внешний доступ"	I	
BW	B9	71	Вход "Разрядность внешней шины"	I	

Окончание таблицы 1.3.3

1	2	3	4	5	6
READY	L10	43	Вход "Готовность"	I	
NMI	A5	80	Вход "Немаскируемое прерывание"	I	
RESET#	E1	9	Вход/выход "Сброс"	I/O	
INST	A9	70	Выход "Чтение команды"	O	
ALE	B10	69	Выход "Разрешение записи адреса" Выход "Адрес действителен"	O O	 ADV#
WR#	K8	40	Выход "Запись" Выход "Запись младшего байта"	O O	 WRL#
BHE#	L9	41	Выход "Разрешение записи старшего байта" Выход "Запись старшего байта"	O O	 WRH#
RD#	A10	68	Выход "Чтение"	O	
CLCOUT	A8	72	Выход "Системный тактовый сигнал"	O	
XTAL1	A7	74	Вывод подключения кварцевого резонатора. Вход тактового сигнала	– I	
XTAL2	B8	73	Вывод подключения кварцевого резонатора	–	
U_{PP}	L7	37	Вход "Режим пониженного потребления"	I	
CDE	D1	6	Не используется, подключается к общему выводу	–	
U_{CC1} (U_{CC})	A6	8, 33, 60, 77, 78	Питание 5 В цифровой части микросхемы	–	
U_{CC2} (U_{REF})	C2	5	Питание 5 В аналоговой части микросхемы	–	
U_{SS} (GND)	B7, K6	12, 32, 35, 36, 53, 55, 57, 64, 75, 76	Общий вывод цифровой части микросхемы	–	
U_{SS} (ANGND)	C1	3, 4	Общий вывод аналоговой части микросхемы	–	
NC		13, 14, 23, 54, 81, 88	Незадействованные выводы	–	

Примечание – Принятые условные обозначения в графе «Тип вывода»:
- I – вход,
- O – выход,
- I/O – вход/выход,
- Z – третье состояние.

2 Описание устройства и принцип работы

2.1 Блок-схема микроконтроллеров

Микросхемы представляют собой СБИС высокопроизводительных микроконтроллеров с 16-разрядным CPU и ОЗУ на кристалле 232 байт. Регистровая архитектура не нуждается в аккумуляторе, и большинство операций может быть выполнено с регистрами. С помощью регистровых операций можно управлять периферийными блоками, расположенными на кристалле. В состав периферийных блоков входят: последовательный порт, АЦП, ШИМ, высокоскоростная подсистема ввода-вывода, содержащая два 16-битных таймера, 8-уровневое FIFO выборки и 8-канальный программируемый выходной генератор, сторожевой таймер.

Блок-схема микроконтроллеров представлена на рисунке 2.1.1.

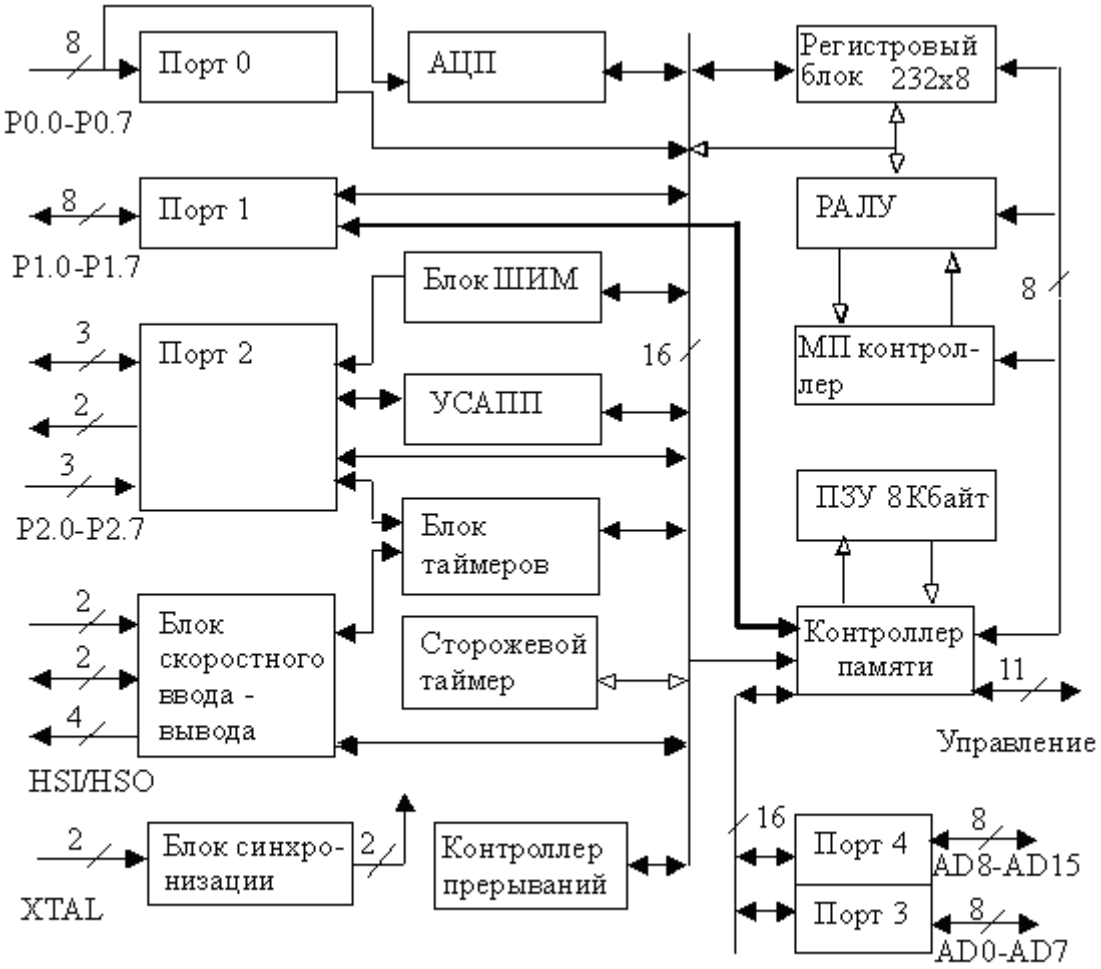


Рисунок 2.1.1 – Блок-схема микроконтроллеров

2.2 Центральный процессор (CPU)

Главными компонентами CPU являются регистровый файл и регистровое арифметическое логическое устройство (РАЛУ). Коммуникации с внешними схемами осуществляются посредством регистров специального назначения (РСН) или через контроллер памяти. РАЛУ не использует аккумулятор. Вместо этого оно оперирует напрямую с 256-байтовым регистровым пространством, состоящим из регистрового файла и регистров специального назначения (РСН). Эффективные операции ввода-вывода возможны благодаря непосредственному управлению вводом-выводом через РСН.

Основные преимущества этой структуры:
- возможность быстрых контекстных переключений;
- отсутствие "узкого места" - аккумулятора;
- быстрые операции ввода-вывода.

16-разрядное CPU соединено с контроллером прерываний и контроллером памяти посредством 16-разрядной шины. Дополнительно имеется 8-разрядная шина, которая передает байты команды из контроллера памяти в CPU, кроме того 16-разрядная шина соединяет CPU с периферийными блоками.

2.3 Контроллер памяти

РАЛУ обменивается данными с внешней памятью через контроллер памяти; в это адресное пространство не входят регистровый файл и регистры специального назначения (РСН). Контроллер памяти содержит контроллер шины, устройство четырехбайтовой очереди команд и подчиненный программный счетчик (Slave PC). Обе внутренние шины ПЗУ и внешней памяти управляются контроллером шины. Запрос доступа к памяти в контроллер шины может придти от РАЛУ или от очереди команд, причем очередь имеет высший приоритет. Запросы из очереди инструкций производятся по адресу из подчиненного программного счетчика. Программный поток (поток команд, программа) считывается из памяти по адресу подчиненного программного счетчика; процессор экономит время, так как адреса редко посылаются в контроллер памяти (перезаписываются). Если адресная последовательность изменяется в результате JUMP, прерываний, CALL (вызова) или возврата из подпрограммы, подчиненный программный счетчик загружается новым значением, очередь заполняется и процесс продолжается. Скорость работы возрастает при использовании очереди, так как обычно сохраняет следующий байт команды доступным. Временные диаграммы выполнения команд приведены без добавления времени ожидания и при выбранной 16-разрядной шине. Перезагрузка подчиненного программного счетчика и запись первого байта нового потока инструкций занимает 4 такта CLKOUT. Это происходит при JUMP (командах перехода).

Когда для отладки кодов используется логический анализатор, нужно представлять, что не возможно определить, когда команда начинает работать после загрузки, так как очередь заполняется до начала исполнения команды.

2.4 Управление CPU

Управление CPU блоком последовательности микрокодов допускает исполнение операций с байтом, словом или двойным словом в 256-битовым регистровом пространстве. Команды для CPU поступают из очереди и временно запоминаются в регистре команд. Блок последовательности микрокодов декодирует команды и генерирует необходимую последовательность событий (сигналов, микрокодов), необходимые для исполнения требуемой функции. На рисунке 2.4.1 приведена блок-схема контроллера памяти, РАЛУ, регистр команд и блок генератора микрокода.

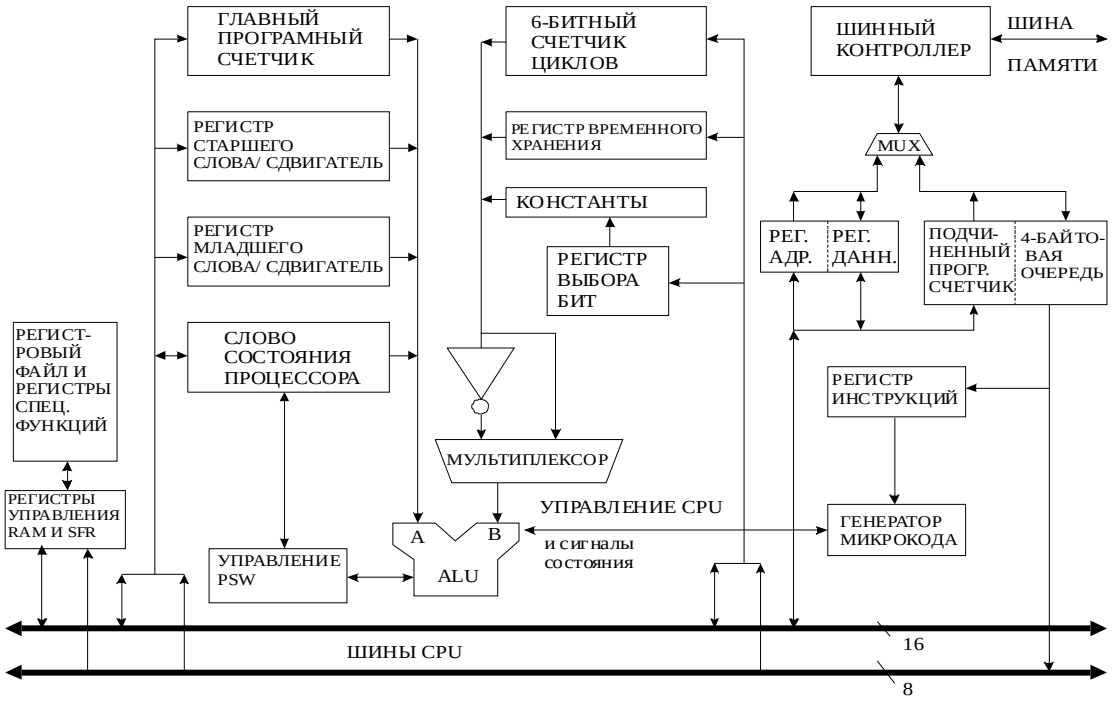


Рисунок 2.4.1 – Блок-схема РАЛУ

2.5 Регистровое АЛУ (РАЛУ)

Главным образом вычисления в микросхемах происходят в РАЛУ. РАЛУ, показанное на рисунке 2.4.1, содержит 17-битное РАЛУ, регистр слова состояния процессора (PSW), программный счетчик, счетчик циклов и три регистра временного хранения. Все регистры 16-разрядные или 17-разрядные (16-разрядов + расширение знаком). Некоторые из регистров способны выполнять простые операции для разгрузки РАЛУ. Отдельный инкремент используется для PC, в то время, когда ему доступны операнды. Однако изменения содержимого PC при JUMP, вызовах, возвратах и прерываниях происходят через РАЛУ. Два временных регистра имеют свою собственную логику сдвигов. Эти регистры используются для операций, требующих логических сдвигов, включая нормализацию, умножение и деление. Регистр "младшего слова" и регистр "старшего слова" используются вместе для 32-битных операций и как временные регистры для большинства команд. Повторные сдвиги управляются 6-битным "счетчиком циклов". Третий временной регистр запоминает второй операнд двухоперандных команд. Он содержит множитель при умножении и делитель при делении. При выполнении вычитания выходные данные этого регистра дополняются до двух перед тем как данные попадают на вход "B" РАЛУ. Несколько констант, такие как 0, 1 и 2, записаны в РАЛУ для ускорения вычислений (создание дополнения числа до двух и выполнение инкремента или декремента). Дополнительно однобитовые маски для команд проверки битов генерируются в регистре констант, основанном на трехбитном регистре выбора бита.

2.6 Внутренние временные соотношения

Микросхемам для функционирования необходим тактовый сигнал на входе XTAL1. XTAL1 и XTAL2, являясь входом и выходом инвертора на кристалле, могут быть использованы для генерации тактового сигнала.

Внутренние операции ИС синхронизируются частотой от внутреннего или внешнего генератора, деленной на 2. Каждые два периода частоты составляют одно "State time", основное время измерения для всех операций ИС при тактовой частоте 20 МГц "State time" (машинный цикл) составляет 100 нс.

Так как микросхемы могут работать на разных частотах, времена даются в "State time", если по-другому не определено. Выходной системный тактовый сигнал, показанный на рисунке 2.6.1, формируется для прослеживания внутреннего машинного цикла при f_C = 20 МГц, t_{CO} = 100 нс.

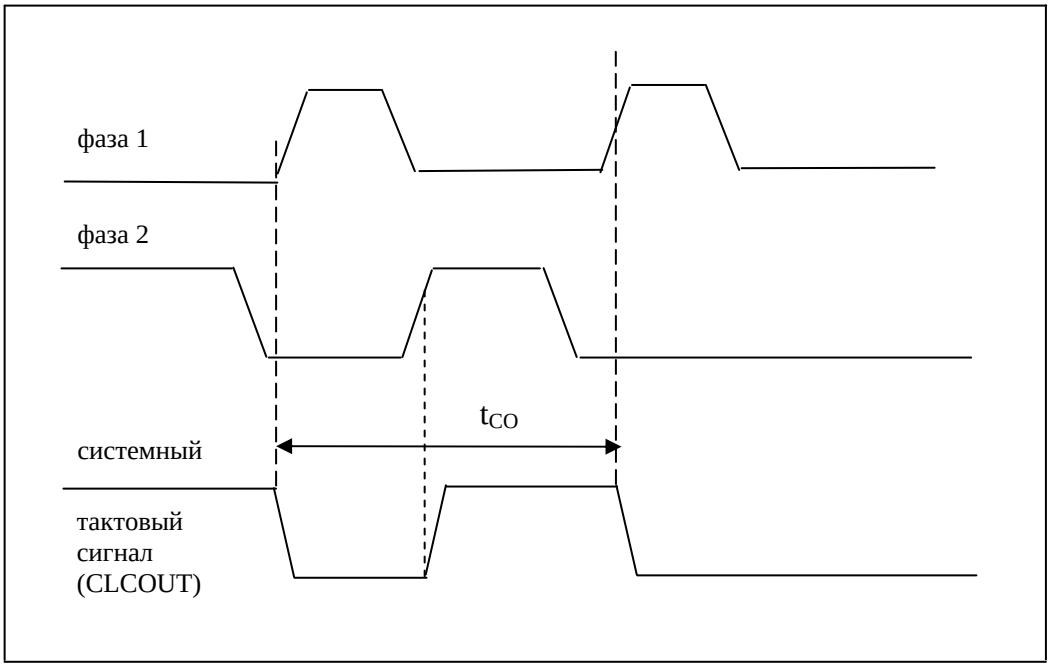


Рисунок 2.6.1 – Внутренние временные соотношения

2.7 Адресное пространство памяти

Адресное пространство микросхем составляет 64 Кбайт, большая часть которого доступна как память программ или память данных. Адреса с 0000H до 00FFH и с 1FFEH до 2080H предназначены для специальных целей. Все другие адреса могут быть использованы для запоминания любых программ или данных или карты памяти периферии. Карта памяти показана на рисунке 2.7.1.

Внешняя память или устройства ввода-вывода	0FFFFH
Внутреннее ПЗУ или внешняя память	4000H
Резерв	2080H
Старшие 8 векторов прерываний	2040H
Ключ (шифр) доступа ПЗУ	2030H
Резерв	2020H
Байт конфигурации кристалла	2019H
Резерв	2018H
8 младших векторов прерываний + 2 специальных прерывания	2014H
Порты 3 и 4	2000H
Внешняя память или внешние устройства ввода-вывода	1FFEH
Память внутренних данных (регистровый файл, регистры специального назначения, указатель стека)	0100H 0000H

Рисунок 2.7.1 – Карта памяти ИС

2.7.1 Регистровый файл

Адреса с 0000H по 00FFH содержат файл регистров и регистры специального назначения. РАЛУ может оперировать с любым из этих 256 внутренних регистров, однако коды (команд) не могут выполняться из этих регистров. Если сделана попытка запустить инструкции из адресов с 000H по 0FFH, инструкции будут запрашиваться из внешней памяти. Эта секция внешней памяти зарезервирована.

Внутреннее ОЗУ по адресам с 18H до 0FFH содержат регистровый файл. Он содержит 232 байта ОЗУ с возможностью доступа по байтам (8 бит), словам (16 бит) или двойным словам (32 бита). Каждый из этих регистров может быть использован РАЛУ как один из 232 "аккумуляторов". Эта область памяти, как источник данного состояния микросхем, сохраняется нетронутой, пока микросхемы находятся в режиме сохранения мощности (Power down mode).

Ячейки с адресами 18H и 19H составляют указатель стека. Эти ячейки не являются регистрами специального назначения и могут быть использованы как стандартные ячейки ОЗУ, если стековые операции не производятся. РАЛУ может с легкостью оперировать ими. Указатель стека должен быть инициализирован пользовательской программой и может указывать на любое место в 64-Кбайтном адресном пространстве. Работа стека происходит с загрузкой вниз, таким образом, указатель стека должен быть инициализирован на два байта выше верхних стековых адресов и должен быть выровнен по адресам слов.

2.7.2 Регистры специального назначения

Регистры с адресами от 00H до 17H являются регистрами управления вводом-выводом или, по-иному, регистрами специального назначения. Все периферийные устройства ИС (исключая порты 3 и 4) управляются через эти регистры. Как показано на рисунке 2.7.2.1, регистры специального назначения (РСН) микросхем расположены в трех окнах.

Функции РСН представлены в таблице 2.7.2.2, альтернативные функции РСН в окне 15 – в таблице 2.7.2.3.

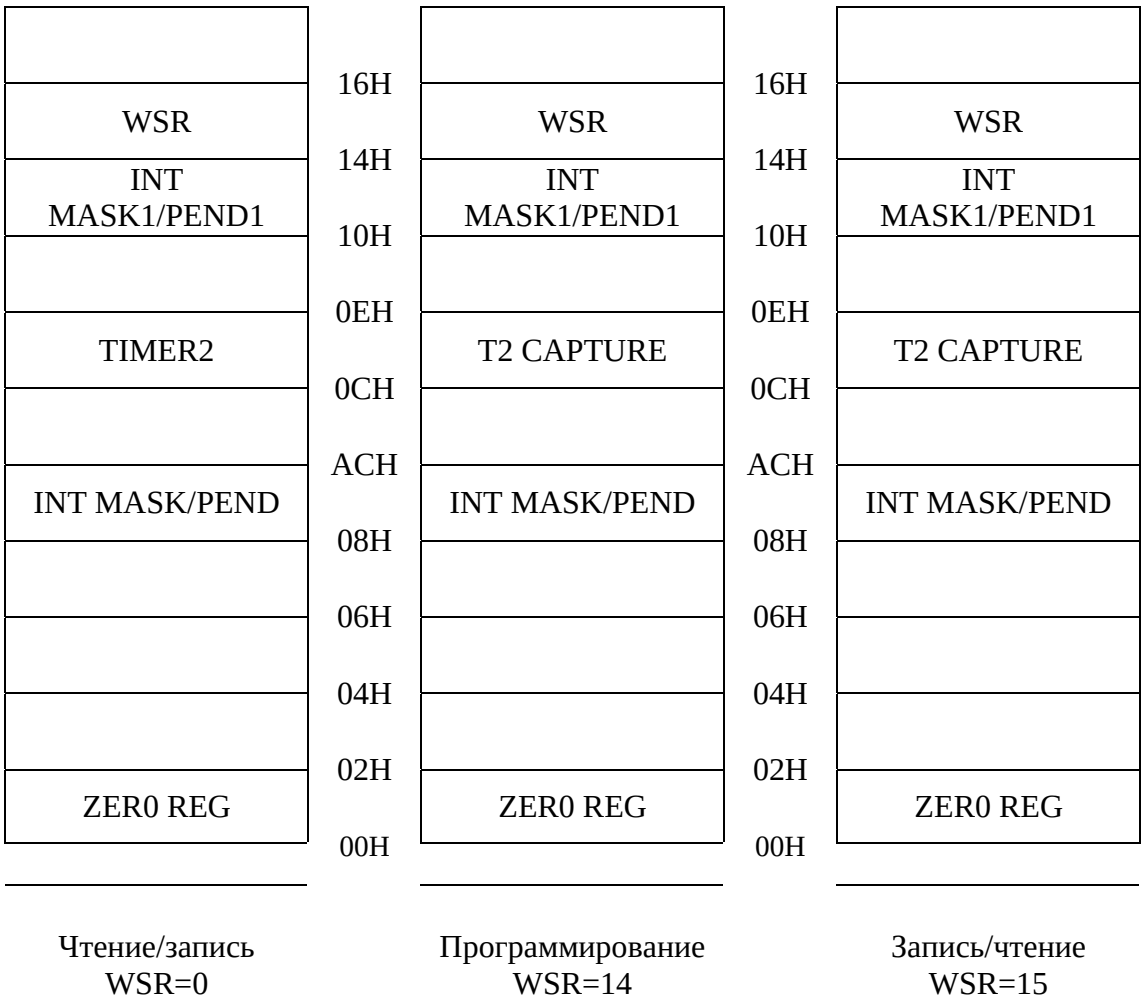


Рисунок 2.7.2.1 - Регистровые окна

Переключения между окнами осуществляются Регистром Выбора Окна (WSR) с адресом 14H для всех окон. Команды PUSHA и POPA загружают и выгружают WSR, таким образом легко сделать выбор между окнами (выбрать одно из трех окон). Только три значения можно записывать в WSR - 0, 14 и 15. Другие значения зарезервированы в будущих реализациях и могут вызывать непредсказуемые действия.

Регистровое окно 0 выбирается при WSR = 0. Как указано в таблице 2.7.2.2, оно (окно) имеет 24 регистра, которые имеют функции, отличающиеся при записи и чтении, т. е. один и тот же регистр может иметь разные функции при записи и чтении.

Управление программированием и тестовые операции - в регистровом окне 14. Регистры, которые не отмечены в этом окне должны быть закрыты от любого чтения или записи. В регистрах окна 15 (WSR = 15) работа PCH изменена. Регистры, которые читаются только в окне 0 записываются только в регистровом окне 15 и наоборот. Причем адреса не меняются. Можно менять окна, не меняя адреса и производить чтение или запись. Как главное исключение - это регистр Таймер 2, он читается и записывается в регистровом окне 0, а в окне 15 по тому же адресу находится регистр захвата Таймера 2, он читается и записывается. Регистры, которые могут читаться и записываться в окне 0 также читаются и записываются в регистровом окне 15, см. таблицу 2.7.2.3.

Таблица 2.7.2.1 – Регистры специального назначения

Указатель стека (SP)	19H	Указатель стека (SP)
IOS2	18H	PWM_CONTROL
IOS1	17H	IOC1
IOS0	16H	IOC0
WSR	15H	WSR
INT_MASK1	14H	INT_MASK1
INT_PEND1	13H	INT_PEND1
SP_STAT	12H	SP_CON
PORT2	11H	PORT2
PORT1	10H	PORT1
PORT0	0FH	BAUD RATE
TIMER2 (HI)	0EH	TIMER2 (HI)
TIMER2 (LO)	0DH	TIMER2 (LO)
TIMER1 (HI)	0CH	IOC2
TIMER1 (LO)	0BH	WATCHDOG
INT_PEND	0AH	INT_PEND
INT_MASK	09H	INT_MASK
SBUF(RX)	08H	SBUF (TX)
HSI_STATUS	07H	HSO_COMMAND
HSI_TIME (HI)	06H	HSO_TIME (HI)
HSI_TIME (LO)	05H	HSO_TIME (LO)
AD_RESULT (HI)	04H	HSI_MODE
AD_RESULT (LO)	03H	AD_COMMAND
ZER0 REG (HI)	02H	ZER0 REG (HI)
ZER0 REG (LO)	01H	ZER0 REG (LO)
	00H	
чтение	WSR=0	запись

РЕЗЕРВ*	10H	
РЕЗЕРВ*	0FH	
РЕЗЕРВ*	0EH	
T2CAPTURE (HI)	0DH	
T2CAPTURE (LO)	0CH	

WSR = 15

Другие SFR в WSR = 15 могут быть считаны, если они были записаны в WSR = 0 и могут быть записаны, если в WSR = 0 они считываются

PPW	04H

WSR = 14

Резервные регистры не записываются и не читаются

Таблица 2.7.2.2 – Функции регистров специального назначения

Регистр	Описание
ZER0 REG	Нулевой регистр всегда считывается как ноль, используется как базовый для индексной адресации и как константа для вычислений и сравнений.
AD_RESULT	АЦП результат (LO/HI) – младший или старший разряды результата преобразования.
AD_COMMAND	АЦП командный регистр – управляет АЦП.
HSI_MODE	Регистр режима HSI – устанавливает режим высокоскоростного ввода.
HSI_TIME	Время HSI старший/младший – содержит время, в течение которого высокоскоростной ввод был включен.
HSO_TIME	Время HSO старший/младший – устанавливает время или счет циклов для высокоскоростного вывода при исполнении команды из командного регистра.
HSO_COMMAND	Командный регистр HSO – определяет, что будет происходить во время, указанное в регистре HSO_TIME.
HSI_STATUS	Регистр состояния HSI – указывает, какой из выводов HSI был переведен в режим HSI, текущее состояние выводов. В окне 15 записываются проверяемые разряды, но не их текущее состояние.
SBUF (TX)	Буфер передачи для последовательного порта – содержит данные для вывода. Последний записанный результат может быть считан в окне 15.
SBUF (RX)	Буфер приема для последовательного порта – сохраняет байт, принятый в последовательный порт. Может быть записан в окне 15.
INT_MASK	Регистр маскирования прерываний – разрешает или запрещает индивидуальные прерывания.
INT_PEND	Регистр ожидания прерываний – указывает, какой сигнал прерывания пришел от одного из источников и не был обслужен (только INT_PENDING).
WATCHDOG	Регистр сторожевого таймера – вызывает автоматический периодический сброс каждые 64K машинных цикла. Возобновляет старший байт счетчика WDT в окне 15.
TIMER1	Таймер 1 старший/младший – старшие/младшие байты таймера 1.
TIMER2	Таймер 2 старший/младший – старшие/младшие байты таймера 2.
IOPORT0	Регистр порта 0 – уровни на выводах порта 0. Резервируются в окне 15.
BAUD_RATE	Регистр, определяющий скорость передачи, загружается периодически. Резервируется в окне 15.
IOPORT1	Регистр порта 1 – используется для чтения/записи в порт 1. Зарезервирован в окне 15.
IOPORT2	Регистр порта 2 – используется для чтения/записи в порт 2. Зарезервирован в окне 15.
SP_STAT	Состояние последовательного порта – указывает состояние последовательного порта.
SP_CON	Управление последовательного порта – используется для установки режима работы последовательного порта.
IOS0	Регистр 0 состояния ввода-вывода – содержит информацию о состоянии HSO. Записывается на выход HSO в окне 15.
IOS1	Регистр 1 состояния ввода-вывода – содержит информацию о состоянии таймеров и HSI.
IOC0	Регистр 0 управления ввода-вывода – управляет альтернативными функциями выводов HSI, источником сброса и синхронизации таймера 2.
IOC1	Регистр 1 управления ввода-вывода – управляет альтернативными функциями порта 2, прерываниями от таймера и прерываниями от HSI.
PWM_CONTROL	Регистр управления ШИМ – управляет длительностью импульсов ШИМ.
INT_PEND1	Регистр ожидания прерываний для 8 векторов прерываний (только INT_PENDING1).
INT_MASK1	Регистр маски прерываний для 8 векторов прерываний.
IOC2	Регистр 2 управления ввода-вывода – управляет функциями таймера 2, АЦП и HSO.
IOS2	Регистр 2 состояния ввода-вывода – содержит информацию о событиях HSO.
WSR	Регистр выбора окна – выбирает регистровое окно.

Таблица 2.7.2.3 – Альтернативные функции PCH в окне 15

Функция	Описание
AD_COMMAND (02H)	Считывание последней записанной команды
AD_RESULT (02H, 03H)	Запись результата в регистр результата АЦП
HSI_MODE (03H)	Чтение значения в HSI_MODE
HSI_TIME (04H, 05H)	Запись в регистр хранения FIFO
HSO_TIME (04H, 05H)	Чтение последнего значения в регистре хранения
HSI_STATUS (06H)	Запись типа события на выводах HSI (разряды HSI_STATUS 0, 2, 4, 6)
HSO_COMMAND (06H)	Чтение последнего значения в регистре хранения
SBUF(RX) (07H)	Запись значения в буфер приема
SBUF(TX) (07H)	Чтение последнего значения из буфера передачи
WATCHDOG (0AH)	Чтение значения старшего байта WDT
TIMER1 (0AH, 0BH)	Запись значения в Timer1
TIMER2 (0CH, 0DH)	Чтение/запись регистра захвата Timer2 (чтение/запись Timer2 при WSR=0)
IOC2* (0BH)	Последняя записанная величина, доступная чтению, исключается бит 7
BAUD_RATE (0EH)	Не может быть считан
PORT0 (0EH)	Нет доступа
SP_STAT (11H)	Установка битов состояния, могут быть установлены биты TI и RI, но они не могут вызывать прерывания
SP_CON (11H)	Чтение текущего байта управления
IOS0 (15H)	Запись в регистр управления выводами HSO. Разряды 6 и 7 недоступны для записи
IOC0* (15H)	Последнее записанное значение разрешено для чтения, за исключением бита 1
IOS1 (16H)	Запись в этот регистр будет устанавливать биты состояния, но не вызывает прерываний, биты 6 и 7 доступны для записи
IOC1 (16H)	Последнее записанное значение доступно для чтения
IOS2 (17H)	Запись в этот регистр устанавливает биты состояния, но не вызывает прерываний
PWM_CONTROL (17H)	Чтение значения рабочего цикла записанного в PWM_CONTROL

_____

* IOC2.7 (CAM CLEAR) и IOC0.1 (T2RST) не защелкиваются и будут читаться как единица (CAM – ассоциативная память модуля HSO).

Внутри пространства РСН несколько регистров и битов адресов помечены "Зарезервировано". В эти ячейки нельзя производить чтение и запись. В зарезервированные биты всегда записан 0 для совместимости с будущими моделями ИС. Значения, читаемые из этих ячеек могут изменяться от микросхемы к микросхеме или от температуры и напряжения. С регистрами и битами, которые не помечены, нужно обращаться как с резервными регистрами и битами. Заметим, что состояние (по умолчанию) внутреннего регистра – 0, тогда как для внешней памяти – это единичный уровень. Это обусловлено тем, что РСН функции обычно запрещены нулевым значением, тогда как стирание памяти – это перевод в единичное состояние. Необходимо соблюдать предосторожность при использовании РСН, как источника для операций или в качестве базовых или индексных режимов для косвенной или индексной адресации. Возможен нежелательный результат, так как внешние события могут изменить РСН или РСН очистится при чтении. Возможность для РСН по изменению содержимого должна быть учтена при работе с этими регистрами. Это особенно важно, когда используются языки высокого уровня, так как они не всегда учитывают все особенности работы РСН регистров. РСН могут работать с байтами или словами, если иначе не определено.

2.7.3 Зарезервированные области памяти

Адреса 1FFEH и 1FFFH используются для порта 3 и 4, соответственно, позволяя легко перенастраивать эти порты при работе с внешней памятью. Если порты 3 и 4 не надо перенастраивать и внутренняя память не используется, с этими ячейками можно работать как с любыми ячейками внешней памяти. Многие резервные и специальные регистры имеют адреса между 2000H и 2080H. В этой области размещены 18 векторов прерывания, байт конфигурации кристалла и ключ секретности доступа. На рисунке 2.7.3.1 показаны адреса и функции этих регистров. Регистры прерываний, конфигурации кристалла, ключа секретности доступа описаны в разделах 5, 16. С одним исключением, все не определенные по функциям адреса в промежутке с 2000H до 207FH, включая "резервные", зарезервированы для использования при тестировании и для будущих разработок. Они должны быть заполнены шестнадцатеричным значением FFH для обеспечения совместимости с будущими устройствами. Адрес 2019H может содержать 20H для предотвращения возможных конфликтов шины в течение цикла записи ССВ (байт конфигурации кристалла).

Примечание – Эти ситуации возможны только для систем с 16-битной шиной и внешней программной памятью.

"Сброс" ИС вызывает чтение потока команд с адреса 2080H. Этот адрес был выбран для обеспечения возможности системы иметь ОЗУ до 8 Кбайт, являющейся продолжением регистрового файла.

	FFFFH
Внутренняя память или устройства ввода/вывода	4000H
Внутреннее ПЗУ или внешняя память	2080H
Резерв	2074H-207FH
Уровни напряжения	2072H-2073H
Слово сигнатуры	2070H-2071H
Резерв	2040H-206FH
Векторы прерываний	2030H-203FH
Ключ секретности	2020H-202FH
Резерв	2019H-201FH
Байт конфигурации кристалла	2018H
Резерв	2015H-2017H
PPW (регистр длительности импульса программирования, для ИС с EPROM)	2014H
Векторы прерываний	2000H-2013H

Рисунок 2.7.3.1 – Резервное пространство памяти

2.7.4 Внутреннее ПЗУ

При использовании внутреннего ПЗУ программист может определить программу пользователя в области внутренней программной памяти с адресами от 2080H до 3FFFH, а также векторы прерываний, регистр конфигурации кристалла (CCR), ключ секретности в диапазоне адресов от 2000H до 207FH. По адресу 2014H содержится регистр программирования Ширины Импульса (PPW). Этот регистр используется в ИС с СППЗУ и зарезервирован в ИС с масочным ПЗУ.

Инструкции (команды) и данные считываются из внутреннего ПЗУ, если вход EA# установлен в высокий уровень и адрес находится между 2000H и 3FFFH. Во всех других случаях данные считываются из внутреннего ОЗУ данных или внешней памяти и инструкции считываются из внешней памяти. Состояние вывода EA# захватывается по фронту нарастания RESET#. ИС имеют биты защиты ПЗУ, позволяющие программе блокировать чтение внутренней программной памяти. В порядке установления секретности команды нельзя исполнять из последних адресов внутреннего ПЗУ, если биты защиты установлены.

2.7.5 Системная шина

Имеется несколько режимов работы ИС с системной шиной: стандартный режим – 16-битная мультиплексированная шина адрес/данные, 8-битовый режим и режим, когда шина может динамически переключаться между 8-битной и 16-битной. Сигналы HOLD#/HLDA# и READY позволяют создавать разнообразные системы памяти. Сигнал READY необходим для расширения импульсов RD# и WR#, чтобы осуществить доступ к медленной памяти.

3 Программные средства

Система команд ИС содержит полный набор арифметических и логических действий над 8-битными операндами типа BYTE и SHORT-INTEGER и над 16-битными операндами типа WORD и INTEGER. Типы данных DOUBLE WORD и LONG-INTEGER поддерживаются для умножения 16×16, для деления 32 на 16, для операций сдвига и 32-разрядного сравнения. Недостающие операции над 32-битными операндами могут быть выполнены как комбинации над 16-битными операндами.

В дополнение к действиям над различными типами данных поддерживается преобразование между этими типами.

Инструкции микроконтроллеров для сложения, вычитания и сравнения не отличаются для беззнаковых слов и знаковых целых. Условные переходы обеспечивают обработку результатов этих операций как знаковых, так и без знаковых величин.

В данном разделе короткие фрагменты программ будут использоваться для иллюстрации работы. В этих примерах будут использоваться символические имена регистров временного хранения:

AX, BX, CX, DX – 16-битные регистры;
AL – младший байт AX;
AH – старший байт AX;
BL – младший байт BX;
CL – младший байт CX;
DL – младший байт DX.

3.1 Типы операндов

Архитектура ИС обеспечивает поддержку различных типов данных, которые удобны для использования в управляющих системах. При рассмотрении этих типов операндов их имена будут заимствованы из языка PL/M-96. Во избежание разночтений имя типа операнда будем обозначать большими буквами: "BYTE" – это беззнаковая 8-битная переменная; "byte" – восемь бит данных любого типа.

БАЙТЫ (BYTES)

BYTES – беззнаковые 8-битные переменные в диапазоне от 0 до 255. Арифметические и родственные им операции могут использовать операнды типа BYTES, но результат должен быть интерпретирован по модулю 256. Логические операции на операндах типа BYTES работают с отдельными битами. Биты в BYTE-операнде маркированы от 0 до 7, причем бит 0 – младший значащий. Ограничений по размещению для операндов типа BYTE нет; они могут располагаться в любом месте адресного пространства.

СЛОВА (WORDS)

WORDS – беззнаковая 16-битная переменная, которая может иметь значения 0...65535. Арифметические и родственные им операции могут быть применены к операндам типа WORD, но результат должен быть интерпретирован по модулю 65536. Логические операции манипулируют с отдельными битами. Биты пронумерованы с 0 до 15, причем 0 бит – младший значащий. Слова должны быть выровнены по границам четных адресов в адресном пространстве. Младший байт должен находиться по четному адресу, а старший байт – по следующему (нечетному) адресу. Адрес слова – это адрес его младшего байта. Операции со словами, имеющими нечетный адрес, не гарантированны к выполнению.

КОРОТКИЕ ЦЕЛЫЕ (SHORT-INTEGERS)

SHORT-INTEGERS – 8-битная знаковая переменная в диапазоне –128...+127. Арифметические операции, вырабатывающие результаты вне этого диапазона, будут устанавливать флаг переполнения в слове состояния программы (PSW). Действительный числовой результат будет тот же, что и для тех же операций с переменными типа BYTE. SHORT-INTEGERS могут быть размещены в любом месте адресного пространства.

ЦЕЛЫЕ (INTEGERS)

INTEGERS – 16-битные знаковые переменные в диапазоне –32768...+32767. Арифметические операции, вырабатывающие результаты вне этого диапазона, будут устанавливать флаг переполнения в PSW. Действительный числовой результат будет тот же, что для тех же операций с переменными типа WORD. Для INTEGERS существуют те же ограничения по выравниванию и те же правила адресации, что и для WORDS.

БИТЫ (BITS)

BITS – однобитные операнды, которые могут принимать булевы значения "истина" и "ложь". В дополнение к обычной поддержке битовых операндов как компонентов BYTE или WORD, ИС обеспечивают непосредственную проверку любого бита во встроенном регистровом файле. Архитектура микросхем требует, чтобы биты были адресованы как компоненты BYTES или WORDS, она не поддерживает прямую адресацию битов, как у 1830ВЕ51/31.

ДВОЙНЫЕ СЛОВА (DOUBLE-WORDS)

DOUBLE-WORDS – это беззнаковые 32-битные переменные, которые могут принимать значения 0...4294967295. Архитектура ИС непосредственно поддерживает такие операнды для операций сдвига, как делимое при делении на 16-битный операнд, как результат умножения двух 16-битных операндов. Для этих операций переменная типа DOUBLE-WORD должна находиться в регистровом файле и должна быть выровнена по адресу, который без остатка делится на 4. Операнд DOUBLE-WORD адресуется адресом своего младшего байта. Операции с DOUBLE-WORD, которые непосредственно не поддерживаются, могут быть выполнены посредством двух операций с операндами типа WORD. Для совместимости с программным обеспечением пользователь должен выполнять требования по адресации операндов DOUBLE-WORDS.

ДЛИННЫЕ ЦЕЛЫЕ (LONG-INTEGERS)

LONG-INTEGERS – 32-битные переменные в диапазоне –2147483648...+2147483647. Архитектура ИС поддерживает такие операнды непосредственно только для операций сдвига, как делимое при делении на 16-битный операнд, как результат умножения 16-битных операндов, при операциях сравнения двойных слов.

LONG-INTEGERS переменные могут быть также нормализованы. Для этих операций переменная типа LONG-INTEGERS должна находиться в регистровом файле и должна быть выровнена по адресу, который без остатка делится на 4. Переменная типа LONG-INTEGER адресуется адресом ее младшего байта.

Операции с LONG-INTEGERS, которые непосредственно не поддерживаются, могут быть выполнены посредством двух операций с операндами типа INTEGER. Для совместимости с программным обеспечением пользователь должен выполнять требования по адресации LONG операндов.

3.2 Адресация операндов

Доступ к операндам в адресном пространстве ИС возможен посредством шести основных режимов адресации. Данный раздел описывает аппаратную поддержку этих режимов адресации.

Шесть базовых режимов адресации, которые будут описаны, называются: прямая регистровая, косвенная, косвенная с автоинкрементированием, непосредственная, короткая индексная, длинная индексная. Несколько других удобных разновидностей адресации могут быть получены путем комбинирования этих базовых режимов адресации со специфическими регистрами, такими как ZERO [0] или указатель стека [SP].

Прямая регистровая адресация

Используется для прямого доступа к регистру из 256-байтного регистрового файла. Регистр выбирается 8-битным полем в инструкции, причем адрес регистра должен соответствовать требованиям по выравниванию в соответствии с типом операнда. В зависимости от инструкции в вычислениях могут участвовать до трех регистров.

Пример:

```
ADD     AX, BX, CX  ;   AX:=BX+CX
MUL     AX, BX      ;   AX:=AX*BX
INCB    CL          ;   CL:=CL+1
```

Косвенная адресация

Используется для доступа к операнду путем размещения его адреса в форме переменной типа WORD в регистровом файле. Записанный адрес должен удовлетворять ограничениям по выравниванию для операнда требуемого типа. Следует отметить, что косвенная адресация может обращаться к операндам в любом месте адресного пространства ИС, включая регистровый файл. Регистр, содержащий косвенный адрес, выбирается 8-битным полем инструкции. Инструкция может содержать только одну косвенную ссылку, остальные операнды в инструкции должны быть прямо адресуемыми.

Пример:

```
LD      AX, [AX]      ;     AX:=MEM_WORD (AX)
ADDB    AL, BL, [CX]  ;     AL:=BL+MEM_BYTE (CX)
POP     [AX]          ;     MEM_WORD (AX):=MEM_WORD(SP); SP:=SP+2
```

Косвенная адресация с автоинкрементированием

Этот режим ничем не отличается от режима обычной косвенной адресации, за исключением того, что переменная типа WORD, содержащая адрес, будет инкрементироваться после того, как будет использована в качестве адреса. При адресации операндов типа BYTES или SHORT-INTEGERS, инкрементирование будет происходить на 1; при адресации операндов типа WORDS или INTEGERS инкрементирование будет производиться на 2.

Пример:

```
LD          AX, [BX]+     ;     AX:=MEM_WORD (BX); BX:=BX+2
ADDB        AL, BL, [CX]+ ;     AL:=BL+MEM_BYTE (CX); CX:=CX+1
PUSH        [AX]+         ;     SP:=SP-2
                          ;     MEM_WORD (SP):=MEM_WORD (AX)
                          ;     AX:=AX+2
```

Непосредственная адресация

Этот режим адресации позволяет использовать операнд из поля инструкции. При операциях с операндами типа BYTE или SHORT-INTEGER это поле имеет размер 8 бит; при операциях с операндами типа WORD или INTEGER это поле имеет размер 16 бит. Инструкция может иметь только одну непосредственную ссылку, остальные операнды должны быть прямо адресуемыми.

Пример:

```
ADD         AX, #340      ;     AX:=AX+340
PUSH        #1234H        ;     SP:=SP-2; MEM_WORD (SP)=1234H
DIVB        AX, #10       ;     AL:=AX/10; AH:=AX MOD 10
```

Короткая индексная адресация

При этом режиме адресации 8-битовое поле в инструкции выбирает переменную типа WORD в регистровом файле, которая должна содержать адрес. Второе 8-битовое поле в инструкции воспринимается как константа со знаком и суммируется с переменной типа WORD для формирования адреса требуемого операнда. Поскольку это поле имеет знак, адрес операнда может находиться в пределах -128...+127 байт от адреса, определенного переменной типа WORD. Инструкция может иметь только одну ссылку с укороченным индексированием, остальные операнды должны быть прямоадресуемыми.

Пример:

```
LD          AX, 12[BX]    ;     AX:=MEM_WORD (BX+12)
MULB        AX, BL, 3[CX];      AX:=BL*MEM_BYTE (CX+3)
```

Длинная индексная адресация

Этот режим адресации такой же, как короткая индексная адресация за исключением того, что поле индекса 16-разрядное. Инструкция может иметь только одну ссылку с длинной индексной адресацией, остальные операнды должны быть прямо адресуемыми.
Пример:

```
AND     AX, BX, TABLE[CX]       ; AX:=BX.AND.MEM_WORD(TABLE+CX)
ST      AX, TABLE[BX]           ;  MEM_WORD(TABLE+BX)=:AX
ADDB    AL, BL, LOOKUP[CX]      ;  AL:=BL+MEM_BYTE(LOOKUP+CX)
```

Адресация регистром ZERO

Первые два байта в регистровом файле содержат фиксированное значение 0. В дополнение к использованию в качестве константы при вычислениях и сравнениях, этот регистр может использоваться как переменная типа WORD при длинной индексной адресации. Эта комбинация режима адресации и регистра позволяет прямо адресовать любую ячейку памяти.
Пример:

```
ADD     AX, 1234[0]  ;      AX:=AX+MEM_WORD(1234)
POP     5678[0]      ;      MEM_WORD(5678):=MEM_WORD(SP)
                     ;      SP:=SP+2
```

Адресация указателем стека (SP)

Указатель стека может быть доступен как регистр 18H во внутреннем регистровом файле. В дополнение к обычным операциям SP, он также поддерживает доступ к операндам в стеке. Например, вершина стека может быть доступна через SP как переменная типа WORD при использовании косвенной адресации. Или указатель стека может быть использован при короткой индексной адресации для доступа к данным, находящимся в стеке.
Пример:

```
PUSH    [SP]         ;      DUPLICATE TOP_OF STACK
LD      AX, 2[SP]    ;      AX:=NEXT_TO_TOP
```

3.3 Слово состояния программы

Слово состояния программы (PSW) - это набор логических флагов, сохраняющих информацию состояния работы пользовательских программ. В PSW имеется два байта: байт текущего слова состояния и младший байт масок прерывания. Рисунок 3.3.1 показывает биты состояния PSW. PSW может быть сохранен в системном стеке командой PUSHF. Только биты прерываний в PSW доступны для прямой адресации.

7	6	5	4	3	2	1	0
Z	N	V	VT	C	X	I	ST

Рисунок 3.3.1 - PSW регистр

Z: Флаг Z (ZERO) индицирует, что результатом операции является ноль. Для инструкций ADDC и SUBC флаг Z очищается, если результат не ноль, но никогда не устанавливается. Эти две инструкции обычно используются вместе с ADD и SUB для исполнения арифметических операций с повышенной точностью. Такая работа флага Z с инструкциями позволяет ему правильно индицировать результат для всех вычислений повышенной точности.

N: Флаг N (Negative) устанавливается для индикации отрицательного результата операции. Следует отметить, что флаг N устанавливается в алгебраическом корректном состоянии даже и при переполнении. Для сдвиговых операций, включая операции нормализации и всех трех видов сдвигов командами SHL, SHR, SHRA, N-флаг принимает значение старшего бита результата. Это верно и при величине сдвига равной нулю.

V: Флаг V (oVerflow) устанавливается при операциях с результатом, выходящим за диапазон данных (операнда) назначения. Для команд SHL, SHLB и SHLL флаг V установится, если старший бит операнда изменяется в любое время во время сдвига. Для операции деления следующие условия используются для установки флага V:

Для операций:		V установлено, если частное
UNSIGNED BYTE DIVIDE	>	255 (0FFH)
UNSIGNED WORD DIVIDE	>	65535 (0FFFFH)
SIGNED BYTE DIVIDE	<	- 127 (81H) или > 127 (7FH)
SIGNED WORD DIVIDE	<	- 32767 или > 32767 (7FFFH)

VT: Флаг VT (oVerflow Trap) устанавливается, когда установлен флаг V, но очищается только инструкциями, непосредственно работающими с этим флагом, например, CLRVT или JVT JNVT. Работа с флагом VT позволяет проверить, было ли переполнение в конце последовательности арифметических операций. Это более эффективно, чем проверка флага V после каждой инструкции.

C: Флаг C (Carry) устанавливается в соответствии с арифметическим переносом из старшего бита АЛУ для арифметических операций или в соответствии со значением последнего "выдвинутого" из АЛУ бита в операциях сдвига. Арифметический заем после операции вычитания – это инверсное значение флага C (т.е. если операция сгенерировала заем, флаг C сбрасывается в 0).

X: Зарезервирован. Должен всегда очищаться при записи в PSW для совместимости с будущими версиями ИС.

I: Бит глобального запрета прерываний запрещает все прерывания, когда очищен, исключая команды NMI, TRAP и невыполнимые коды команд.

ST: Флаг ST (STicky bit) устанавливается, когда при сдвиге вправо единица была сдвинута сначала во флаг C, а затем была утеряна ("выдвинута"). Флаг ST не определен после операции умножения. Флаг ST может использоваться вместе с флагом C для управления округлением после сдвига вправо. Рассмотрим умножение 8×8 и затем усечение результата до 12 бит:

```
MULUB    AX,CL,DL   ;   AX:=CL*DL
SHR      AX,#4      ;   Правый сдвиг 4 раза
```

Если флаг C установлен после сдвига, это значит, что биты, выдвинутые вправо, имели значение "больше или равно" по отношению к 1/2 младшего бита конечного результата (LSB). В обратном случае флаг C индицирует, что выдвинутые биты имели значение, меньшее 1/2 LSB. Если флаг ST не использовать, округление производится по значению C флага (результат увеличивают на 1, если C установлен). Флаг ST позволяет получить более точное округление при вычислении.

Правила округления с использованием флага ST

C	ST	Значение выдвинутых битов
0	0	Value =0
0	1	0 < Value <1/2 LSB
1	0	Value = 1/2 LSB
1	1	Value > 1/2 LSB

Неточная оценка погрешности может быть главной причиной ошибок в числовых вычислениях; использование флага ST расширяет возможности программиста.

Младшие 8 бит PSW – индивидуальные маски восьми младших источников прерываний ИС. Эти 8 бит могут быть доступны, как 8 бит байта (INT_MASK – адрес 8H) во внутреннем регистровом файле. Отдельный регистр (INT_MASK1 – адрес 13H) содержит управляющие биты для старших восьми прерываний. Логическая "1" в этих битовых позициях разрешает обслуживание соответствующих прерываний. Бит 9 в PSW запрещает все прерывания. Если этот бит очищен, то все прерывания заблокированы. Заметим, что запросы прерываний фиксируются в регистре INT_PEND, даже если они заблокированы. Исполнение соответствующих обслуживающих программ будут происходить в соответствии с их приоритетами (прерываний) по мере их разрешения. Дальнейшая информация по структуре прерываний ИС – в разделе 5.

3.4 Система команд

Система команд ИС содержит полный набор арифметических и логических действий над 8-битовыми операндами BYTE и SHORT-INTEGER и над 16-битными операндами типа WORD и INTEGER.

Типы данных двойное слово (DOUBLE-WORD) и длинное целое (LONG) (32-битные) поддерживаются для результатов 16×16 умножения и деления 32 на 16, а также для сдвиговых операций и 32-битного сравнения. Остальные операции с 32-битными операндами могут быть выполнены комбинацией 16-битных операций.

Например:
```
ADD     AX, CX
ADDC    BX, DX
```
выполняется 32-битное сложение, а последовательность:
```
SUB     AX, CX
SUB     BX, DX
```
выполняет 32-битное вычитание. Операции с REAL (в операциях с плавающей точкой) не поддерживаются аппаратно, но поддерживаются библиотекой операций с плавающей точкой FPAL-96, имеющей набор команд одинарной точности стандарта IEEE для арифметики с плавающей точкой. Использование этого программного обеспечения существенно облегчено наличием команды NORML (нормализацией 32-битных переменных) и наличием флага ST в PSW. В дополнение к действиям над различными типами данных ИС поддерживают преобразование форматов между ними.

Времена выполнения команд приведены в таблицах 3.5.3. и 3.5.4. Эти времена даны для 16-разрядной шины без задержек. Встроенное ПЗУ работает с 16-битной шиной без задержек. Когда идет работа с 8-битной внешней памятью или добавляются циклы ожидания, CPU также вводит циклы ожидания для предварительной загрузки очереди. Производительность для 8-битной внешней шины сложно точно вычислить, однако можно указать ее в пределах от 10 до 30 % от базовой на выполняемом наборе команд. Лучший способ измерения производительности – прогон реальной программы и измерение с использованием эмулятора или с TIMER1. Времена для команд с косвенной и индексной адресацией приведены для двух областей памяти: РСН/внутреннее ОЗУ (0HFFH) и память, с которой работает контроллер памяти (100H-0FFFFH). Любая инструкция, которая использует любой операнд, который запрашивается посредством контроллера памяти (например, ADD r1, 5000H [0]), длится на 2-3 машинных периода больше, чем при работе с РСН/ОЗУ пространством. Любой доступ к данным во внутреннем ПЗУ происходит через контроллер памяти.

Использование регистров

Архитектура ИС позволяет пользоваться 256-байтным регистровым файлом. Часть из этих регистров используются для управления устройствами ввода/вывода и для других специальных функций выполняемых указателем стека и регистром ZERO. Остальные 230 байт доступны для распределения программистом.

Адресация 32-битных операндов

Эти операнды формируются из двух 16-битных слов в памяти. Младшее значащее слово из двойного слова всегда находится по младшему адресу, даже если данные находятся в стеке (это значит, что старшее значащее слово должно быть загружено в стек первым). Двойное слово имеет адрес своего младшего значащего байта. Следует отметить, что архитектура ИС поддерживает некоторые операции с двойными словами. Для этих операций двойное слово должно находиться во внутреннем регистровом файле и должно иметь адрес, который делится на 4 без остатка.

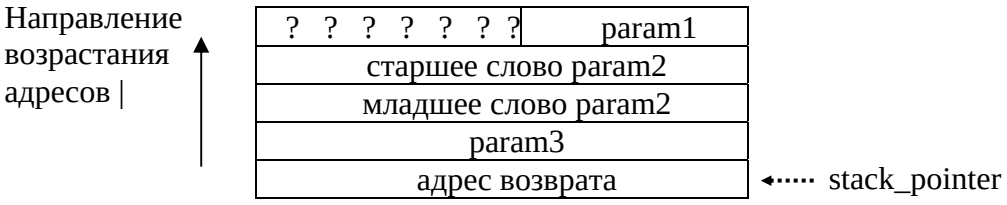
Связь с подпрограммами

Параметры передаются подпрограммам через стек. Параметры записываются в стек в порядке, в котором они перечислены в исходном тексте. Восьмибитные параметры (BYTE или SHORT-INTEGER) заносятся в стек с неопределенным старшим байтом. 32-битные параметры (LONG-INTEGER, DOUBLE-WORD, REAL) заносятся в стек как две 16-битных величины; старшая значащая величина заносится в стек первой.

Рассмотрим, например, следующую процедуру языка PL/M-96:

```
example _ procedure: PROCEDURE
(param1, param2, param3);
DECLARE     param1 BYTE,
            param2 DWORD,
            param3 WORD
```

Когда эта процедура будет вызываться, стек должен содержать параметры в следующем порядке:



Если процедура возвращает значение по команде вызова (в отличие от модифицирования глобальных переменных), тогда результат возвращается в PLMREG. PLMREG затем рассматривается как 8-, 16- или 32-битная переменная в соответствии с типом процедуры.

Стандартный вызов подпрограмм, заимствованный из PL/M-96, должен иметь следующие главные черты.

1 Процедура может в любой момент использовать 8 байт регистрового файла, начиная с PLMREG как ячейки временного хранения данных, находящиеся в теле процедуры.

2 Команда, вызывающая процедуру, должна учитывать, что 8 байт данных в регистровом файле, начиная с PLMREG, модифицируются процедурой.

3 Слово состояния программы не сохраняется и не восстанавливается процедурами, поэтому команда вызова должна учитывать, что флаги состояния (Z, N, V, VT, C, ST) модифицируются процедурой.

4 Результаты выполнения процедуры, всегда возвращаются в PLMREG.

PL/M-96 позволяет описывать процедуры INTERRUPT, которые будут исполняться после обслуживания предыдущего прерывания. Эти процедуры не подчиняются правилам обычных процедур. Параметры не могут передаваться и возвращаться. Поскольку эти процедуры могут выполняться в любой момент времени (отсюда термин прерывание – interrupt), эти процедуры обязаны сохранять состояния PSW и PLMREG при входе в процедуру и восстанавливать их перед выходом из процедуры.

3.5 Защита программного обеспечения

Функциональные возможности ИС помогают предотвратить ошибки программного и аппаратного обеспечения. Защита обеспечивается против исполнения запрещенных кодов (прерывания по запрещенному коду операции). Инструкция RST может вызывать сброс, если программный счетчик выходит за пределы допустимого адресного пространства. Эта (RST) инструкция имеет код 0FFH, поэтому, если процессор читает шинные линии, которые имеют предзаряд высокого уровня, то он самостоятельно производит сброс (RESET). Рекомендуется неиспользованные области памяти инструкций заполнять командами NOP (нет операций) и периодически командами JAMP, чтобы переходить к программам обработки ошибок или к инструкциям RST. Это важная особенность в операциях с просмотровой таблицей, если имеет место нежелаемый результат.

Сторожевой таймер (WDT) – дальнейшая защита от программных и аппаратных ошибок. Когда используется WDT для защиты программного обеспечения (программы) желательно сбрасывать его в одном месте в программе, уменьшая этим возможность нежелательного сброса WDT.

Секция программы, которая сбрасывает WDT, должна проверять другие секции программы на правильность работы. Это можно сделать, проверяя на принадлежность их приемлемым значениям. Простое использование программного таймера для сброса WDT каждые 10 мс должно обеспечивать защиту только для катастрофических ошибок.

В таблице 3.5.1 приведена система команд микроконтроллеров.

В таблице 3.5.2 приведена длина команды (в байтах), через дробь указан код операции.

Таблица 3.5.1 – Система команд микроконтроллеров

Мнемоника	Операнды	Операции (примечание 1)	Флаги (примечание II)						Примечания
			Z	N	C	V	VT	ST	
1	2	3	4	5	6	7	8	9	10
ADD/ADDB	2	D ← D + A	√	√	√	√	↑	-	
ADD/ADDB	3	D ← B + A	√	√	√	√	↑	-	
ADDC/ADDCB	2	D ← D + A + C	↓	√	√	√	↑	-	
SUB/SUBB	2	D ← D - A	√	√	√	√	↑	-	
SUB/SUBB	3	D ← B - A	√	√	√	√	↑	-	
SUBC/SUBCB	2	D ← D - A + C - 1	↓	√	√	√	↑	-	
CMP/CMPB	2	D - A	√	√	√	√	↑	-	
MUL/MULU	2	D,D + 2 ← D x A	-	-	-	-	-	-	2
MUL/MULU	3	D,D + 2 ← B x A	-	-	-	-	-	-	2
MULB/MULUB	2	D,D + 1 ← D x A	-	-	-	-	-	-	3
MULB/MULUB	3	D,D + 1 ← B x A	-	-	-	-	-	-	3
DIVU	2	D←(D,D+2)/A,D+2 ←остаток	-	-	-	√	↑	-	2
DIVUB	2	D←(D,D+1)/A,D+1 ←остаток	-	-	-	√	↑	-	3
DIV	2	D←(D,D+2)/A,D+2 ←остаток	-	-	-	√	↑	-	
DIVB	2	D←(D,D+1)/A,D+1 ←остаток	-	-	-	√	↑	-	
AND/ANDB	2	D ← D AND A	√	√	0	0	-	-	
AND/ANDB	3	D ← B AND A	√	√	0	0	-	-	
OR/ORB	2	D ← D OR A	√	√	0	0	-	-	
XOR/XORB	2	D ← D (ecxl. or) A	√	√	0	0	-	-	
LD/LDB	2	D ← A	-	-	-	-	-	-	
ST/STB	2	A ← D	-	-	-	-	-	-	
LDBSE	2	D ← A; D + 1 ← SIGN(A)	-	-	-	-	-	-	3,4
LDBZE	2	D ← A; D + 1 ← 0	-	-	-	-	-	-	3,4
PUSH	1	SP ← SP - 2; (SP) ← A	-	-	-	-	-	-	
POP	1	A ←(SP); SP + 2	-	-	-	-	-	-	
PUSHF	0	SP ← SP - 2; (SP) ← PSW; PSW ← 0000H; I ← 0	0	0	0	0	0	0	
POPF	0	PSW←(SP); SP←SP+2; I← √	√	√	√	√	√	√	
SJMP	1	PC ← PC + 11-бит поле	-	-	-	-	-	-	5
LJMP	1	PC ← PC + 16-бит поле	-	-	-	-	-	-	5
BR (косвенный)	1	PC ← (A)	-	-	-	-	-	-	
SCALL	1	SP ← SP - 2; (SP)←PC; PC←PC+11-бит поле	-	-	-	-	-	-	5
LCALL	1	SP ← SP - 2;(SP)←PC; PC ← PC + 16-бит поле	-	-	-	-	-	-	5
RET	0	PC ← (SP); SP ← SP + 2	-	-	-	-	-	-	
J (условные)	1	PC←PC + 8-бит поле (если выполняется ветвление)	-	-	-	-	-	-	5
JC	1	переход если C=1	-	-	-	-	-	-	5
JNC	1	переход если C=0	-	-	-	-	-	-	5
JE	1	переход если Z=1	-	-	-	-	-	-	5
JNE	1	переход если Z=0	-	-	-	-	-	-	5
JGE	1	переход если N=0	-	-	-	-	-	-	5
JLT	1	переход если N=1	-	-	-	-	-	-	5
JGT	1	переход если N=0 и Z=0	-	-	-	-	-	-	5
JLE	1	переход если N=1 или Z=1	-	-	-	-	-	-	5

Продолжение таблицы 3.5.1

1	2	3	4	5	6	7	8	9	10
JH	1	переход если C=1 и Z=0	-	-	-	-	-	-	5
JNH	1	переход если C=0 или Z=1	-	-	-	-	-	-	5
JV	1	переход если V=1	-	-	-	-	-	-	5
JNV	1	переход если V=0	-	-	-	-	-	-	5
JVT	1	переход если VT=1;очистка VT	-	-	-	-	0	-	5
JNVT	1	переход если VT=0;очистка VT	-	-	-	-	0	-	5
JST	1	переход если ST=1	-	-	-	-	-	-	5
JNST	1	переход если ST=0	-	-	-	-	-	-	5
JBS	3	переход если определ. бит.= 1	-	-	-	-	-	-	5,6
JBC	3	переход если определ. бит.= 0	-	-	-	-	-	-	5,6
DJNZ/ DJNZW	1	D ← D - 1 если D ≠ 0 то PC←PC+8 бит поле	-	-	-	-	-	-	5 10
DEC/DECB	1	D ← D - 1	√	√	√	√	↑	-	
NEG/NEGB	1	D ← 0 - D	√	√	√	√	↑	-	
INC/INCB	1	D ← D + 1	√	√	√	√	↑	-	
EXT	1	D ← D; D + 2 ← Sign(D)	√	√	0	0	-	-	2
EXTB	1	D ← D; D + 1 ← Sign(D)	√	√	0	0	-	-	3
NOT/NOTB	1	D ← Logical Not (D)	√	√	0	0	-	-	
CLR/CLRB	1	D ← 0	1	0	0	0	-	-	
SHL/SHLB/ SHLL	2	C ← ст.бит....мл.бит ← 0 — —	√	√	√	√	↑	-	7
SHR/SHRB/ SHRL	2	0 → ст.бит....мл.бит → C — —	√	√	√	0	-	√	7
SHRA/SHRAB/ SHRAL	2	ст.бит → ст.б....мл.б → C — —	√	√	√	0	-	√	7
SETC	0	C ← 1	-	-	1	-	-	-	
CLRC	0	C ← 0	-	-	0	-	-	-	
CLRVT	0	VT ← 0	-	-	-	-	0	-	
RST	0	PC ← 2080H	0	0	0	0	0	0	8
DI	0	запрет всех прерываний (I←0)	-	-	-	-	-	-	
EI	0	разрешение (I←1)	-	-	-	-	-	-	
NOP	0	PC ← PC + 1	-	-	-	-	-	-	
SKIP	0	PC ← PC + 2	-	-	-	-	-	-	
NORML	2	Сдвиг влево пока ст. бит не станет =1; D ← счет сдвигов	√	√	0	-	-	-	7
TRAP	0	SP ← SP-2; (SP) ← PC; PC ← (2010H)	-	-	-	-	-	-	9
PUSHA	1	SP ← SP-2; (SP) ← PSW; PSW ← 0000H; SP ← SP-2; (SP)←IMASK1/WSR; IMASK1←00H	0	0	0	0	0	0	
POPA	1	IMASK1/WSR←(SP); SP←SP+2 PSW←(SP); SP←SP + 2	√	√	√	√	√	√	
IDLPD	1	Режим IDLE если ключ = 1; Режим энергосбережения если ключ = 2; (Power down) Иначе сброс;	-	-	-	-	-	-	
CMPL	2	D-A	√	√	√	√	↑	-	
BMOV	2	[PTR_HI]+ ← [PTR_LOW]+; до тех пор, пока счетчик не станет= 0.	-	-	-	-	-	-	

Примечание I к таблице 3.5.1

1 Если мнемоника оканчивается на "B" - операции выполняются с байтом, иначе выполняется операция со словом. Операнды D, B и A должны соответствовать правилам выравнивания для требуемых типов операндов. Операнды D и B располагаются в регистровом файле, A может быть расположен в любом месте памяти.
2 D, D + 2 - последовательность слов в памяти; D - двойное слово.
3 D, D + 1 - последовательность бит в памяти; D - слово.
4 Замена байтов словом.
5 Поле дополнения до двух.
6 Определяет бит как 1 из 2048 в регистровом файле.
7 "L" (длинный) суффикс указывает на операцию двойного слова.
8 Инициирует сброс установкой низкого уровня RESET#. Программа будет инициализировать все необходимые регистры с начального адреса 2080H.
9 Ассемблер не допускает такую мнемонику.
10 DJNZW - инструкция не гарантируемая для исполнения.

Примечание II к таблице 3.5.1 (для каждой команды показано, как она изменяет флаги)

1 Символ "√" означает, что флаг устанавливается или сравнивается в соответствии с результатом операции.
2 Символ "-" означает, что флаг не модифицируется.
3 Символ "1" или "0" означает, что флаг устанавливается или сбрасывается всегда.
4 Символ "↑" означает, что команда может устанавливать флаг (в результате вычислений и в соответствии с назначением флага), но не может сбросить его. Тот же смысл только для сброса флага имеет символ "↓".
5 Символ "?" означает, что флаг переходит в неопределенное (случайное) состояние.

Таблица 3.5.2 - Длина команды (в байтах)/код операции

Мнемоника	Прямая	Непо-средст-венная	Косвенная		Индексная	
			нормальная(1)	A - инкре-мент(1)	корот-кая(1)	длин-ная(1)
ADD(3-OP)	4/44	5/45	4/46	4/46	5/47	6/47
SUB(3-OP)	4/48	5/49	4/4A	4/4A	5/4B	6/4B
ADD(2-OP)	3/64	4/65	3/66	3/66	4/67	5/67
SUB(2-OP)	3/68	4/69	3/6A	3/6A	4/6B	5/6B
ADDC	3/A4	4/A5	3/A6	3/A6	4/A7	5/A7
SUBC	3/A8	4/A9	3/AA	3/AA	4/AB	5/AB
CMP	3/88	4/89	3/AB	3/AB	4/8B	5/8B
ADDB(3-OP)	4/54	4/55	4/56	4/56	5/57	6/57
SUBB(3-OP)	4/58	4/59	4/5A	4/5A	5/5B	6/5B
ADDB(2-OP)	3/74	3/75	3/76	3/76	4/77	5/77
SUBB(2-OP)	3/78	3/79	3/7A	3/7A	4/7B	5/7B
ADDCB	3/B4	3/B5	3/B6	3/B6	4/B7	5/B7
SUBCB	3/B8	3/B9	3/BA	3/BA	4/BB	5/BB
CMPB	3/98	3/99	3/9A	3/9A	4/9B	5/9B
MUL(3-OP)	5/(2)	6/(2)	5/(2)	5/(2)	6/(2)	7/(2)
MULU(3-OP)	4/4C	5/4D	4/4E	4/4E	5/4F	6/4F
MUL(2-OP)	4/(2)	5/(2)	4/(2)	4/(2)	5/(2)	6/(2)
MULU(2-OP)	3/6C	4/6D	3/6E	3/6E	4/6F	5/6F
DIV	4/(2)	5/(2)	4/(2)	4/(2)	5/(2)	6/(2)
DIVU	3/8C	4/8D	3/8E	3/8E	4/8F	5/8F
MULB(3-OP)	5/(2)	5/(2)	5/(2)	5/(2)	6/(2)	7/(2)
MULUB(3-OP)	4/5C	4/5D	4/5E	4/5E	5/5F	6/5F
MULB(2-OP)	4/(2)	4/(2)	4/(2)	4/(2)	5/(2)	6/(2)
MULUB(2-OP)	3/7C	3/7D	3/7E	3/7E	4/7F	5/7F
DIVB	4/(2)	4/(2)	4/(2)	4/(2)	5/(2)	6/(2)
DIVUB	3/9C	3/9D	3/9E	3/9E	4/9F	5/9F
AND(3-OP)	4/40	5/41	4/42	4/42	5/43	6/43
AND(2-OP)	3/60	4/61	3/62	3/62	4/63	5/63
OR(2-OP)	3/80	4/81	3/82	3/82	4/83	5/83
XOR	3/84	4/85	3/86	3/86	4/87	5/87
ANDB(3-OP)	4/50	4/51	4/52	4/52	5/53	5/53
ANDB(2-OP)	3/70	3/71	3/72	3/72	4/73	4/73
ORB(2-OP)	3/90	3/91	3/92	3/92	4/93	5/93
XORB	3/94	3/95	3/96	3/96	4/97	5/97
PUSH	2/C8	3/C9	2/CA	2/CA	3/CB	4/CB
POP	2/CC	-	2/CE	2/CE	3/CF	4/CF
LD	3/A0	4/A1	3/A2	3/A2	4/A3	5/A3
LDB	3/B0	3/B1	3/B2	3/B2	4/B3	5/B3
ST	3/C0	-	3/C2	3/C2	4/C3	5/C3
STB	3/C4	-	3/C6	3/C6	4/C7	5/C7
LDBSE	3/BC	3/BD	3/BE	3/BE	4/BF	5/BF
LBSZE	3/AC	3/AD	3/AE	3/AE	4/AF	5/AF

Продолжение таблицы 3.5.2 - Длина команды (в байтах)/код операции

Мнемоника	Длина/код операции
PUSHF	1/F2
POPF	1/F3
PUSHA	1/F4
POPA	1/F5
TRAP	1/F7
LCALL	3/EF
SCALL	2/28-2F(3)
RET	1/F0
LJMP	3/E7
SJMP	2/20-27(3)
BR[]	2/E3
JNST	1/D0
JST	1/D8
JNH	1/D1
JH	1/D9
JGT	1/D2
JLE	1/DA
JNC	1/B3
JC	1/D8
JNVT	1/D4
JVT	1/DC
JNV	1/D5
JV	1/DD
JGE	1/D6
JLT	1/DE
JNE	1/D7

Мнемоника	Длина/код операции
JE	1/DF
JBC	3/30-37
JBS	3/38-3F
DJNZ	3/EO
DJNZW	3/E1(4)
NORML	3/0F
SHRL	3/0C
SHLL	3/0D
SHRAL	3/0E
SHR	3/08
SHRB	3/18
SHL	3/09
SHLB	3/19
SHRA	3/0A
SHRAB	3/1A
CLRC	1/F8
SETC	1/F9
DI	1/FA
EI	1/FB
CLRVT	1/FC
NOP	1/FD
RST	1/FF
SKIP	2/00
IDLPD	1/F6
BMOV	3/C1

Примечания к таблице 3.5.2.

1 Косвенная и косвенная автоинкрементная адресация имеют те же коды операции, как и короткая и длинная индексная адресация. Если второй байт четный, выполняется косвенная или короткая индексная адресация. Если нечетный, выполняется косвенная автоинкрементная или длинная индексная адресация;

2 Код операции для знакового умножения и деления расширен префиксом "FE".

3 3 младших значащих бита кода операции объединяются с 8 битами в 11-битное поле с дополнением до двух.

4 Инструкция DJNZW не гарантируется для исполнения.

Таблица 3.5.3 - Время исполнения команды в машинных циклах(1)

Мнемоника	Прямая	Непосред-ственная	Косвенная		Индексная	
			нормальная*	А-инкремент*	корот-кая*	длин-ная*
ADD(3-OP)	5	6	7/10	8/11	7/10	8/11
SUB(3-OP)	5	6	7/10	8/11	7/10	8/11
ADD(2-OP)	4	5	6/8	7/9	6/8	7/9
SUB(2-OP)	4	5	6/8	7/9	6/8	7/9
ADDC	4	5	6/8	7/9	6/8	7/9
SUBC	4	5	6/8	7/9	6/8	7/9
CMP	4	5	6/8	7/9	6/8	7/9
ADDB(3-OP)	5	5	7/10	8/11	7/10	8/11
SUBB(3-OP)	5	5	7/10	8/11	7/10	8/11
ADDB(2-OP)	4	4	6/8	7/9	6/8	7/9
SUBB(2-OP)	4	4	6/8	7/9	6/8	7/9
ADDCB	4	4	6/8	7/9	6/8	7/9
SUBCB	4	4	6/8	7/9	6/8	7/9
CMPB	4	4	6/8	7/9	6/8	7/9
MUL(3-OP)	16	17	18/21	19/22	19/22	20/23
MULU(3-OP)	14	15	16/19	17/19	17/20	18/21
MUL(2-OP)	16	17	18/21	19/22	19/22	20/23
MULU(2-OP)	14	15	16/19	17/19	17/20	18/21
DIV	26	27	28/31	29/32	29/32	30/33
DIVU	24	25	26/29	27/30	27/30	28/31
MULB(3-OP)	12	12	14/17	13/15	15/18	16/19
MULUB(3-OP)	10	10	12/15	12/16	12/16	14/17
MULB(2-OP)	12	12	14/17	15/18	15/18	16/19
MULUB(2-OP)	10	10	12/15	13/15	12/16	14/17
DIVB	18	18	20/23	21/24	21/24	22/25
DIVUB	16	16	18/21	19/22	19/22	20/23
AND(3-OP)	5	6	7/10	8/11	7/10	8/11
AND(2-OP)	4	5	6/8	7/9	6/8	7/9
OR(2-OP)	4	5	6/8	7/9	6/8	7/9
XOR	4	5	6/8	7/9	6/8	7/9
ANDB(3-OP)	5	5	7/10	8/11	7/10	8/11
ANDB(2-OP)	4	4	6/8	7/9	6/8	7/9
ORB(2-OP)	4	4	6/8	7/9	6/8	7/9
XORB	4	4	6/8	7/9	6/8	7/9
PUSH (внутр.стек)	6	7	9/12	10/13	10/13	11/14
POP (внутр. стек)	8	-	10/12	11/13	11/13	12/14
PUSH (внеш. стек)	8	9	11/14	12/15	12/15	13/16
POP (внеш. стек)	11	-	13/15	14/16	14/16	15/17
LD,LDB	4,4	5,4	5/8	6/8	6/9	7/10
ST,STB	4,4	-	5/8	6/8	6/9	7/10
LDBSE	4	4	5/8	6/8	6/9	7/10
LBSZE	4	4	5/8	6/8	6/9	7/10
BMOV	внутренняя / внутренняя: 6+8 на слово внешняя / внутренняя: 6+11 на слово внешняя / внешняя: 6+14 на слово					

*Время для операндов: РСН и внутреннее ОЗУ (0-1FFH) / контроллер памяти (200H-0FFFFH).

Примечание - Время выполнения команд через контроллер памяти может быть на один или два цикла больше в зависимости от числа байт в предварительной очереди. Внутренний стек 200H-1FFH и внешний стек 200H-0FFFFH.

Таблица 3.5.4 - Время выполнения команд в машинных циклах

Мнемоника	Время	Мнемоника	Время
PUSHF(внутр. стек) POPF (внутр. стек) PUSHA(внутр. стек) POPA (внутр. стек)	6 7 12 12	PUSHF(внешн. стек) POPF (внешн. стек) PUSHA(внешн. стек) POPA (внешн. стек)	8 10 18 18
TRAP (внутр. стек) LCALL(внутр. стек) SCALL(внутр. стек) RET (внутр. стек)	16 11 11 11	TRAP (внешн. стек) LCALL(внешн. стек) SCALL(внешн. стек) RET (внешн. стек)	18 13 13 14
CMPL CLR/CLRB NOT/NOTB NEG/NEGB	7 3 3 3	DEC/DECB EXT/EXTB NC/INCB	3 4 3
LJMP	7		
SJMP BR (косвенный) JNST,JST JNH,JH JGT,JLE JNC,JC JNVT,JVT JNV,JV JGE,JLT JNE,JE JBC,JBS	7 7 7 4/8 переход не происходит/переход происходит 4/8 переход не происходит/переход происходит 4/8 переход не происходит/переход происходит 4/8 переход не происходит/переход происходит 4/8 переход не происходит/переход происходит 4/8 переход не происходит/переход происходит 4/8 переход не происходит/переход происходит 5/9 переход не происходит/переход происходит		
DJNZ DJNZW(1)	5/9 переход не происходит/переход происходит 5/9 переход не происходит/переход происходит		
NORML SHRL SHLL SHRAL SHR/SHRB SHL/SHLB SHRA/SHRAB	8+1 на сдвиг (9 для 0 сдвиг) 7+1 на сдвиг (8 для 0 сдвиг) 7+1 на сдвиг (8 для 0 сдвиг) 7+1 на сдвиг (8 для 0 сдвиг) 6+1 на сдвиг (7 для 0 сдвиг) 6+1 на сдвиг (7 для 0 сдвиг) 6+1 на сдвиг (7 для 0 сдвиг)		
CLRC SETC DI EI CLRVT NOP RST SKIP IDLPD	2 2 2 2 2 2 15 (включая запись байта конфигурации) 3 8/25(правильный ключ/неправильный ключ)		

Примечание - Команда DJNZW не гарантируется для исполнения.

4 Периферийные устройства

Этими устройствами являются пять главных периферийных блоков ИС: блок широтно-импульсной модуляции (PWM), Таймер 1 и Таймер 2, устройство высокоскоростных входов-выходов, последовательный порт и аналого-цифровой преобразователь. За исключением высокоскоростных входов и выходов (HSIO), каждое периферийное устройство используется отдельно.

Четыре отдельные секции, входящие в состав HSIO, совместно образуют гибкую систему ввода-вывода, базирующуюся на таймерах/счетчиках. В HSIO входят 16-битный таймер (Таймер 1), 16-битный реверсивный счетчик (Таймер 2), блок программируемых входов (HSI) и блок программируемых высокоскоростных выходов (HSO). С очень малыми затратами ресурсов CPU устройство HSIO может измерять ширину импульса, формировать сигналы и периодически производить прерывания. В зависимости от режима HSIO может производить работу с 18 таймерами/счетчиками и регистрами захвата/сравнения.

4.1 Выход блока широтно-импульсной модуляции (PWM)

Цифро-аналоговое преобразование (D/A) может быть произведено на выходе блока широтно-импульсной модуляции (PWM). Выходной сигнал – импульс с изменяемой скважностью, с повторением каждые 256 или 512 циклов системного тактового сигнала, если подключен делитель на 2. Изменение скважности производится записью в PWM регистр.

4.2 Таймеры

В ИС для пользователя доступны два 16-разрядных таймера. Первый обозначен, как Таймер 1, второй – Таймер 2. Таймер 1 используется для синхронизации событий в реальном масштабе времени, тогда как Таймер 2 тактируется снаружи и синхронизирует действия по отношению к внешним событиям. Таймеры являются времязадающей базой блоков высокоскоростных входов (HSI) и высокоскоростных выходов (HSO) и могут считаться встроенной частью этих блоков.

Таймер 1 – это автономный таймер, который инкрементируется каждые восемь машинных циклов. Таймер 1 может вызывать прерывания при переполнении.

Таймер 2 считает переключения по передним и задним фронтам по любому входу – T2CLK или HSI.1. Таймер 2 может быть читаемым и записываемым, может быть сброшен аппаратно, программно и блоком HSO. Он может использоваться как счетчик с инкременцией/декременцией (при управлении по P2.6) и его значение может быть захвачено T2CAP-регистром. Прерывания могут генерироваться после захвата значения Таймера 2, и если значение Таймера 2 выходит за границы 0FFFFH/0000H или 7FFFH/8000H (при обратном счете).

4.3 Высокоскоростные входы (HSI)

Устройство высокоскоростных входов (HSI) может захватывать значение Таймера 1, когда событие имеет место на одном из четырех входов (HSI.0 – HSI.3). Четыре типа событий могут фиксироваться: положительный фронт, отрицательный фронт, положительный или отрицательный фронт или каждый восьмой положительный фронт. Блок-схема HSI приведена на рисунке 4.3.1.

Когда значение из регистра HSI_TIME считано, следующий FIFO регистр загружается в регистр хранения. Три типа HSI-прерываний могут генерироваться, когда значение передается из FIFO в регистр хранения, когда FIFO (независимо от буферного регистра) запоминает четыре или более событий.

Это позволяет оптимизировать HSI для ожидаемой частоты прерываний. Независимо от HSI операций, состояние HSI выводов отображается 4 битами регистра HSI_STATUS. Также, независимо от HSI операций, HSI.0 может использоваться для внешнего прерывания, если вывод (HSI.0) не установлен как вход HSI.

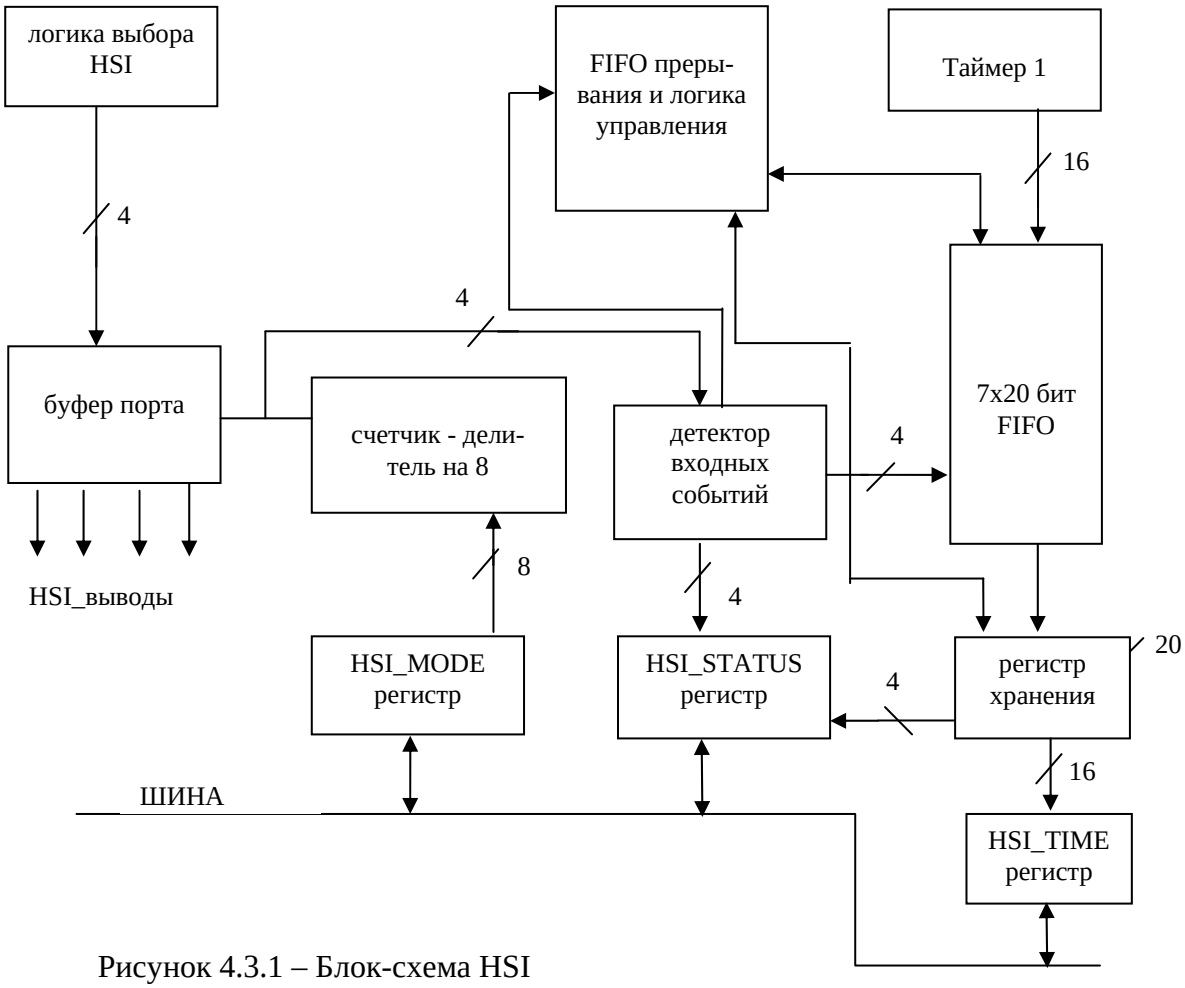


Рисунок 4.3.1 – Блок-схема HSI

4.4 Высокоскоростные выходы (HSO)

Устройство высокоскоростных выходов может генерировать события в определенное время или на базе Таймера 1 или Таймера 2 с минимальным управлением от CPU.

Блок-схема устройства HSO показана на рисунке 4.4.1. До восьми задержанных событий можно запомнить в CAM устройстве HSO одновременно. Команды, поступающие в HSO, записываются в HSO_COMMAND для формирования события и затем в HSO_TIME для согласования со значением таймера.

14 различных типов событий могут обрабатываться HSO: 8 внешних и 6 внутренних. Два вектора прерываний базируются на HSO: один – для внешних событий, другой – для внутренних. Внешние события представляют переключения по одному или нескольким выводам HSO.0 – HSO.5. Внутренние события включают в себя установку до четырех программных таймеров, сброс Таймера 2 и старт аналого-цифрового преобразования. Флаги программных таймеров могут устанавливаться HSO и дополнительно вызывать прерывания.

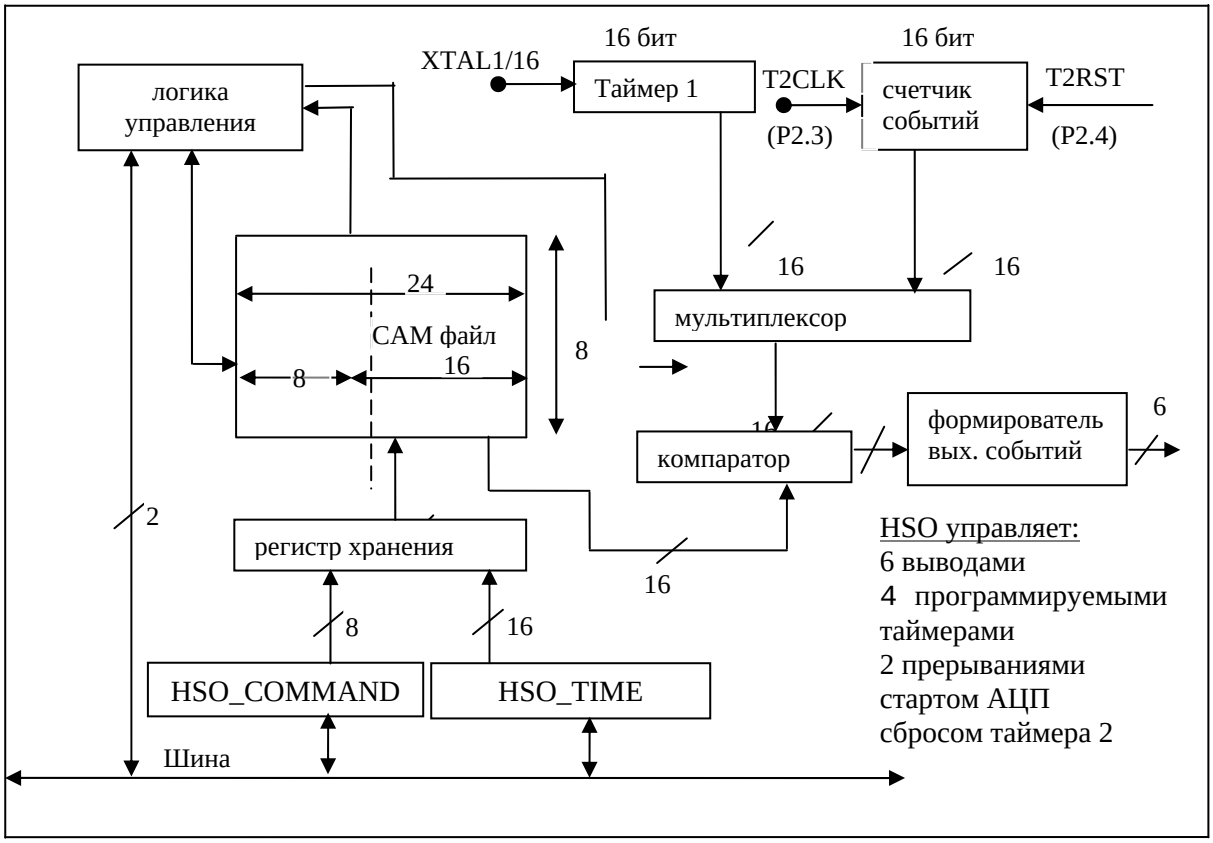


Рисунок 4.4.1 – Блок-схема HSO

4.5 Последовательный порт

Последовательный порт ИС функционально совместим с последовательным портом микроконтроллерного семейства 1830. Имеются один синхронный и три асинхронных режима. Асинхронный режим полностью дуплексный, то есть передача и прием в одно время. Двойная буферизация предусмотрена для приемника таким образом, что второй байт может быть принят раньше, чем первый байт принятой информации считан. Передатчик имеет также двойную буферизацию, позволяющую записывать байты (в буфер), пока идет передача.

Регистр состояния последовательного порта (SP_STAT) содержит биты переполнения, четности, ошибок, прерывания приема и передачи.

Режимы последовательного порта:

1 РЕЖИМ 0 (Mode 0)

Режим 0 – это синхронный режим, обычно используемый для увеличения скорости ввода-вывода с помощью регистра сдвига. В этом режиме TXD выдает набор из восьми импульсов, тогда как RXD служит для передачи и приема данных. Данные передаются по 8 бит, начиная с младшего бита. Следует отметить, что это единственный режим, в котором RXD используется как выход.

2 РЕЖИМ 1 (Mode 1)

Это стандартный асинхронный режим связи. Посылка содержит 10 бит: стартовый бит (0), 8 бит данных (младший – первый), стоповый (1). Если четность разрешена (SPCON.2), то бит четности посылается за восьмым битом данных, и четность проверяется на приеме.

3 РЕЖИМ 2 (Mode 2)

Режим 2 – это асинхронный 9-битный режим. Обычно он используется совместно с режимом 3 для мультипроцессорных коммуникаций.

Данные содержат: стартовый бит (0), 9 бит данных (младший – первый), стоповый бит (1). При передаче девятый бит может быть установлен в 1, через установку бита TB8 в управляющем регистре перед записью в SBUF(TX). Бит TB8 очищается при каждой передаче, поэтому он должен быть установлен перед записью в SBUF(TX) каждый раз, когда это требуется. При приеме прерывание последовательного порта и прерывание приема (RI) не будут устанавливаться, если девятый принятый бит не будет установлен.

4 РЕЖИМ 3 (Mode 3)

Это асинхронный 9-битный режим. Посылка данных идентична посылке для режима 2. Отличие заключается в том, что при передаче бит четности может быть разрешен (PEN = 1), и тогда девятый бит будет принимать четное значение. Бит TB8 может использоваться, если бит четности запрещен (PEN = 0). В режиме 3 прием данных будет всегда вызывать прерывание вне зависимости от состояния 9-ого бита. Если PEN = 0, то девятый бит запоминается и может быть прочитан из RB8. Если PEN = 1, то RB8 станет флагом ошибки четности на приеме (RPE).

4.6 Блок АЦП

Аналоговый интерфейс ИС содержит блок захвата и хранения аналогового сигнала, 8-канальный мультиплексор и блок 10-разрядного аналого-цифрового преобразователя.

Аналоговый сигнал выбирается на одном из 8 аналоговых входов (ACH0 – ACH7), которые входят в порт 0. АЦ-преобразования производят на одном из входов, с использованием последовательной аппроксимации; результат равен отношению входного напряжения к опорному напряжению. Если отношение – единица, то результат – все 1. Преобразование может быть запущено записью в регистр A/D_Command или командой HSO.

4.7 Порты ввода-вывода

ИС имеют пять 8-битных портов ввода-вывода. Часть из них чисто входные, часть работают только на выход, некоторые двунаправленные, а некоторые имеют альтернативные функции. В дополнение к ним выводы устройства быстрого ввода-вывода могут использоваться как линии обычного ввода-вывода, если не нужны специальные возможности устройства быстрого ввода-вывода.

Входные порты соединены с внутренней шиной микроконтроллеров через входные буфера. Выходные порты имеют регистры-защелки для выходных данных и выходные буфера. Двунаправленные порты содержат выходной буфер, регистр-защелку и входной буфер.

Порт 0 – входной; может использоваться как аналоговый вход для АЦП. Порт 1 – квази-двунаправленный. Порт 2 содержит 3 типа выводов: квази-двунаправленные, входные, выходные.

Порты 3 и 4 – двунаправленные; к ним подключена шина внешнего адреса/данных.

4.8 Сторожевой таймер

Сторожевой таймер (WDT) позволяет восстанавливать нормальную работу при сбоях программ. Если WDT разрешен, он будет вызывать аппаратный сброс, если программа не очищает его каждые 64K машинных цикла.

Когда таймер переполняется, он переводит вход RESET# в состояние низкого уровня, сбрасывая ИС и другие устройства, подключенные к выводу RESET#.

Работа WDT разрешается первой же его очисткой. Если она разрешена, то запрещена может быть только аппаратным сбросом.