

Глава 14. ОСОБЕННОСТИ ПРИМЕНЕНИЯ МИКРОКОНТРОЛЛЕРОВ ФИРМЫ CYGNAL

14.1. Эволюционные комплекты фирмы Cygnal

Фирма Cygnal делает все возможное для того, чтобы облегчить труд разработчиков, занимающихся изучением особенностей и возможностей новых микроконтроллеров C8051Fxxx. Для каждого семейства микроконтроллеров фирмой Cygnal производится эволюционный комплект (Development Kit), позволяющий ознакомиться с этим семейством, быстро и легко опробовать фрагменты программ, а иногда и аппаратной реализации отдельных узлов на новой элементной базе. В состав комплекта поставки любого кита входят:

- Собственно эволюционная плата для выбранного типа микроконтроллера;
- Адаптер-программатор, преобразующий входной интерфейс персонального компьютера RS232C в интерфейс программирования выбранного типа микроконтроллера (JTAG или I2C). В последнее время поставляется комбинированный адаптер EC2, поддерживающий оба интерфейса программирования. Адаптер выполнен в пластиковом корпусе с размерами 50x58x26 мм³. Внутри корпуса находится контроллер программатора, выполненный на микросхеме C8051F012 (см. раздел 14.1.5). В качестве интерфейсной схемы RS232C использована микросхема Sipex3223. Кроме того, на плате имеется стабилизатор напряжения на 3.3В, кварцевый резонатор и два светодиода. Входной разъем для RS232C - DB9F (розетка). Выходной разъем - двухрядный штырьковый разъем типа PLD10 (2x5). В нормальном рабочем режиме адаптер питается от эволюционной платы через 10-проводный плоский кабель. Напряжение питания 3 – 3.6 В. Кроме того, адаптер может питаться от внешнего источника питания с выходным напряжением 5 – 9 В, для чего предусмотрено стандартное гнездо.
- Настенный малогабаритный источник питания с размерами 55x42x35. Выходное напряжение – 9В, ток нагрузки до 100 мА.
- Стандартный компьютерный кабель с двумя разъемами DB9F (розетка) – DB9 (вилка) для подсоединения адаптера к персональному компьютеру. Для подсоединения к персональному компьютеру настольного типа, имеющему 25-пиновый выходной разъем последовательного порта DB25, необходим переходник DB25-DB9.
- 10-проводный плоский кабель с двумя разъемами DPS10 (2x5) для соединения адаптера и эволюционной платы.
- Техническое описание и руководство по эксплуатации.
- CD-ROM с программным обеспечением.

В состав программного обеспечения входит:

- Фирменная среда программирования Cygnal Integrated Development Tools.
- Комплект программного обеспечения фирмы Keil (макросемблер, линкер, библиотекарь, и эволюционная версия компилятора языка 'C' с ограничением по размеру выходного кода – 2К).
- Инсталляционная утилита Setup.exe.
- Исходные тексты примеров и файлов определения регистров.
- Документация.

Вся описанная выше комплектность одинакова для всех типов эволюционных комплектов. Т.о., единственное отличие этих комплектов заключается в различном исполнении собственно эволюционных плат. Для примера, в следующих разделах приведем описания четырех наиболее популярных эволюционных комплектов: 005, 020, 226 и 300.

14.1.1. Эволюционный комплект C8051F005

Принципиальная схема собственно эволюционной платы микроконтроллера C8051F005 показана на рис.14.1[22].

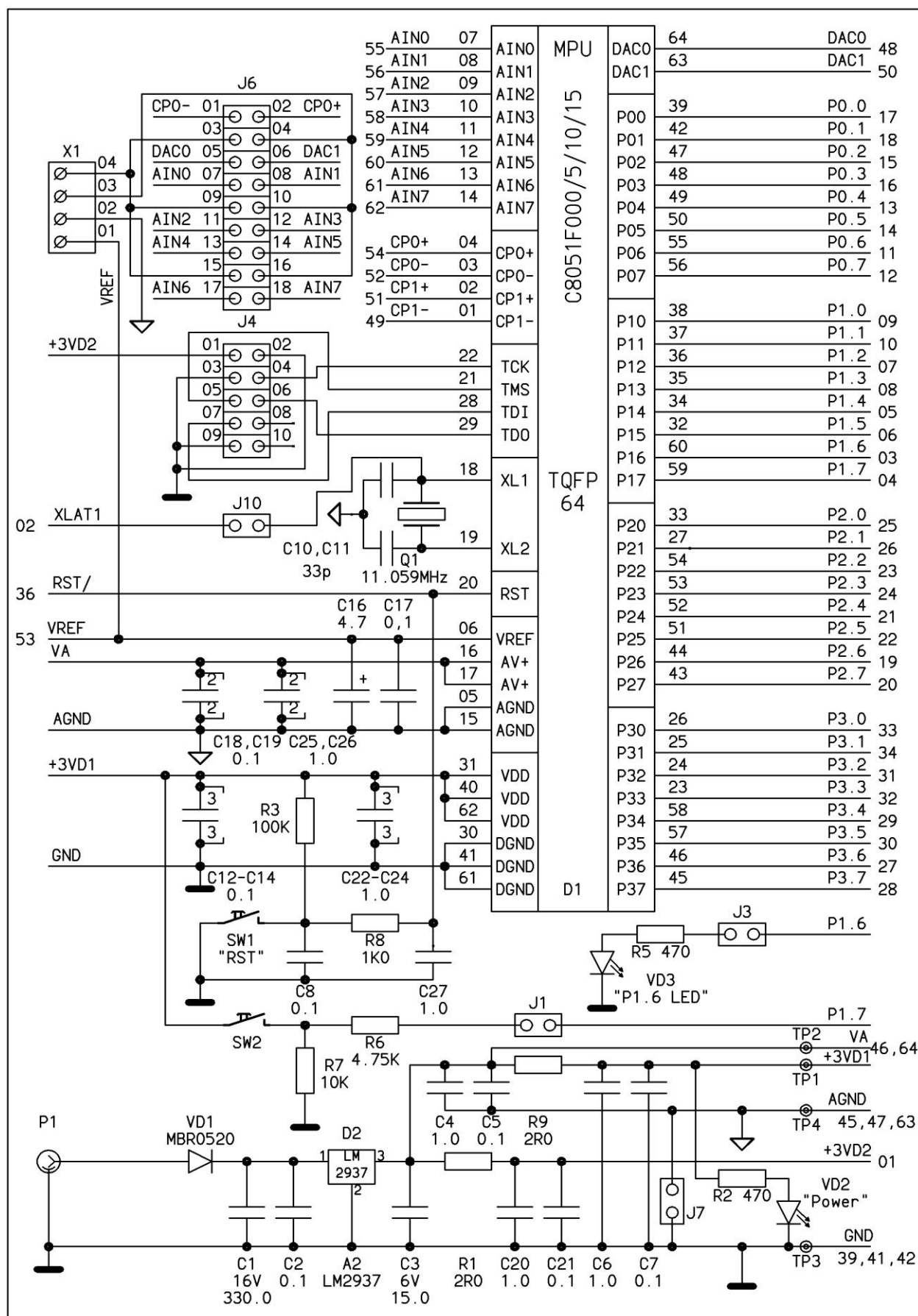


Рис.14.1. Принципиальная схема эволюционной платы C8051F005

Эволюционный контроллер содержит следующие функциональные узлы: собственно микроконтроллера D1 (C8051F005), стабилизатор питания A2 (LM2937-3.3) на напряжение 3.3V, светодиод состояния линии P1.6.- VD1, светодиод наличия питания - VD2, кнопка сброса - SW1, кнопка прерывания - SW2, Установочное место под внешний кварцевый генератор Q1, штырьковый двухрядный разъем J4 программирования JTAG PLD10 (2x5), штырьковый двухрядный разъем J6 аналоговых входов/выходов PLD18 (2x9), выходной штырьковый разъем, на который выведены все выводы корпуса микроконтроллера, за исключением четырех сигналов JTAG интерфейса PLD64 (2x32), контактные площадки, соединенные параллельно с контактами выходного разъема, а также два небольших макетных поля для аналоговой и цифровой частей. Размер эволюционной платы равен 85x110 мм². Макетные поля довольно маленькие. Меньшее макетное поле предназначено для аналоговых узлов и имеет 8x8 металлизированных отверстий с шагом 2,54 мм. С двух сторон этого поля располагаются шины аналоговой земли AGND и питания VA+. Второе поле большего размера предназначено для установки цифровых элементов. Оно имеет 8x20 металлизированных отверстий с шагом 2,54 мм. С двух узких сторон этого поля располагаются шины цифровой земли GND и питания VDD (+3VDD2). На всех шинах также с шагом 2,54 мм имеются металлизированные отверстия.

Перечень штырьковых разъемов платы приведен в таблице 14.1.

Таблица 14.1

Перечень разъемов эволюционной платы C8051F0xx

№№	Назначение
J1	Разъем для установки перемычки, которая может подключить кнопку SW2 к выводу P1.7 микроконтроллера.
J2	Выходной двухрядный 64 контактный разъем PLD64 (2x32), на который выведены все выводы микроконтроллера. Номера выводов указаны цифрами возле выходных связей. Рядом с этим разъемом расположены два ряда металлизированных отверстий, каждое из которых соединено с соответствующим контактом разъема J2 (см. табл.14.3).
J3	Разъем для установки перемычки, которая может подключить светодиод VD3 к выводу P1.6 микроконтроллера.
J4	Разъем для подключения адаптера-программатора (см. табл.14.2).
J5	Отсутствует.
J6	Разъем для коммутации аналоговых сигналов на клеммник винтовой четырехконтактный X1 типа 310-04-2-3.
J7	Разъем для установки перемычки, которая соединяет аналоговую и цифровую земли.
J8	Отсутствует.
J9	Отсутствует.
J10	Разъем для установки перемычки, которая может подключать вывод 02 разъема J2 - внешнего тактового генератора к выводу 18 микроконтроллера.
X1	Клеммник винтовой четырехконтактный типа 310-04-2-3 для подключения внешних аналоговых сигналов
P1	Коаксиальный разъем для подачи внешнего питания +5 - +9В.

В таблице 14.2 приведены контакты и описание связей разъема JTAG, используемого для внутрисистемного программирования и отладки. Вообще говоря, любое изделие, проектируемое на микроконтроллерах фирмы Cygnal, должно быть оснащено таким же разъемом для подключения адаптера-программатора EC2.

Таблица 14.2.

Перечень контактов разъема JTAG

Контакт	Описание
1	Вход питания от эволюционной платы .3.0-3.6V
2, 3, 9	Цифровая земля
4	TCK
5	TMS
6	TDO
7	TDI
8, 10	Не используются

Кроме этого, эволюционная плата содержит кнопку внешнего сброса SW1 и светодиод наличия питания VD2. Поскольку микроконтроллер имеет встроенный тактовый генератор, который после сброса начинает работать на частоте 2 МГц, внешний кварцевый резонатор Q1 не установлен, но установочное место для него предусмотрено. Подразумевается, что пользователь сам установит выбранный им самим кварцевый резонатор.

Эволюционная плата выполнена по технологии SMD (поверхностного монтажа) с защитным покрытием ("зеленкой") и оснащена всеми необходимыми надписями, облегчающими работу пользователя..

Как видно из описания, макетные поля эволюционной платы достаточно малы, поэтому дополнительные элементы можно частично устанавливать на самой эволюционной плате, а частично - на дополнительной макетной плате. На эволюционной плате имеет смысл устанавливать только общесистемные элементы (например, интерфейс RS232, интерфейс LCD (жидкокристаллического индикатора), таймер реального времени RTC и т.п.).

Для связи с дополнительной платой предназначен выходной штырьковый разъем J2 (PLD64), перечень контактов которого приведен в таблице 14.3.

Таблица 14.3

Выходной разъем J2 эволюционной платы C8051F0xx

Контакт	Описание	Контакт	Описание
1	+VD цифровое питание	2	XLAT1
3	P1.6	4	P1.7
5	P1.4	6	P1.5
7	P1.2	8	P1.3
9	P1.0	10	P1.1
11	P0.6	12	P0.7
13	P0.4	14	P0.5
15	P0.2	16	P0.3
17	P0.0	18	P0.1
19	P2.6	20	P2.7
21	P2.4	22	P2.5
23	P2.2	24	P2.3
25	P2.0	26	P2.1
27	P3.6	28	P3.7
29	P3.4	30	P3.5
31	P3.2	32	P3.3
33	P3.0	34	P3.1
35		36	RST/
37		38	
39	GND	40	
41	GND	42	GND
43		44	
45	AGND	46	+VA аналоговое питание
47	AGND	48	DAC0
49	CP1-	50	DAC1
51	CP1+	52	CP0-
53	VREF	54	CP0+
55	AIN0	56	AIN1
57	AIN2	58	AIN3
59	AIN4	60	AIN5
61	AIN6	62	AIN7
63	AGND	64	+VA аналоговое питание

Следует отметить, что связь двух плат, эволюционной и дополнительной макетной, с помощью разъема J2 осуществить достаточно сложно ввиду того, что стандартный плоский кабель с двумя разъемами DPS64 (2x32) найти достаточно сложно. Однако, после несложной доработки этого разъема, пользоваться платой становится намного удобнее. Доработка сводится к аккуратному выпайиванию из разъема J2 контактов 41-46 и соединению контактов 39 и 40. При этом, эволюционная плата легко соединяется с дополнительной макетной платой двумя гибкими кабелями. Первый 40-проводный плоский стандартный кабель с двумя разъемами DPS40 (2x20) широко используется в персональных компьютерах для подключения жесткого диска. В нашем случае он будет связывать все цифровые сигналы двух плат. Второй 20-проводный плоский стандартный кабель с двумя разъемами DPS20 (2x10) также можно найти в компьютерных магазинах или фирмах. Он будет связывать аналоговые сигналы двух плат. Очевидно, что для подключения аналоговых сигналов можно использовать разъем J6 и 18-проводный плоский кабель с разъемами DPS18 (2x9), однако на этот разъем не выведено аналоговое питание +VA и опорное напряжение VREF.

14.1.2. Эволюционный комплект C8051F020

Принципиальная схема эволюционной платы для улучшения читаемости разбита на три фрагмента[23]. В принципиальных схемах приведены оригинальные обозначения элементов. На рис.14.2 показан узел микроконтроллер с обвязкой. На рис.14.3 показан узел стабилизатора напряжения. И на рис.14.4 показан узел интерфейса RS232C.

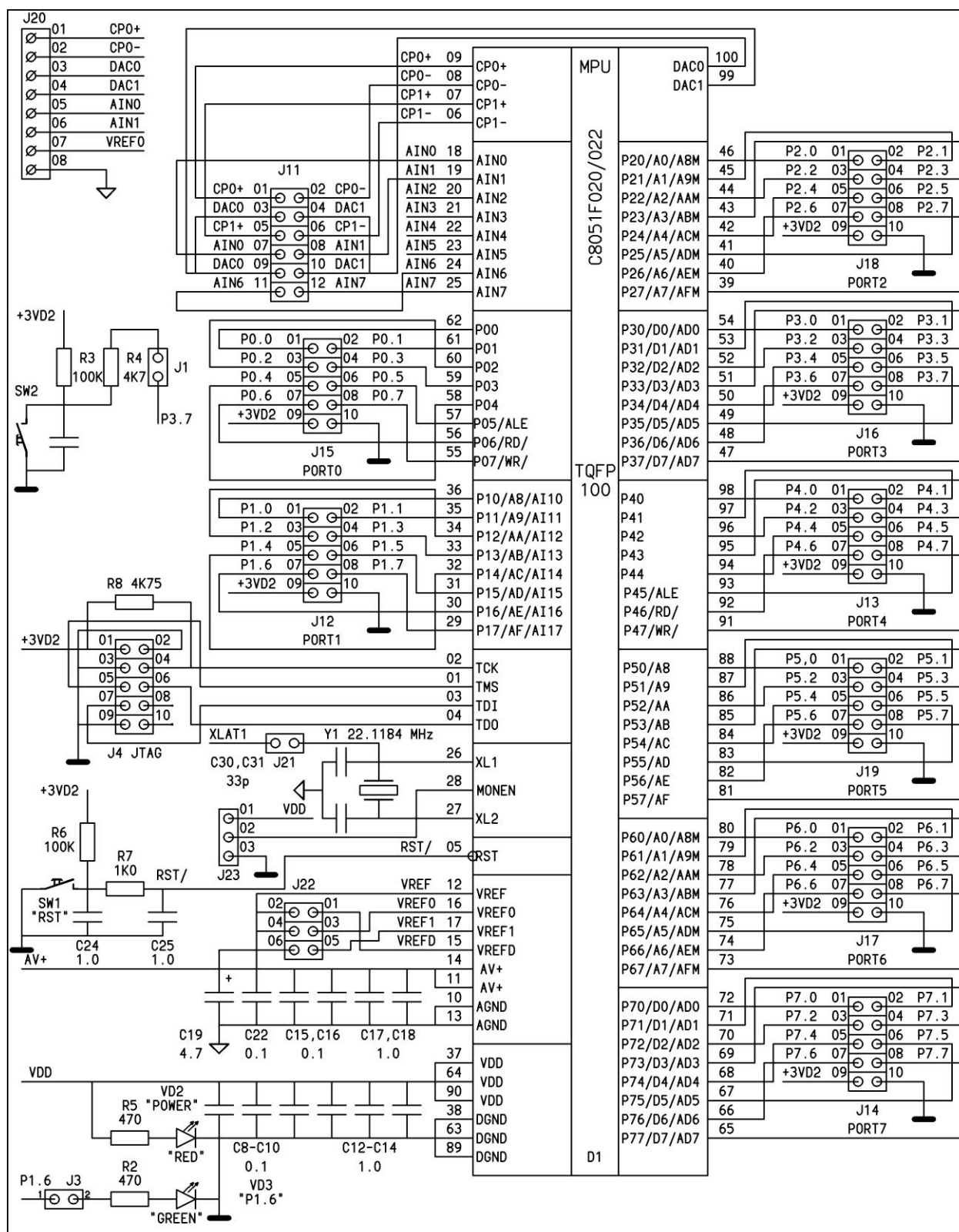


Рис.14.2. Узел микроконтроллера эволюционной платы C8051F020

Узел микроконтроллера содержит следующие функциональные узлы: собственно микроконтроллера D1 (C8051F020), светодиод наличия питания "POWER" - VD2 (красный), светодиод состояния линии «P1.6» - VD3 (зеленый), кнопка сброса "RST" - SW1, кнопка прерывания - SW2. Линии ввода/вывода всех портов P0-P7 выведены на штырьковые разъемы PLD10 (2x5) J15, J12, J18, J16, J13, J19, J17 и J14 соответственно. Установочное место под внешний кварцевый генератор Y1

на плате пустое, т.е. пользователь сам устанавливает необходимый тип резонатора. Штырьковый двухрядный разъем J4 - PLD10 (2x5) предназначен для программирования и отладки через интерфейс JTAG. Штырьковый двухрядный разъем J11 - PLD10 (2x5) предназначен для подключения аналоговых входов/выходов. Имеются также штырьковые разъемы J22 и J23 для коммутации опорного напряжения и установки режима встроенного монитора питания соответственно.

Размер эволюционной платы равен 85x110 мм². Эволюционная плата совсем не имеет макетных полей. Вместо этого, все сигнальные линии выведены на один системный трехрядный дюймовый разъем J24 типа C96 (отечественный аналог - СНП59-96В-23-1). Перечень контактов разъема приведен в таблице 14.4 (на рисунке разъем не показан).

Таблица 14.4

Таблица контактов системного разъема эволюционной платы C8051F020

Контакт	Наименование	Контакт	Наименование	Контакт	Наименование
A1	+3VD2 (+3.3VDC)	B1	DGND	C1	XLAT1
A2	MONEN	B2	P1.7	C2	P1.6
A3	P1.5	B3	P1.4	C3	P1.3
A4	P1.2	B4	P1.1	C4	P1.0
A5	P2.7	B5	P2.6	C5	P2.5
A6	P2.4	B6	P2.3	C6	P2.2
A7	P2.1	B7	P2.0	C7	P3.7
A8	P3.6	B8	P3.5	C8	P3.4
A9	P3.3	B9	P3.2	C9	P3.1
A10	P3.0	B10	P0.7	C10	P0.6
A11	P0.5	B11	P0.4	C11	P0.3
A12	P0.2	B12	P0.1	C12	P0.0
A13	P7.7	B13	P7.6	C13	P7.6
A14	P7.4	B14	P7.3	C14	P7.2
A15	P7.1	B15	P7.0	C15	P6.7
A16	P6.6	B16	P6.5	C16	P6.4
A17	P6.3	B17	P6.2	C17	P6.1
A18	P6.0	B18	P5.7	C18	P5.6
A19	P5.5	B19	P5.4	C19	P5.3
A20	P5.2	B20	P5.1	C20	P5.0
A21	P4.7	B21	P4.6	C21	P4.5
A22	P4.4	B22	P4.3	C22	P4.2
A23	P4.1	B23	P4.0	C23	TMS
A24	TCK	B24	TDI	C24	TDO
A25	RST/	B25	DGND	C25	VUNREG
A26	AGND	B26	DAC1	C26	DAC0
A27	CP1-	B27	CP1+	C27	CP0-
A28	CP0+	B28	VREF	C28	VREFD
A29	VREF0	B29	VREF1	C29	AIN0.7
A30	AIN0.6	B30	AIN0.5	C30	AIN0.4
A31	AIN0.3	B31	AIN0.2	C31	AIN0.1
A32	AIN0.0	B32	AGND	C32	AV+ (+3.3VDC Analog)

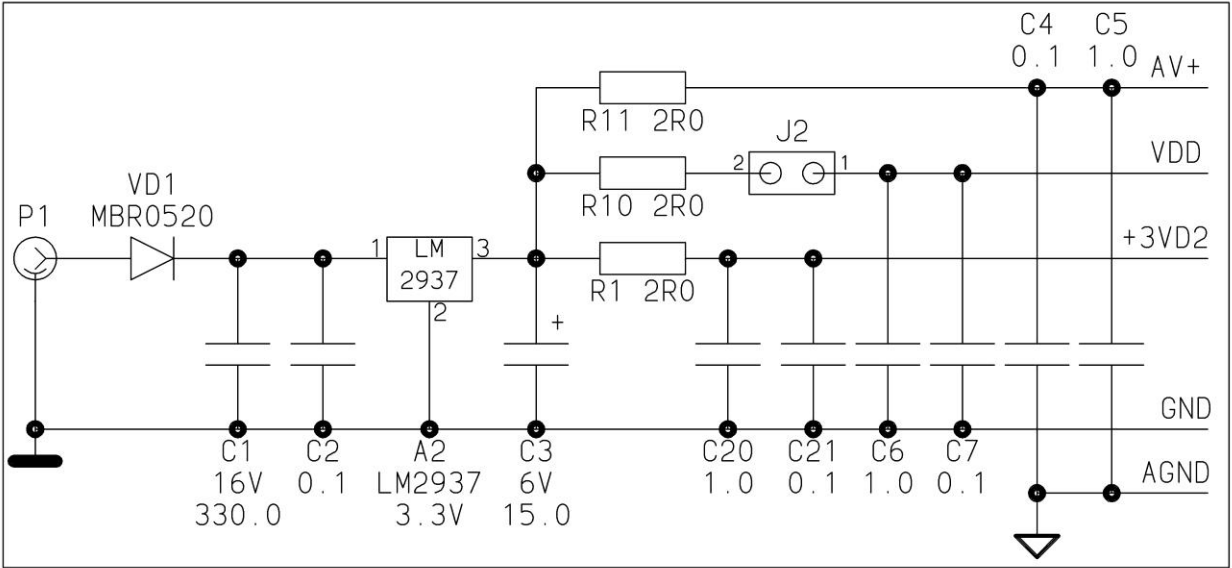


Рис.14.3. Узел стабилизатора напряжения эволюционной платы C8051F020

Узел стабилизатора питания выполнен на микросхеме A2 (LM2937-3.3V) на напряжение 3.3V по стандартной схеме.

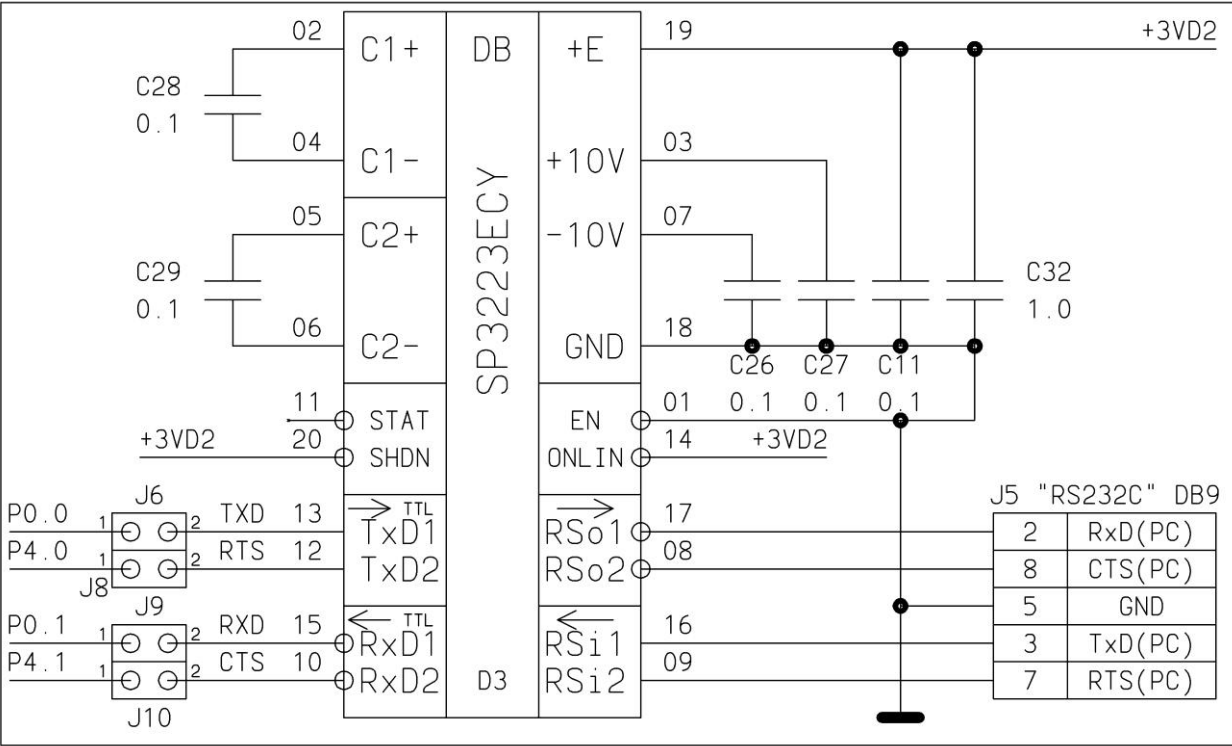


Рис.14.4 Узел интерфейса RS232C эволюционной платы C8051F020

Узел интерфейса RS232C также выполнен по стандартной схеме включения на микросхеме D3 (SP3223ECY).

Перечень штырьковых разъемов платы приведен в таблице 14.5.

Таблица 14.5

Перечень разъемов эволюционной платы эволюционной платы C8051F020

№№ или	Назначение
-----------	------------

название	
J1	Разъем для установки переключки, которая может подключить кнопку SW2 к выводу P3.7 микроконтроллера.
J2	Разъем для установки переключки подключения питания к микроконтроллеру D1
J3	Разъем для установки переключки, которая может подключить светодиод VD3 к выводу P1.6 микроконтроллера.
J4	Разъем для подключения адаптера-программатора JTAG (см. табл.14.2)
J5	Выходной разъем интерфейса RS232C (см. табл.14.6)
J6	Разъем для установки переключки, которая может подключать вход TxD1 микросхемы D3 к выводу P0.0 микроконтроллера.
J7	Отсутствует
J8	Разъем для установки переключки, которая может подключать вход TxD2 микросхемы D3 к выводу P4.0 микроконтроллера.
J9	Разъем для установки переключки, которая может подключать выход RxD1 микросхемы D3 к выводу P0.1 микроконтроллера.
J10	Разъем для установки переключки, которая может подключать выход RxD2 микросхемы D3 к выводу P4.1 микроконтроллера.
J11	Разъем для подключения аналоговых сигналов
J12	Разъем для подключения линий ввода/вывода порта 1
J13	Разъем для подключения линий ввода/вывода порта 4
J14	Разъем для подключения линий ввода/вывода порта 7
J15	Разъем для подключения линий ввода/вывода порта 0
J16	Разъем для подключения линий ввода/вывода порта 3
J17	Разъем для подключения линий ввода/вывода порта 6
J18	Разъем для подключения линий ввода/вывода порта 2
J19	Разъем для подключения линий ввода/вывода порта 5
J20	Клеммник винтовой восьмиконтактный типа 310-08-2-3 для подключения внешних аналоговых сигналов (см. табл.14.7)
J21	Отсутствует
J22	Разъем для установки переключки, которые коммутируют опорное напряжение
J23	Разъем для установки переключки, которая определяет состояние входа MONEN (вход разрешения внутреннего монитора питания микроконтроллера)
J24	Системный трехрядный дюймовый разъем C96 (см. табл.14.4)

В таблице 14.6 приведены контакты выходного разъема интерфейса RS232C эволюционной платы C8051F020.

Таблица 14.6

Выходной разъем интерфейса RS232C эволюционной платы C8051F020

Контакт	Описание
2	RxD (вход PC) – в контроллере соединен с P0.0
3	TxD (выход PC) – в контроллере соединен с P0.1
8	CTS (вход PC) – в контроллере соединен с P4.0
7	RTS (выход PC) – в контроллере соединен с P4.1
5	GND

В таблице 14.7 приведены контакты клеммника винтового, предназначенного для подключения внешних аналоговых сигналов.

Таблица 14.7

Контакты винтового клеммника J20 эволюционной платы C8051F020

Контакт	Описание
1	CP0+ Неинвертирующий вход аналогового компаратора 0

2	CP0-	Инвертирующий вход аналогового компаратора 0
3	DAC0	Выход цифро-аналогового преобразователя 0
4	DAC1	Выход цифро-аналогового преобразователя 1
5	AIN0.0	Вход 0 первого аналого-цифрового преобразователя ADC0
6	AIN0.1	Вход 1 первого аналого-цифрового преобразователя ADC0
7	VREF0	Опорное напряжение
8	AGND	Аналоговая земля

Описанный эволюционный комплект позволяет производить разработку и отладку тестового программного обеспечения, а также на практике изучить особенности работы встроенных аналоговых и цифровых узлов. Кроме того, плата может использоваться в качестве готового узла микроконтроллера в составе составного многоплатного контроллера.

14.1.3. Эволюционный комплект C8051F226

Принципиальная схема эволюционной платы показана на рис.14.5 (приведены оригинальные обозначения элементов)[24].

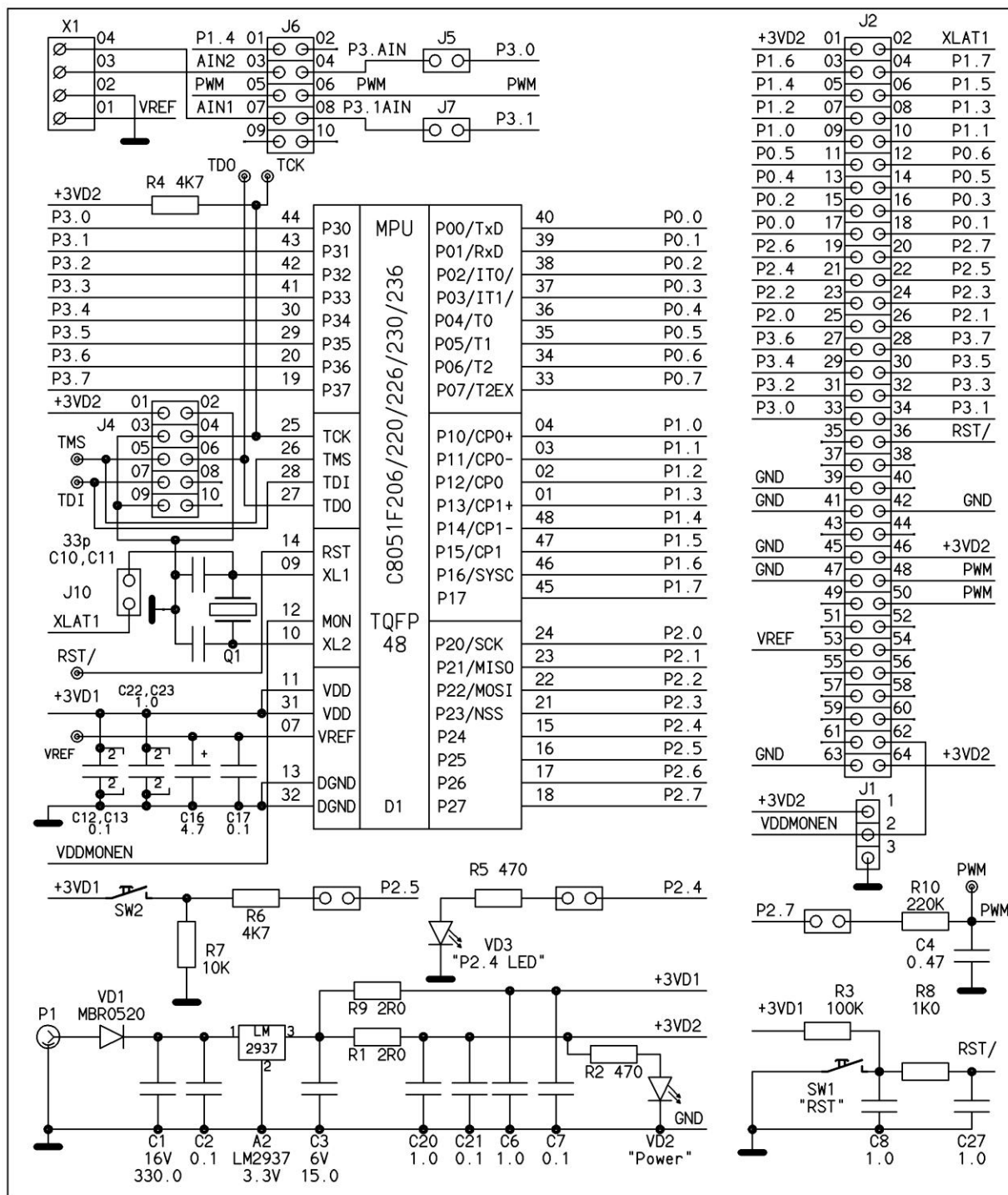


Рис.14.5. Принципиальная схема эволюционной платы C8051F2xx

Эволюционный контроллер содержит следующие функциональные узлы: собственно микроконтроллера D1 (C8051F226), стабилизатор питания A2 (LM2937-3.3) на напряжение 3.3V, светодиод состояния линии P2.4 - VD3, светодиод наличия питания - VD2, кнопка сброса - SW1, кнопка прерывания - SW2. Установочное место под внешний кварцевый генератор Q1 на плате пустое, т.е. пользователь сам устанавливает необходимый тип резонатора.

На плате установлен ряд штырьковых разъемов: Штырьковый двухрядный разъем J4 типа PLD10 (2x5), предназначенный для программирования по интерфейсу JTAG. Штырьковый двухрядный разъем J6 аналоговых входов/выходов PLD10 (2x5). Имеется выходной штырьковый разъем PLD64 (2x32), на который выведены все выводы корпуса микроконтроллера, за исключением четы-

рех сигналов JTAG интерфейса. Кроме того, имеются контактные площадки, соединенные параллельно с соответствующими контактами выходного разъема. Имеются также два небольших макетных поля для установки дополнительных элементов аналоговой и цифровой частей. Размер эволюционной платы равен 85x110 мм². Макетные поля довольно маленькие. Меньшее макетное поле предназначено для аналоговых узлов и имеет 8x8 металлизированных отверстий с шагом 2,54 мм. С двух сторон этого поля располагаются шины земли и питания. Второе поле большего размера предназначено для установки цифровых элементов. Оно имеет 8x20 металлизированных отверстий с шагом 2,54 мм. На всех шинах также с шагом 2,54 мм имеются металлизированные отверстия.

Перечень штырьковых разъемов платы приведен в таблице 14.8.

Таблица 14.8

Перечень разъемов эволюционной платы C8051F2xx

№№ или названия	Назначение
"P2.4"	Разъем для установки перемычки, которая может подключить светодиод VD3 к выводу P2.4 микроконтроллера.
"P2.5"	Разъем для установки перемычки, которая может подключить кнопку SW2 к выводу P2.5 микроконтроллера.
"P2.7"	Разъем для установки перемычки, которая может подключить вывод P2.7 микроконтроллера к внешней выходной RC цепочки, необходимой при работе широтоимпульсного модулятора.
J1	Разъем для установки перемычки, которая определяет состояние входа VDDMONEN (вход разрешения внутреннего монитора питания микроконтроллера)
J2	Выходной двухрядный 64 контактный разъем PLD64 (2x32), на который выведены все выводы микроконтроллера. Рядом с этим разъемом расположены два ряда металлизированных отверстий, каждое из которых соединено с соответствующим контактом разъема J2 (см. табл. 14.9)
J3	Отсутствует.
J4	Разъем для подключения адаптера-программатора JTAG (см. табл. 14.2).
J5	Разъем для установки перемычки, которая может подключить линию порта P3.0 к контакту 4 разъема J6 при настройке линии порта P3.0 на использование в качестве входа 0 (P3.0AIN) аналогового мультиплексора
J6	Разъем для коммутации аналоговых сигналов на клеммник винтовой четырехконтактный X1 типа 310-04-2-3.
J7	Разъем для установки перемычки, которая может подключить линию порта P3.1 к контакту 8 разъема J6 при настройке линии порта P3.1 на использование в качестве входа 1 (P3.1AIN) аналогового мультиплексора
J8	Отсутствует.
J9	Отсутствует.
J10	Разъем для установки перемычки, которая может подключать вывод 02 разъема J2 - внешнего тактового генератора к выводу 09 микроконтроллера.
X1	Клеммник винтовой четырехконтактный типа 310-04-2-3 для подключения внешних аналоговых сигналов
P1	Коаксиальный разъем для подачи внешнего питания +5 - +9В.

Как видно из описания, макетные поля эволюционной платы достаточно малы, поэтому дополнительные элементы можно частично устанавливать на самой эволюционной плате, а частично - на дополнительной макетной плате. На эволюционной плате имеет смысл устанавливать только общесистемные элементы (например, интерфейс RS232/RS485 или интерфейс жидкокристаллического индикатора LCD и т.п.). Для связи с внешней макетной платой предназначен разъем J2, описание контактов которого приведено в таблице 14.9.

Таблица 14.9

Выходной разъем J2 эволюционной платы C8051F2xx

Контакт	Описание	Контакт	Описание
1	+VD цифровое питание	2	XLAT1
3	P1.6	4	P1.7
5	P1.4	6	P1.5
7	P1.2	8	P1.3
9	P1.0	10	P1.1
11	P0.6	12	P0.7
13	P0.4	14	P0.5
15	P0.2	16	P0.3
17	P0.0	18	P0.1
19	P2.6	20	P2.7
21	P2.4	22	P2.5
23	P2.2	24	P2.3
25	P2.0	26	P2.1
27	P3.6	28	P3.7
29	P3.4	30	P3.5
31	P3.2	32	P3.3
33	P3.0	34	P3.1
35		36	RST/
37		38	
39	GND	40	
41	GND	42	GND
43		44	
45	GND	46	+3VD2
47	GND	48	PWM
49		50	PWM
51		52	
53	VREF	54	
55		56	
57		58	
59		60	
61		62	
63	GND	64	+3VD2

Для этой эволюционной платы также можно использовать прием ее модернизации описанный в конце раздела 14.1.1.

14.1.4. Эволюционный комплект C8051F300

Принципиальная схема эволюционной платы C8051F30x [25] разделена на 3 фрагмента. В принципиальных схемах приведены оригинальные обозначения элементов. На рис.14.6 показан узел микроконтроллер с обвязкой. На рис.14.7 показан узел стабилизатора напряжения. И на рис.14.8 показан узел интерфейса RS232C.

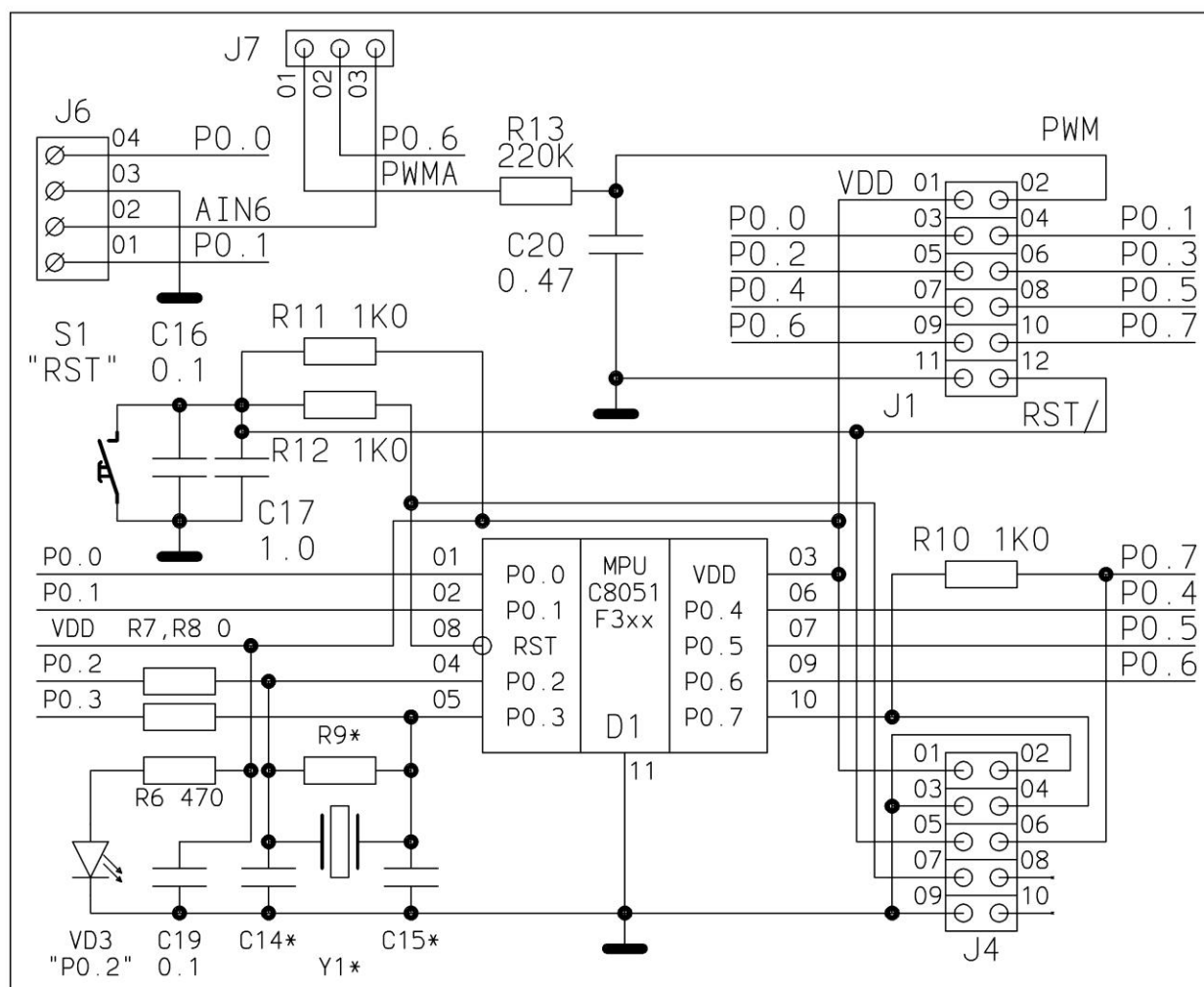


Рис.14.6. Узел микроконтроллера эволюционной платы С8051F30х

Узел микроконтроллера содержит собственно микроконтроллер D1 (C8051F300), кнопку сброса S1, светодиод питания VD3 и набор штыревых разъемов. Перечень штыревых разъемов приведен в таблице 14.14.

Таблица 14.10

Перечень разъемов эволюционной платы C8051F30x

№№	Назначение
J1	Выходной разъем типа PLD12 (2x6) (см. табл. 14.11)
J2	Переключатель для подключения питания к микроконтроллеру
J3	Разъем для установки переключателей, которые предназначены для подключения сигналов от кнопки S2 (к линии порта P0.3), светодиода VD2 (к линии порта P0.2) и сигналов интерфейса RS232C
J4	Разъем для подключения адаптера-программатора EC-2 (см. табл.14.12)
J5	Выходной разъем интерфейса RS232C (см. табл.14.13)
J6	Клеммник винтовой для подключения внешних аналоговых сигналов типа 310-04-2-3
J7	Разъем для установки переключателя, который может подключить на вывод P0.6 либо клеммник винтовой J6/2, либо RC-фильтр для исследования широтно-импульсного модулятора PWM

В таблице 14.11 приведены контакты выходного разъема эволюционной платы C8051F3xx.

Таблица 14.11

Контакты выходного разъема эволюционной платы C8051F30x

Контакт	Описание		Контакт	Описание
---------	----------	--	---------	----------

1	VDD цифровое питание	2	PWM
3	P0.0	4	P0.1
5	P0.2	6	P0.3
7	P0.3	8	P0.5
9	P0.6	10	P0.7
11	GND	12	RST/

Микроконтроллеры семейства C8051F30х, в отличие от микроконтроллеров остальных семейств, программируются через интерфейс C2. При этом также как и для JTAG режима программирования, используется адаптер-программатор EC2. Однако, наименование сигналов программирования отличается от приведенного ранее в таблице 14.2 для интерфейса JTAG, и приведено в таблице 14.12.

Таблица 14.12

Контакты разъема программирования J4 по интерфейсу C2

Контакт	Описание
1	Вход питания от эволюционной платы .3.0-3.6V
2	Цифровая земля
4	C2DAT (для интерфейса JTAG - TCK)
5	TMS
6	TDO
7	C2CLK(для интерфейса JTAG - TDI)
3, 8, 9, 10	Не используются

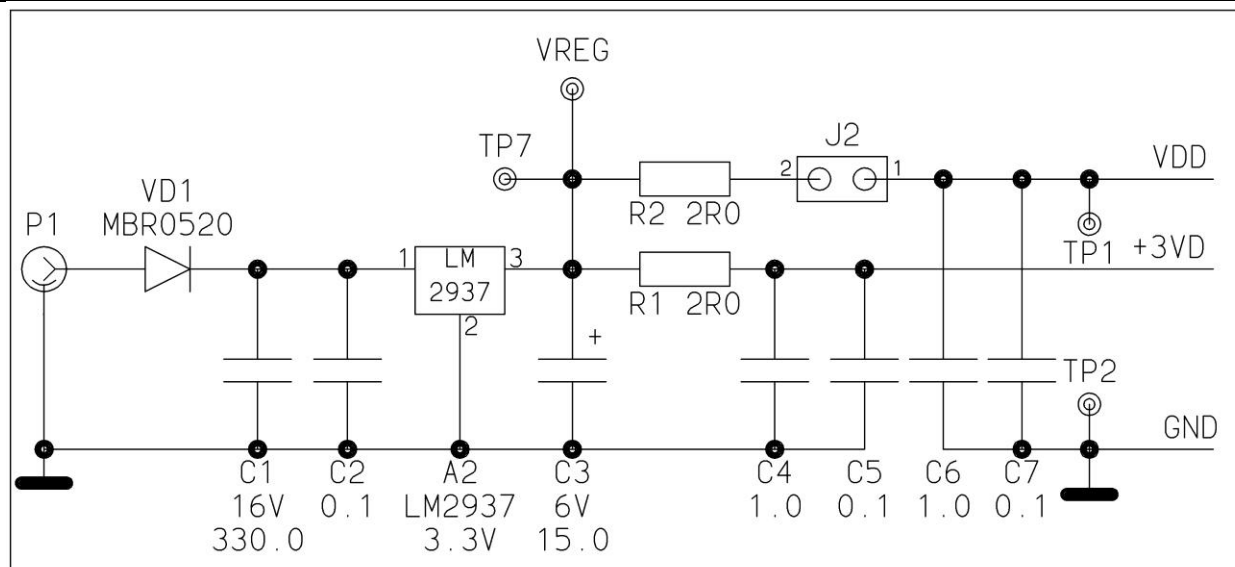


Рис.14.7. Узел стабилизатора напряжения эволюционной платы C8051F3xx

Узел стабилизатора питания выполнен на микросхеме A2 (LM2937-3.3V) на напряжение 3.3V по стандартной схеме.

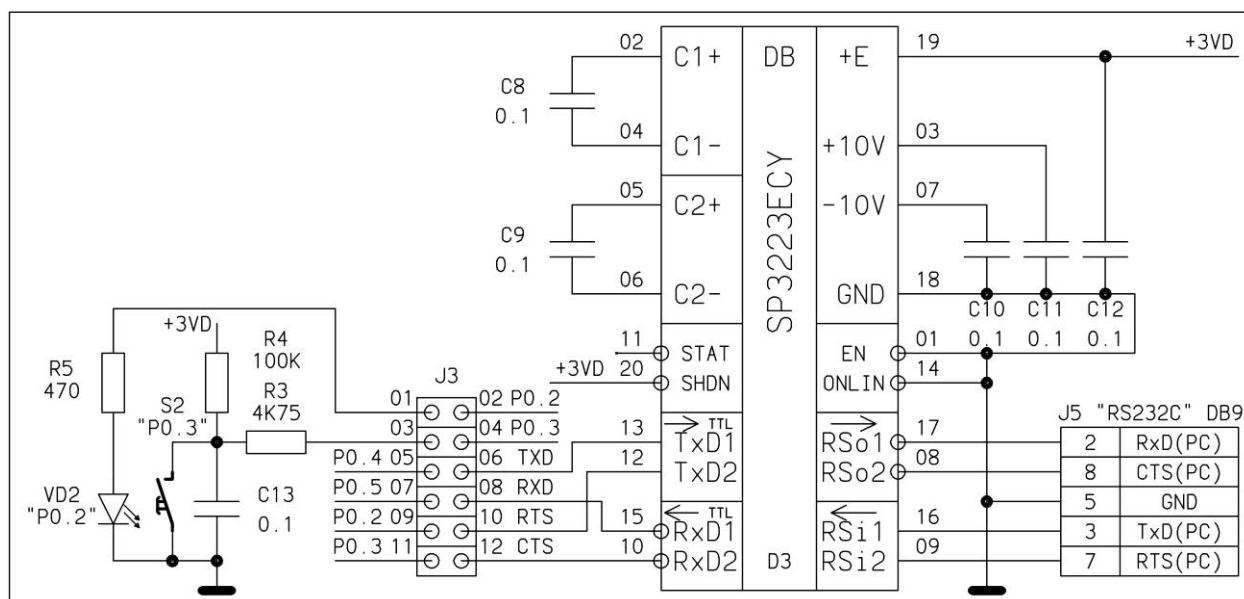


Рис.14.8 Узел интерфейса RS232C эволюционной платы C8051F3xx

Узел интерфейса RS232C также выполнен по стандартной схеме включения на микросхеме D3 (SP3223ECY). Контакты выходного разъема приведены в таблице 14.13.

Таблица 14.13

Выходной разъем интерфейса RS232C эволюционной платы C8051F30x

Контакт	Описание
2	RxD (вход PC) – в контроллере соединен с P0.5
3	TxD (выход PC) – в контроллере соединен с P0.4
8	CTS (вход PC) – в контроллере соединен с P0.3
7	RTS (выход PC) – в контроллере соединен с P0.2
5	GND

Размер эволюционной платы равен 85x103 мм². Макетные поле одно, но довольно большое, имеет 11x26 металлизированных отверстий с шагом 2,54 мм. С двух сторон этого поля располагаются шины земли GND и питания VA+. На всех шинах также с шагом 2,54 мм имеются металлизированные отверстия.

14.1.5. Программатор-адаптер EC2

Конечно же, каждый из специалистов, ознакомившись с содержанием предыдущих разделов и планируя использовать микроконтроллеры Cygnal в своих изделиях, безусловно, задумается над тем, как их программировать. Готовые адаптеры-программаторы, а также необходимое программное обеспечение и кабели, и конечно же сами микроконтроллеры производства фирмы Cygnal можно приобрести в фирме "ATOS"[26]. Однако, многие специалисты могут изъявить желание изготовить такой программатор собственными силами. Конечно же при этом придется тщательно изучить документацию по интерфейсам JTAG и C2, дополнительные сведения о особенностях встроенного в микроконтроллеры интерфейса и написать свою собственную программу (firmware) для микроконтроллера.

Для ознакомления с опытом разработки адаптера-программатора, на рис.14.9 приводится принципиальная схема адаптера-программатора EC-2, любезно предоставленная фирмой Cygnal[27]. Естественно, что схема приводится без оригинального программного обеспечения.

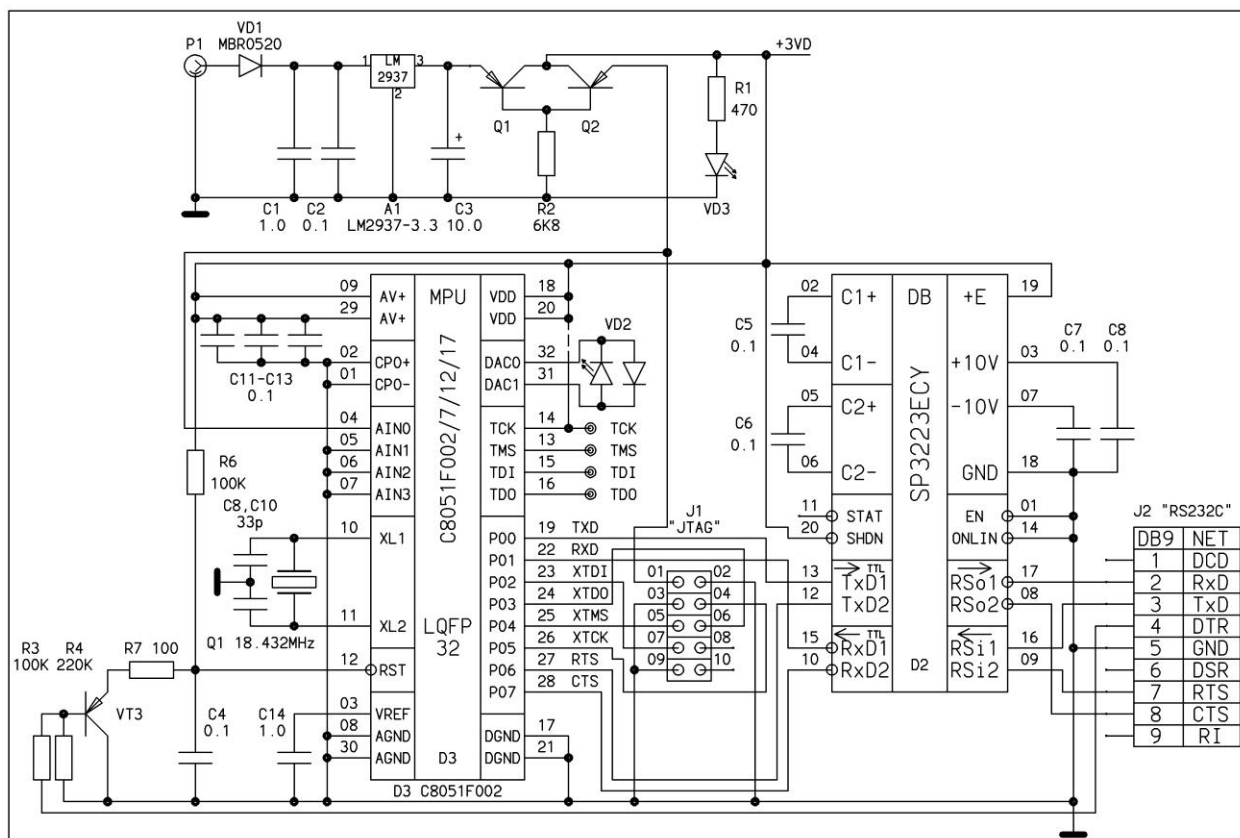


Рис.14.9. Принципиальная схема адаптера-программатора EC-2 фирмы Cygnal

17

Приведенная принципиальная схема полезна и для общего ознакомления, и еще тем, что дает представление об используемых сигналах интерфейса RS232C. Это может понадобиться при изготовлении или приобретении готового узла оптической развязки (изоляция)[28] персонального компьютера от остального подключаемого экспериментального оборудования: адаптера-программатора, эволюционной платы и дополнительных макетных плат. Такой узел желательно использовать в условиях активной работы с измерительными приборами при подключенном персональном компьютере во избежание его выхода из строя. Ведь известно, что большинство персональных компьютеров у нас используются без заземления в связи с особенностями нашей сети электропитания (отсутствия шины заземления), а корпус такого компьютера, в зависимости от используемого в нем блока питания, может находиться под напряжением до ~110V.

14.2. Поддержка пользователей фирмой Cygnal

Кроме технических описаний собственно микроконтроллеров, о которых мы говорили в разделе 1.1., на сайте фирмы Cygnal находится еще множество рекомендаций по применению выпускаемых микроконтроллеров в различных условиях и для различных целей. Многие из этих рекомендаций имеют еще дополнительный zip файл с примерами программирования различных узлов. Список имеющихся рекомендаций по применению приведен в таблице 14.14.

Описание всех приведенных примеров не имеет смысла, т.к. большая часть из них носит узкоспециальный характер и не представляет интереса для широкого круга специалистов. Поэтому, в следующих разделах мы ознакомимся только с наиболее важными и часто используемыми особенностями применения микроконтроллеров фирмы Cygnal, а также приведем некоторые программные примеры, представляющие интерес для большинства читателей.

Таблица 14.14

Список рекомендаций по применению, имеющихся на сайте фирмы Cygnal [29]

Имя	Оригинальное название	
An01.pdf	Configuring the Port I/O Crossbar Decoder	Настройка коммутатора ресурсов
An02.pdf	Configuring the Internal and External Oscillators	Настройка внутреннего и внешнего тактовых

		генераторов
An03.pdf	Using the On-Chip Temperature Sensor	Использование встроенного датчика температуры
An04.pdf	Integrating Keil 8051 Tools into the Cygnal IDE	Встраивание пакета компиляторов «Keil 8051 Tools» в оболочку «Cygnal IDE»
An05.pdf	Programming FLASH through the JTAG Interface	Программирование Flash памяти через JTAG интерфейс
An06.pdf	Interfacing an External SRAM to the C8051F000	Подключение внешней памяти SRAM к микроконтроллерам семейства C8051F0xx
An07.pdf	Implementing 16-Bit PWM Using the PCA	Организация 16-битного ШИМ модулятора на базе программируемого массива/счетчика PCA
An08.pdf	Implementing a Real-Time Clock	Организация таймера реального времени
An09.pdf	Writing to FLASH from Application Code	Запись в Flash память из исполняемой программы
An14.pdf	16-Bit PWM Using an On-Chip Timer	Организация 16-битного ШИМ модулятора на базе таймера
An11.pdf	Using C8051Fxxx in 5 Volt Systems	Использование микроконтроллеров C8051Fxxx в 5-вольтовых системах
An12.pdf	C8051F0xx Bootloader Considerations and Example	Примеры использования режима автозагрузки кодов (Bootloader) в микроконтроллерах C8051F0xx
An13.pdf	Serial Communication with the SMBus	Применение последовательного интерфейса SMBus
An14.pdf	Hand Soldering Tutorial for Fine Pitch QFP Devices	Рекомендации по ручной пайке и демонтажу корпусов QFP.
An15.pdf	Software UART Examples	Применение последовательного интерфейса UART
An16.pdf	Power Management Techniques and Calculation	Рекомендации по управлению питанием
An17.pdf	Using the C8051Fxxx On-Chip Interface Utilities DLL	Использование встроенного интерфейса DLL в C8051Fxxx
An18.pdf	Improving ADC Resolution By Oversampling and Averaging	Повышение точности АЦП методами Oversampling и Averaging
An19.pdf	Calculating Settling Time For Switched Capacitor ADC's	Расчет времени установки АЦП
An20.pdf	FLASH Security User's Guide	Рекомендации по защите кодов
An21.pdf	Converting from the Metalink ASM51 Assembler to the Keil A51 Assembler	Преобразование исходных текстов, написанных на ассемблере «Metalink ASM51» в «Keil A51»
An22.pdf	Annotated 'C' Examples for the 'F02x Family	Примеры программ на «С» для семейства C8051F02x
An23.pdf	Using the DAC as a Function Generator	Использование ЦАП в качестве функционального генератора
An24.pdf	Pin Sharing Techniques for the C2 Interface	Совместное (многофункциональное) использование выводов интерфейса C2 (C8051F3xx)
An25.pdf	Integrating Raisonance 8051 Tools into the Cygnal IDE	Встраивание пакета «Raisonance 8051 Tools» в оболочку «Cygnal IDE»
An26.pdf	Integrating Tasking 8051 Tools into the Cygnal IDE	Встраивание пакета «Tasking 8051 Tools» в оболочку «Cygnal IDE»
An27.pdf	FLASH Programming via the C2 Interface	Программирование Flash памяти через C2 интерфейс
An28.pdf	Software SPI Examples for the C8051F30x Family	Примеры программной эмуляции интерфейса SPI в семейства C8051F30x
An29.pdf	Writing to FLASH from Application Code on 'F30x Devices	Запись в Flash память из исполняемой программы в семейства C8051F30x
An30.pdf	Code Banking Using the Keil 8051 Tools	Рекомендации по переключению банков памяти при использовании "Keil 8051" утилит
An31.pdf	Porting Considerations from 'F02x to 'F12x	Особенности перехода от семейства F02x к семейству F12x

An32.pdf	Integrating Dunfield 8051 Tools into the Cygnal IDE	Встраивание пакета «Dunfield 8051 Tools» в оболочку «Cygnal IDE»
An33.pdf	Embedded Ethernet Reference Design	Рекомендации по созданию систем с Ethernet
An34.pdf	Multiple-Device JTAG Configuration in the Cygnal IDE	Рекомендации по настройке сети интерфейса JTAG
An35.pdf	Cache Optimizations for C8051F12x	Оптимизация Cache в семействе C8051F12x
An37.pdf	Lithium Ion Battery Charger Using C8051F300	Пример реализации зарядного устройства Li аккумуляторов на микроконтроллере F300
An38.pdf	Power Management Techniques for the F30x and F31x	Управление энергопотреблением в семействах F30x и F31x
An39.pdf	USB Firmware Programmer's Guide	Руководство программиста по USB интерфейсу
An40.pdf	Integrating HI-TECH 8051 Tools into the Cygnal IDE	Встраивание пакета «HI-TECH 8051 Tools» в оболочку «Cygnal IDE»
An41.pdf	SMBus Communication for the F30x and F31x	Пример использования интерфейса SMBus (I ² C) в микроконтроллерах F30x и F31x

14.3. Особенности использования микроконтроллеров Cygnal с 5-вольтовым окружением

Все микроконтроллеры фирмы Cygnal работают при напряжениях питания от 2.7V до 3.6V. В тоже время большинство современных микроконтроллерных систем и элементной базы широкого применения работают при напряжении питания 5V. Учитывая это обстоятельство, разработчики фирмы Cygnal обеспечили совместимость всех портов ввода/вывода и некоторых других выводов микроконтроллеров с 5-вольтовым окружением[30].

Очевидно, что при разработке микроконтроллерной системы со смешанным питанием (3V и 5V), следует решить три проблемы:

- Обеспечить требуемые напряжения питания;
- Обеспечить корректную работу 5-вольтового выхода на 3-вольтовый вход;
- Обеспечить корректную работу 3-вольтового выхода на 5-вольтовый вход.

Первая из проблем обычно не вызывает особенных сложностей. Пятивольтовое питание для всей системы формируется обычным образом. Трехвольтовое питание небольшой мощности (напомним, что потребление большинства микроконтроллеров C8051Fxxx не превышает 12mA, и достигает 50mA только для C8051F120 при 100MHz) обычно формируется их пятивольтового с использованием низкошумящих преобразователей LDO (Low Dropout Regulator), которые выпускаются многими фирмами. Примерами могут служить микросхемы LM3940[33], семейство микросхем TPS769xx[31] или ряд микросхем фирмы MAXIM[32].

Вторая проблема согласования 5-вольтового выхода с 3-вольтовым входом микроконтроллеров фирмы Cygnal решена при создании микроконтроллеров. Это означает, что любой пятивольтовый выход может быть напрямую соединен с любым входом портов ввода/вывода и другими входами, для которых специально оговорена совместимость с пяти-вольтовой логикой в документации, например вход RST/.

Третья проблема также учтена разработчиками микроконтроллеров, но требует подключения на каждый выход микроконтроллера, управляющий 5-вольтовым входом внешнего элемента, дополнительный подтягивающий резистор между выходом и источником питания +5V. Кроме того, соответствующий вывод микроконтроллера должен быть переведен в режим работы с «открытым истоком». Вообще проблема выбора оптимального резистора заключается в следующем:

- Необходимо обеспечить минимальный ток через выход микроконтроллера;
- Необходимо обеспечить достаточный ток для срабатывания внешнего входа;
- Необходимо обеспечить рассасывание паразитной емкости внешнего входа для обеспечения максимального быстродействия.

В [30] приведена методика расчета подтягивающего резистора исходя из трех вышеприведенных предпосылок. Там же рекомендуется использовать резистор с номиналом 7K5, при этом обеспечивается ток ~0.6mA и мощность ~3.2mW.

В тоже время следует заметить, что при использовании линии порта ввода/вывода в обычном ключевом режиме (а не с «открытым истоком»), на его выходе генерируются напряжения при-

мерно 0.1V (при низком логическом уровне) и 2.9V (при высоком логическом уровне), что большинство современных цифровых микросхем воспринимает, как ноль и единицу. Иными словами, необходимость использования внешнего подтягивающего резистора и перевод вывода в режим с «открытым истоком» должна быть тщательно проанализирована разработчиком системы исходя из выбранной элементной базы.

14.4. Особенности настройки коммутатора ресурсов Crossbar

Как видно из анализа описаний различных семейств микроконтроллеров, все они оснащены различными модификациями коммутатора ресурсов Crossbar. Микроконтроллеры различных семейств имеют различное количество портов ввода/вывода (от 1 до 8). Напомним, что каждая линия порт ввода/вывода может быть настроена как входная или нормальная выходная линия (ключевой режим), или как линия с "открытым истоком". В дополнение к этому, каждая линия имеет индивидуальную, программно отключаемую, подтяжку уровней "Pull-Up".

Коммутатор ресурсов Crossbar, переключает периферийные внутренние ресурсы на линии ввода/вывода портов 0, 1 и 2. Кроме этого, в некоторых семействах, коммутатор ресурсов коммутирует на выходы портов также и входные аналоговые сигналы.

Настройка выводов внутренних цифровых периферийных узлов осуществляется путем программирования специальных регистров XBR. При этом, установка соответствующего бита каждого внутреннего периферийного узла в этих регистрах приводит к подключению его входов/выходов к выводам порта, которые соединены с физическими выводами корпуса микроконтроллера. Часть внутренних периферийных узлов микроконтроллера может быть запрещена, при этом соответствующие им биты в регистрах XBR должны быть очищены (в низком состоянии).

Входы/выходы включенных периферийных узлов присваиваются выводам портов в соответствии с так называемой таблицей приоритетной перекодировки (Crossbar Priority Decode Table), например на рис.2.5. (для C8051F0xx и C8051F018-9). Примеры настройки коммутатора ресурсов Crossbar приведены также в [34]. В этой таблице по вертикали перечислены выводы всех периферийных устройств, которые могут быть выведены на входы/выводы портов P0...P2. Выводы периферийных устройств перечислены в порядке убывания приоритета. Например, наивысший приоритет имеет интерфейс SMBus, имеющий сигналы SDA и SCL, затем следует интерфейс SPI, имеющий сигналы SCK, MISO, MOSI и NSS, затем последовательный порт UART с сигналами TX и RX, затем программируемый массив-счетчик PCA, имеющий выходы CEX0,...,CEX4, и т.д.

Назначение выводов активных периферийных устройств всегда начинается с младших битов порта P0. Далее, при полном заполнении порта P0, начинается заполнение порта P1, а если его не хватит, то и P2. Заполнение происходит в соответствии с приоритетом со сдвигом в сторону свободных младших выводов. Например, если устройство SMBus активно, то его сигналы занимают младшие разряды порта P0: SDA - P0.0; SCL - P0.1. Если же устройство SMBus пассивно (выключено), а следующее активное устройство - SPI, то его сигналы занимают младшие разряды: SCK - P0.0; MISO - P0.1; MOSI - P0.2; NSS - P0.3. Если и устройство SPI пассивно, а активно устройство UART, то его сигналы займут младшие разряды: TX - P0.0; RX - P0.1. Иными словами, следующие активные устройства всегда занимают более младшие свободные выводы портов P0...P2. Выводы, оставшиеся свободными после назначения активных периферийных устройств могут быть использованы, как программируемые входы/выводы общего назначения.

После назначения активных периферийных устройств необходимо разрешить работу (установить активность) коммутатора ресурсов CrossBar путем установки бита XBARE. Пока CrossBar не активна, все линии портов остаются в режиме ввода вне зависимости от назначений регистров XBRn.

В более сложных семействах микроконтроллеров настройка Crossbar осуществляется аналогично описанной, за исключением того, что в следствие увеличения количества встроенных периферийных узлов может изменяться количество регистров XBRn. Если коммутатор ресурсов семейства осуществляет также коммутацию и аналоговых входов, то они имеют более высокий приоритет. Т.е., если в отдельном регистре аналоговые входы назначены на определенные линии портов, эти линии при распределении выводов цифровых ресурсов пропускаются. Также происходит настройка и в случае наличия встроенного интерфейса внешней памяти, при этом для сигналов WR/, RD/, ALE/, а также шина данных и адреса также резервируются отдельные линии портов или

целые байтовые порты. Процедура настройки портов в более сложных микроконтроллерах показана в соответствующих главах.

14.5. Особенности подключения внешней памяти

При создании сложных микроконтроллерных систем достаточно часто возникает ситуация, когда внутренней оперативной памяти микроконтроллера не хватает для решения поставленной задачи. Многие семейства микроконтроллеров фирмы Cygnal оснащены для этих целей специальным интерфейсом внешней памяти. Примером могут служить семейства C8051F02х, C8051F04х, C8051F12х. В этом случае подключение внешней памяти происходит достаточно просто, как это описано в разделе 5.6.15. На рис.5.10 и 5.11 показана схемотехника подключения внешней памяти к этим семействам в мультиплексированном и немultipлексированном режимах соответственно. Так же, на рис.5.11 и 5.13 показаны временные диаграммы для этих случаев.

Однако, достаточно часто необходимо подключить память или аналогичное внешнее устройство, например, таймер реального времени к микроконтроллеру, не имеющему интерфейса внешней памяти. Рассмотрим этот случай на примере микроконтроллера C8051F005.

На рис.14.10 показан вариант подключения внешней памяти данных к микроконтроллеру. При этом, интерфейс внешней памяти формируется программно. Собственно, в программном формировании интерфейса памяти нет ничего принципиально нового, за исключением первоначальной настройки Crossbar и переключения портов при чтении и записи.

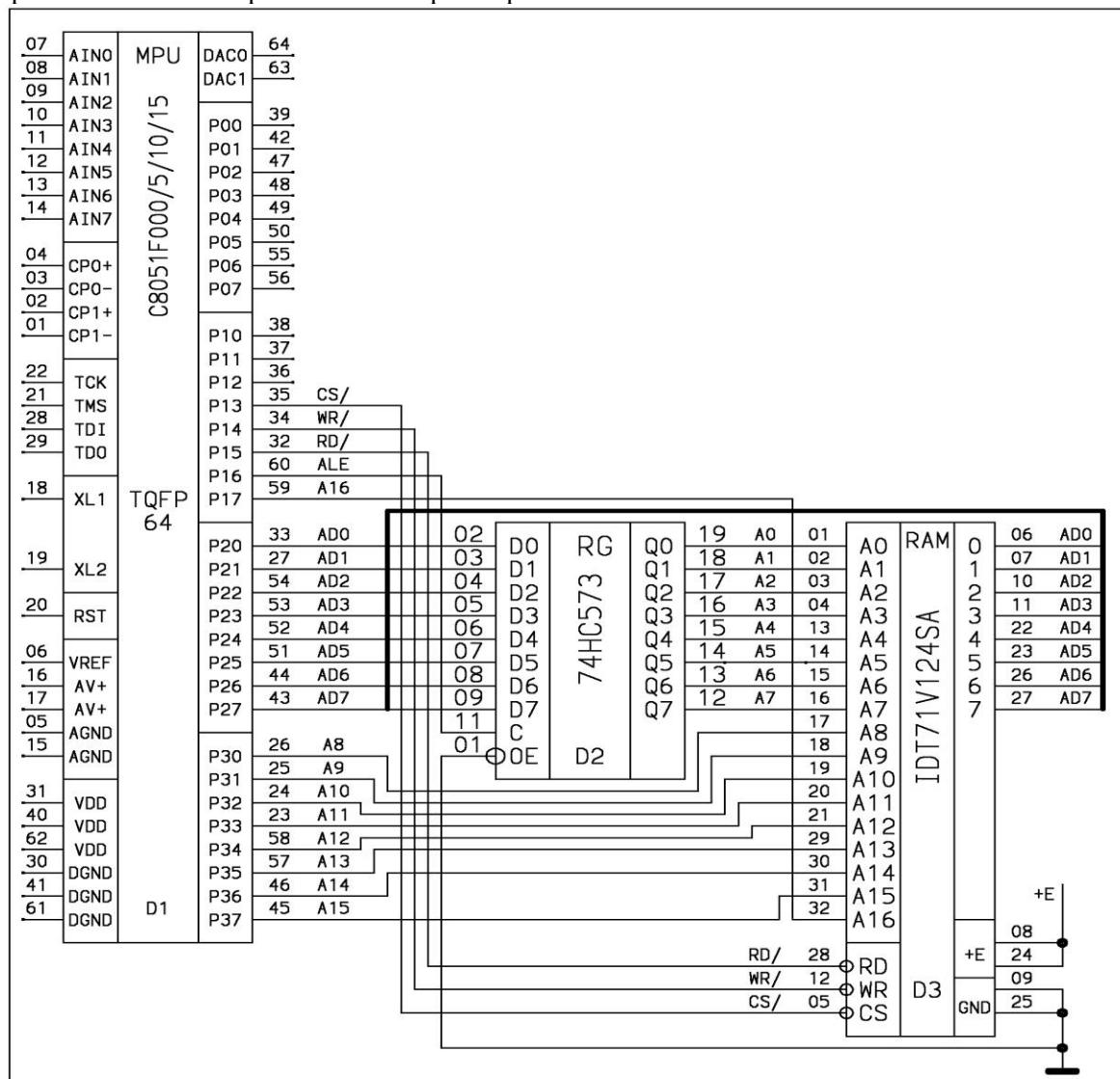


Рис.14.10 Вариант подключения внешней памяти данных к микроконтроллеру C8051F005

Рассмотрим фрагменты настройки подсистемы памяти на языке C (Keil).
Определим используемые линии портов ввода/вывода:

```
#define      byte      unsigned char

#define      DATA      P2
#define      DATACF      PRT2CF
#define      ADDR      P3
#define      ADDRCONF      PRT3CF
sbit      A16      = P1^7;
sbit      ALE      = P1^6;
sbit      RD      = P1^5;
sbit      WR      = P1^4;
sbit      CS      = P1^3;
```

Приведем подпрограмму инициализации интерфейса памяти с выбором страницы 64К:

```
void SRAM_Init (bit Page)
{
    DATACF=0;          // настройка шины данных P2 на ввод
    DATA=0xFF;
    ADDRCONF=0xFF;     // настройка старшего байта адреса P3 на вывод
    ADDR=0;            // установка старшего байта адреса =0
    PRT1CF |= 0xF8;     // установка используемых битов порта P1 на вывод
    A16 = Page;        // страница определяется значением бита Page
    RD=WR=CS=1;        // установка управляющих сигналов в начальное состояние
    ALE=0;
}
```

22

Подпрограмма чтения байта из адреса Address будет иметь вид:

```
byte SRAM_Read (unsigned int Address)
{
    byte    CH;
    CS=0;          // выбираем SRAM
    ADDR=Address&0xFF00/256; // устанавливаем старший байт адреса
    DATACF=0xFF;   // устанавливаем шину AD на вывод
    DATA=Address&0x00FF; // устанавливаем младший байт адреса
    ALE=1;         // защелкиваем младший байт адреса
    ALE=0;
    DATACF=0;      // устанавливаем шину AD на ввода
    DATA=0xFF;
    RD=0;          // разрешаем чтение
    CH=DATA;       // читаем данные у переменную CH
    RD=CS=1;       // отменяем сигналы CS и RD
    return CH;     // возвращаем прочитанный байт
}
```

Подпрограмма записи байта по адресу Address будет иметь вид:

```
void SRAM_Write (unsigned int Address, byte Data)
{
    CS=0;          // выбираем SRAM
    ADDR=Address&0xFF00/256; // устанавливаем старший байт адреса
    DATACF=0xFF;   // устанавливаем шину AD на вывод
    DATA=Address&0x00FF; // устанавливаем младший байт адреса
    ALE=1;         // защелкиваем младший байт адреса
    ALE=0;
    DATA=Data;    // устанавливаем данные на шину
    WR=0;          // записываем данные
}
```

```

WR=1;
DATA CF=0;           // устанавливаем шину AD на ввод
DATA=0xFF;
CS=1;                // отменяем сигнал CS
}

```

Очевидно, что это только один вариант написания подобных подпрограмм. Можно, например, установку битов RD, WR и CS в исходное состояние выделить в отдельную подпрограмму. Можно также отдельно устанавливать старший байт адреса, или наоборот, ввести в подпрограммы чтения и записи еще и установку страницы. Нам, в данном случае, важно было только показать переключения портов на ввода или вывод.

На рис.14.11 показан вариант подключения к микроконтроллеру C8051F005 (D1) таймера реального времени DS12887 (D2), приемопередатчика интерфейса RS485 - MAX3485 (D3) и интерфейса LCD индикатора. В данном случае, формирование интерфейса происходит аналогично с той лишь разницей, что у таймера реального времени нет второго старшего байта.

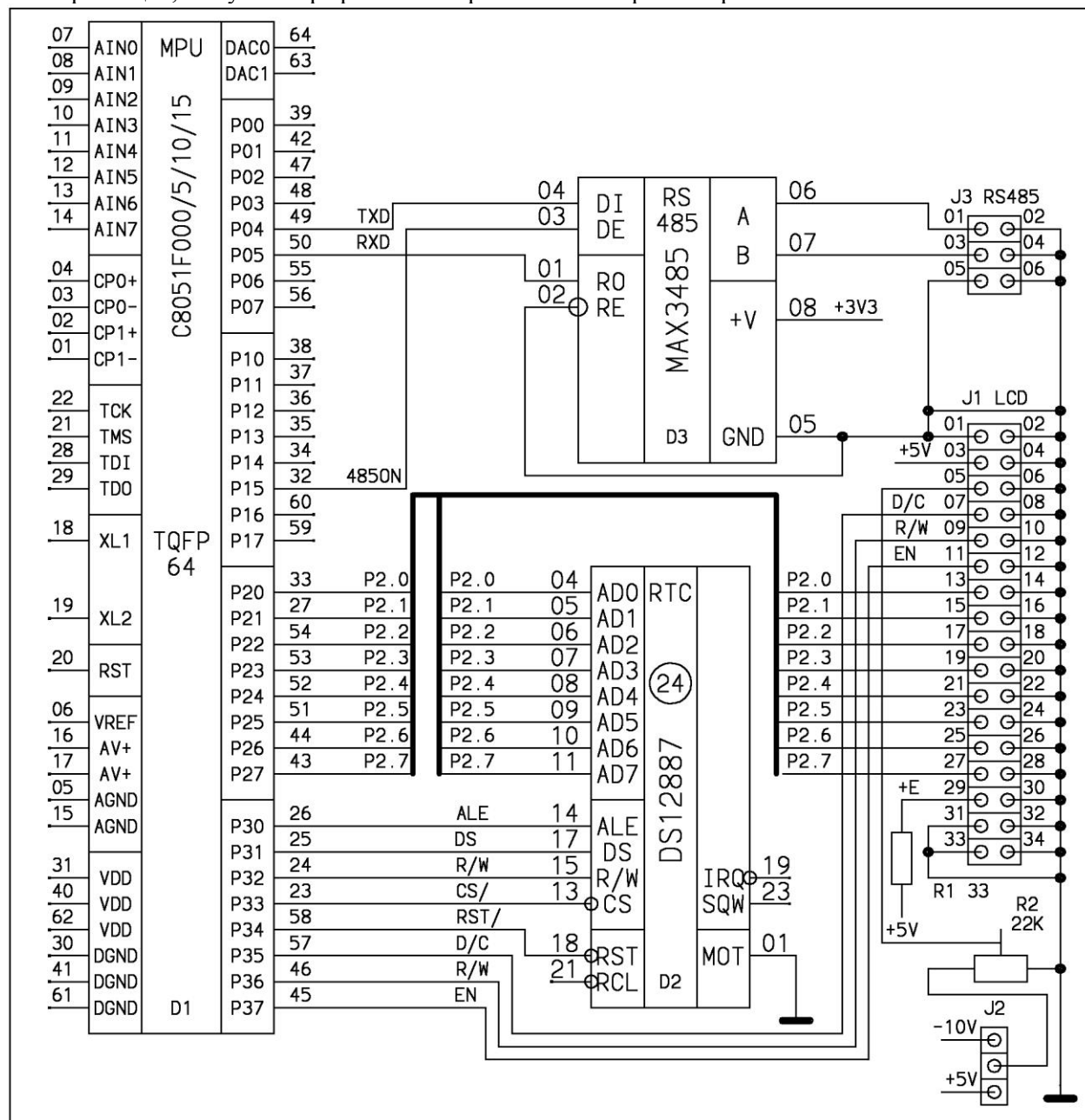


Рис.14.11. Вариант подключения к микроконтроллеру C8051F005 таймера реального времени, приемопередатчика RS485 и интерфейса LCD индикатора.

Уместно будет напомнить, что поскольку в соответствии с таблицей приоритетов Crossbar назначение ресурсов производится последовательно начиная с порта P0, рекомендуется под интерфейс внешней памяти и аналогичных узлов отводить порты P2 и P3, а при необходимости занятия линий порта P1, выбирать разряды начиная со старшего.

14.6. Пример программы с использованием последовательного порта, интерфейса LCD и датчика температуры

В этом разделе мы приводим вариант тестовой программы, иллюстрирующей настройку Crossbar, использующей последовательный порт, датчик температуры, работу с прерываниями и интерфейс жидкокристаллического индикатора LCD. Программа ориентирована на использование процессора C8051F005. Аппаратная реализация интерфейсов LCD и RS485 выполнена в соответствии с рис.14.11. Другие примеры программ, использующие датчик температуры и операции с последовательным портом можно найти в [35-37].

В современных микроконтроллерных системах управления особую популярность получили так называемые командно-информационные сети - CI-LAN (Command-Information LAN). Такие сети широко используются в качестве распределенного интеллектуального интерфейса персонального компьютера в различных системах управления и контроля. Они строятся с использованием интерфейса RS485 или модернизированного интерфейса RS232C. Командно-информационные сети содержат только один персональный компьютер (или главный контроллер) и ряд периферийных контроллеров, называемых станциями. Инициатором всех обменов данными в сети является только персональный компьютер, он выдает команды одной или нескольким станциям. Станции не могут самостоятельно инициировать обмен данными. Они только посылают информацию в ответ на команду персонального компьютера. Отсюда и произошло название – «командно-информационные сети» [38-31].

В командно-информационных сетях данные передаются фреймами или пакетами. Фрейм – это структурированная последовательность байтов. В обычных LAN используются достаточно большие фреймы с переменной длиной и сложной структурой. В CI-LAN фреймы минимизированы и их длина фиксирована. Благодаря этому существенно упрощается программное обеспечение сети как в PC, так и в периферийных станциях. Кроме того, единственный инициатор обмена данными в сети сводит к минимуму возможные конфликтные ситуации, и следовательно, также значительно упрощает протокол их обнаружения и устранения.

В рассматриваемом примере командно-информационной сети MISNET (Middle CI-LAN) используются фреймы длиной 6 байт. Структуры командного и информационного фрейма приведены в таблицах 14.15 и 14.16 соответственно.

Таблица 14.15.

Структура командного фрейма

№№	ИМЯ БАЙТА	СОДЕРЖАНИЕ	НАЗНАЧЕНИЕ
1	IFrame[0]	Preamble	Преамбула. Используется для синхронизации приема фреймов
2	IFrame[1]	Address	Адрес периферийной станции (контроллера)
3	IFrame[2]	Command Code	Код команды
4	IFrame[3]	Data Code Byte 1	Первый байт данных. Если не используется, значение равно 0
5	IFrame[4]	Data Code Byte 2	Второй байт данных. Если не используется, значение равно 0
6	IFrame[5]	Control Sum	Контрольная сумма

Таблица 14.16.

Структура информационного фрейма

№№	ИМЯ БАЙТА	СОДЕРЖАНИЕ	НАЗНАЧЕНИЕ
1	OFrame[0]	Preamble	Преамбула. Используется для синхронизации приема фреймов
2	OFrame[1]	Address	Адрес периферийной станции (контроллера)
3	OFrame[2]	Status Code	Код состояния: 0x00 – Нормальное завершение команды 0xEE – Флаг ошибки
4	OFrame[3]	Data Code Byte 1	Первый байт данных. Если не используется, значение равно 0

			При ошибочной ситуации передается код ошибки.
5	OFram[4]	Data Code Byte 2	Второй байт данных. Если не используется, значение равно 0 При ошибочной ситуации может передаваться дополнительный код ошибки
6	OFram[5]	Control Sum	Контрольная сумма

Приведенные структуры фреймов позволяют передавать до 255 команд. Команда 0x00 обычно используется для передачи статуса сети всем станциям.

Текст приводимой программы написан на языке «С» для компилятора Keil. С целью сокращения объема, в тексте используется только несколько команд.

```

//*****
// Test Program CI-LAN MISNET v.4.0
//*****
#include      <C8051F000.h>                // Файл определения регистров

#include      <stdio.h>

#define      FALSE          0
#define      TRUE           1
#define      WORK           1              // Используется при отладке
#define      byte           unsigned char
#define      VER            4

#define      SYSCLK         11059200       // Частота используемого кварцевого резонатора
#define      BAUDRATE       115200        // Рабочая частота последовательного порта
#define      SAMPLE_RATE    50000         // Частота АЦП
#define      INT_DEC        256

//*****
// Определения битов и байтов, используемых интерфейсом LCD

sbit         LCD_DC         = P3^5;        // Линия выбора режима LCD
sbit         LCD_RW         = P3^6;        // Линия выбора операции чтения/записи LCD
sbit         LCD_EN         = P3^7;        // Линия выборки LCD
#define      LCD_DATA       P2             // Байт шины данных LCD
#define      LCD_DATA_CF    PRT2CF         // Регистр настройки порта шины LCD
#define      LCD_LEN        20             // Длина строки LCD

// Определение бита управления каналом RS485

sbit         ON485          = P1^5;        // Линия управления передатчиком RS485

// Определения 16-битных регистров, используемых аналоговыми узлами

sfr16        DP             = 0x82;        // Регистр указателя данных
sfr16        TMR3RL         = 0x92;        // Регистр автозагрузки таймера 3
sfr16        TMR3           = 0x94;        // Регистр счетчика таймера 3
sfr16        ADC0           = 0xbe;        // Регистр данных ADC0
sfr16        ADC0GT         = 0xc4;        // Регистр верхнего порога ADC0
sfr16        ADC0LT         = 0xc6;        // Регистр нижнего порога ADC0
sfr16        RCAP2          = 0xca;        // Регистр захвата/автозагрузки таймера 2
sfr16        T2             = 0xcc;        // Регистр счетчика таймера 2
sfr16        DAC0           = 0xd2;        // Регистр данных DAC0
sfr16        DAC1           = 0xd5;        // Регистр данных DAC1

//*****

```

```
// Объявления прототипов используемых функций
```

```
void          main          (void);          // Главная функция
void          Monitor       (void);          // Монитор команд
void          Start_Init    (void);          // Начальная инициализация
void          General_Routines (void);        // Подпрограммы общего назначения
                                                    // Пустая функция

void          Time          (unsigned Delay); // Задержка
void          SendB         (byte SendByte); // Передать байт по UART
byte          GetB          (void);          // Принять байт по UART
void          Send_Frame    (byte CODE, byte DATA1, byte DATA2); // Передать фрейм
void          Get_Frame     (void);          // Получить фрейм
byte          Output_CS     (byte NUM);      // Расчет контрольной суммы командного фрейма
byte          Input_CS      (byte NUM);      // Расчет контрольной суммы фрейма ответа
void          WDT           (void);          // Перезапуск охранного таймера WDT

void          SYSClk_Init   (void);          // Инициализация системного генератора
void          PORT_Init     (void);          // Инициализация портов
void          UART0_Init    (void);          // Инициализация последовательного порта
void          ADC0_Init     (void);          // Инициализация ADC0
void          Timer3_Init   (int counts);    // Инициализация таймера 3

void          ADC0_ISR      (void);          // Опрос ADC0
void          Get_Temperature(void);

                                                    // Расчет температуры

void          LCD_Init      (void);          // Инициализация LCD
byte          Get_Data      (void);          // Получение данных с шины
void          Set_Data      (byte CH);       // Установить данные на шину
void          LCD_Time      (char Delay);    // Задержка для LCD
void          LCD_Wait      (void);          // Ожидание готовности
void          LCD_Put       (byte DB);       // Поместить байт в LCD
void          LCD_SetCom    (byte DB);       // Поместить команду в LCD
void          LCD_Reset     (void);          // Сбросить LCD
void          LCD_Clear     (void);          // Очистить LCD и вернуть курсор на адрес 0
void          LCD_Home      (void);          // Вернуть курсор на адрес 0
void          LCD_ON        (void);          // Включить дисплей
void          LCD_OFF       (void);          // Выключить дисплей
void          LCD_Mode      (byte DB);       // Курсор вправо/влево (0x02) & Сдвиг (0x01)

void          LCD_DCB       (byte DB);       // Дисплей (0x04), курсор (0x02), мигание (0x01)
                                                    // включить/выключить

void          LCD_Move      (byte DB);       // Дисплей/Курсор(0x08) &
                                                    // Сдвиг вправо/влево (0x04)

void          LCD_AddrSet   (byte ADDR);     // Установить адрес RAM дисплея
void          LCD_AddrStr   (byte X, byte Y); // Установить адрес строки
byte          LCD_ReCode    (char CH);       // П/программа перекодировки DOS->LCD
void          LCD_WriteData (byte DAT);      // Записать символ без перекодировки
void          LCD_WriteCData (byte DAT);     // Записать символ с перекодировкой
void          LCD_String    (char *s);       // Записать строку (с перекодировкой)
void          LCD_XYString  (byte X, byte Y, char *s); // Записать строку (с перекодировкой)
byte          LCD_ReadData  (void);          // Прочитать символ
byte          LCD_AddrGet   (void);          // Получить адрес RAM LCD

//*****
code byte     ADDR = 0x01;                    // Адрес станции в CI-LAN
```

// Таблица перекодировки DOS->LCD

```
code byte      LCDCODE[125] = {    0xdb,  0xd1,  0xdc,  0xd9,  0x41,  0xA0,  0x42,  0xa1,
                                0xe0,  0x45,  0xA3,  0xA4,  0xa5,  0xa6,  0x4b,  0xa7,
                                0x4d,  0x48,  0x4f,  0xa8,  0x50,  0x43,  0x54,  0xa9,
                                0xaa,  0x58,  0xe1,  0xab,  0xac,  0xe2,  0xad,  0xae,
                                0x62,  0xaf,  0xb0,  0xb1,  0x61,  0xb2,  0xb3,  0xb4,
                                0xe3,  0x65,  0xb6,  0xb7,  0xb8,  0xb9,  0xba,  0xbb,
                                0xbc,  0xbd,  0x6f,  0xbe,  0x70,  0x63,  0xbf,  0x79,
                                0xe4,  0x78,  0xe5,  0xc0,  0xc1,  0xe6,  0xc2,  0xc3,
                                0xc4,  0xc5,  0xc6,  0xc7};
```

//**

```
data long      REZ;                // глобальная десятичная величина ADC0
data byte      IFrame[7];          // Командный входной фрейм
data byte      OFrame[7];          // Информационный выходной фрейм
data byte      Buffer[20];          // Буфер данных
data byte      PTR;                // Указатель общего назначения
data byte      Byte;               // Входной байт
data byte      FRAME_PTR;          // Указатель фрейма
data byte      FRAME_RDY;          // Флаг готовности фрейма
data int       TEMP_INT;           // Целая часть температуры
data int       TEMP_FRAC;          // Дробная часть температуры
data long      TEMPER;             // Температура в сотых градуса
```

//**

```
void Time (unsigned Delay) // Подпрограмма временной задержки
{
    while (Delay--)
        WDT();
}
```

//**

```
void SendB (byte SendByte) // Подпрограмма передачи байта в UART
{
    #if      WORK
        while (!TI) WDT();    // Ожидание завершения передачи предыдущего байта
    #endif
    TI=0;
    Time (1);
    SBUF=SendByte;
}
```

//**

```
byte GetB (void) // Подпрограмма приема байта из UART
{
    byte InputByte;

    while (!RI) {WDT();}    // Ожидание завершения приема
    InputByte = SBUF;
    RI = 0;
    return InputByte;
}
```

//**

```
void Get_Frame (void) // Подпрограмма приема фрейма
{
    Byte=GetB();            // Получить байт
```

```

        if (!FRAME_PTR)                // Если байт первый, сравнить с кодом преамбулы
        {
            if (Byte==0xAA) goto Next;
            else                return;
        }
Next:
    IFrame[FRAME_PTR++]=Byte;    // Записать байт в буфер фрейма команды
    if (FRAME_PTR>=6)
    {
        ES=0;                    // Запретить прерывания UART
        FRAME_RDY=1;            // Выставить флаг готовности фрейма
        TI=1;
    }
}
//*****
void UART0_INT (void) interrupt 4 // Прерывание последовательного порта UART
{
    Get_Frame ();
}

//*****
byte Input_CS (byte NUM) // Подпрограмма расчета контрольной суммы входного фрейма
{
    byte CS=0;
    for (PTR=0; PTR<NUM; PTR++)
        CS=(CS+IFrame[PTR])&0xFF;
    return CS;
}

//*****
byte Output_CS (byte NUM) // Подпрограмма расчета контрольной суммы выходного фрейма
{
    byte CS=0;
    for (PTR=0; PTR<NUM; PTR++)
        CS=(CS+OFrame[PTR])&0xFF;
    return CS;
}

//*****
void Send_Frame (byte CODE, byte DATA1, byte DATA2) // Подпрограмма отправки фрейма
{
    OFrame[0]=0xAA;
    OFrame[1]=ADDR;
    OFrame[2]=CODE;
    OFrame[3]=DATA1;
    OFrame[4]=DATA2;
    OFrame[5]=Output_CS(5);
    Time (23);
    ON485=1;
    Time (20);
    for (PTR=0; PTR<6; PTR++)
    {
        SendB(OFrame[PTR]); Time (10);
    }
    #if WORK
    while (!TI) WDT();                // Ожидание завершения отправки последнего байта
    #endif
    Time (20);
    ON485=0;
    TI=0;
}

```



```

}

//*****
void Monitor (void) // Подпрограмма монитора команд
{
    if (IFrame[1]!=ADDR)
        {LCD_XYString (1,1,"Address Error"); return;} // Сообщение о чужом адресе
    if (IFrame[5]!=Input_CS(5))
        {LCD_XYString (1,1,"CS Error"); return;} // Сообщение об ошибке контрольной суммы
    switch (IFrame[2])
    {
        case 0x01:    LCD_Clear ();           // Команда 1 - Очистить LCD
                     Send_Frame (0,0,1);
                     break;
        case 0x02:    LCD_Reset ();           // Команда 2 - Сбросить LCD
                     Send_Frame (0,0,2);
                     break;
        case 0x03:    LCD_Home ();            // Команда 3 – Курсор в позицию 0
                     Send_Frame (0,0,3);
                     break;
        case 0x04:    LCD_ON ();              // Команда 4 – LCD включить
                     Send_Frame (0,0,4);
                     break;
        case 0x05:    LCD_OFF ();             // Команда 5 – LCD выключить
                     Send_Frame (0,0,5);
                     break;
        case 0x06:    LCD_WriteData (IFrame[3]); // Команда 6 – Записать символ в LCD
                     Send_Frame (0,0,6);
                     break;
        case 0x07:    LCD_AddrStr (IFrame[3],IFrame[4]); // Команда 7 – Установить адрес LCD
                     Send_Frame (0,0,7);
                     break;
        case 0x08:    Get_Temperature();      // Передать температуру и вывести на LCD
                     Send_Frame (0,TEMP_INT,TEMP_FRAC);
                     sprintf(Buffer,"Temperature is %+02d.%02d\n", TEMP_INT, TEMP_FRAC);
                     LCD_XYString (2,1,Buffer);
                     break;

        // Здесь можно вставить произвольный набор команд с общим количеством 255

        default:      // Если команда отсутствует, на LCD выводится сообщение об ошибке
                     // и передается фрейм ошибки 2

                     LCD_XYString (1,1,"Command Error");
                     Send_Frame (0xEE,0,2);
                     break;
    }
    Time(2);
}

//*****
byte Get_Data (void) // Получить байт с шины данных LCD и другой периферии, если она есть, например
{
    // если установлен таймер реального времени или внешняя память данных
    LCD_DATA_CF=0;
    LCD_DATA=0xFF;
    return LCD_DATA;
}

//*****

```

```

void Set_Data (byte CH) // Выставить данные на шину
{
    LCD_DATA_CF=0xFF;
    LCD_DATA=CH;
}

//*****
void LCD_Init (void) // Инициализация LCD
{
    LCD_EN=0;
    LCD_DC=0;
    LCD_RW=0;
    Time(250);
    Time(250);
    LCD_Put (0x38);          // Function (0x20) + 8-bit (0x10) + Two-Line (0x08)
    Time(200);
    LCD_Put (0x38);
    Time(100);
    LCD_Put (0x38);          // Function (0x20) + 8-bit (0x10) + Two-Line (0x08)
    Time(100);
    LCD_Reset();
}

//*****
void LCD_Put (byte DB) // Передать байт данных в LCD
{
    LCD_DC=0;

    LCD_RW=0;
    LCD_EN=1;
    Set_Data (DB);
    Time (40);
    LCD_EN=0;
}

//*****
void LCD_SetCom (byte DB) // Передать байт команды в LCD
{
    LCD_EN=1;
    LCD_DC=0;
    LCD_RW=0;
    LCD_Put (DB);
    LCD_Wait ();
}

//*****
void LCD_Wait (void) // Ожидать готовность LCD
{
    LCD_DC    =0;
    LCD_EN    =1;
    LCD_RW    =1;
    while (Get_Data() & 0x80 != 0) WDT();
    LCD_EN    =0;
    Time (40);
    LCD_RW    =0;
}

//*****
void LCD_Reset (void) // Сброс LCD

```

```

{
    LCD_SetCom (0x38);    // Function (0x20) + 8-bit (0x10) + Two-Line (0x08)
    LCD_SetCom (0x0E);    // Instruction (0x08) + Display ON (0x04) + Cursor ON (0x02)
    LCD_SetCom (0x06);    // Instruction (0x04) + Move Right (0x02)
}

//*****
void LCD_Clear (void)    {LCD_SetCom(0x01);} // Очистить LCD
void LCD_Home (void)    {LCD_SetCom(0x02);} // Курсор в позицию 0
void LCD_ON (void)      {LCD_SetCom(0x0C);} // LCD включить
void LCD_OFF (void)     {LCD_SetCom(0x08);} // LCD выключить

//*****
void LCD_AddrStr (byte X, byte Y) // Установить адрес курсора LCD
{
    byte    Offset[4] = {0x00,0x40,0x14,0x54};

    LCD_SetCom (0x80|Offset[Y]|(X & 0x3F));
}

//*****
byte LCD_ReCode (char CH) // Подпрограмма перекодировки DOS -> LCD
{
    if      ((CH<0x20)&(CH!=0x0d))    return 0x20;
    else if (CH==0x0d)                return 0x7e;
    else if ((CH<0xA7)&(CH<123))      return CH;
    else if ((CH>=123)&(CH<176))      return LCDCODE[CH-124];
    else if ((CH>=224)&(CH<240))      return LCDCODE[CH-172];
    else if (CH>240)                  return 0x20;
}

//*****
void LCD_WriteData (byte DAT) // Записать символ без перекодировки
{
    LCD_EN=1;
    LCD_RW=0;
    LCD_DC=1;
    Set_Data (DAT);
    Time (40);
    LCD_EN=0;
    LCD_Wait ();
    LCD_DC=0;
}

//*****
void LCD_WriteCData (byte DAT) // Записать символ с перекодировкой
{
    LCD_EN=1;
    LCD_RW=0;
    LCD_DC=1;
    Set_Data (LCD_ReCode(DAT));
    Time (40);
    LCD_EN=0;
    LCD_Wait ();
    LCD_DC=0;
}

//*****
byte LCD_ReadData (void) // Прочитать символ
{

```

```

byte    DAT;

    LCD_EN=1;
    LCD_RW=1;
    LCD_DC=1;
    DAT = Get_Data ();
    Time (40);
    LCD_EN=0;
    LCD_Wait ();
    LCD_RW=0;
    LCD_DC=0;
    return DAT;
}

//*****
byte LCD_AddrGet (void) // Прочитать текущий адрес RAM
{
byte    DAT;

    LCD_EN      =1;
    LCD_DC      =0;
    LCD_RW      =1;
    DAT=Get_Data ();
    LCD_EN      =0;
    LCD_RW      =0;
    return  DAT&0x7F;
}

//*****
void LCD_String (char *s) // Записать строку, начиная с текущего адреса
{
int      i;

    for (i=0;i<LCD_LEN;i++)
    {
        if (s[i]=='\n') break;
        if (s[i]==0) break;
        LCD_WriteCData (s[i]);
    }
}

//*****
void LCD_XYString (byte X, byte Y, char *s) // Записать строку с адреса X.Y
{
    LCD_AddrStr (X,Y);
    LCD_String (s);
}

//*****
void Get_Temperature (void) // Подпрограмма получения и вычисления температуры
{

    EA = 0;                // запретить прерывания
    TEMPER = REZ;          // получить последнее измерение
    EA = 1;                // разрешить прерывания

    TEMPER = TEMPER - 41857;
    TEMPER = (TEMPER * 100L) / 154;
    TEMP_INT = TEMPER / 100;
}

```

```

    TEMP_FRAC = TEMPER - (TEMP_INT * 100);
}

//*****
void SYSCLK_Init (void) // Инициализация системного тактового генератора
{
    OSCXCN = 0x67;           // Запустить внешний генератор 11.0592MHz
    while (!OSCXCN&0x80)
    WDT();                   // Ожидать установления XTLVLD
    OSCICN = 0x88;           // Выбрать внешний генератор для SYSCLK
                             // и разрешить работу детектора пропадания частоты
}

//*****
void PORT_Init (void) // Инициализация портов и периферии
{
    XBR0 = 0x16;             // Разрешить периферию SPI, UART0, PCA0-1;
    XBR1 = 0x00;
    XBR2 = 0x40;             // Включить Crossbar
    PRT0CF = 0x55;           // Назначить входы/выходы порта 0
                             // P0.7 <- CEX1
                             // P0.6 <- CEX0
                             // P0.5 <- RxD
                             // P0.4 <- TxD
                             // P0.3 <- NSS
                             // P0.2 <- MOSI
                             // P0.1 <- MISO
                             // P0.0 <- SCK

    PRT1CF |= 0x20;          // Назначить входы/выходы порта 1
                             // P1.5 <- 485ON

    PRT2CF = 0xFF;           // Ключевой режим P2 - Push-pull Outputs - Data Bus
    PRT3CF = 0xFF;           // Ключевой режим P3 - Push-pull Outputs
                             // P3.7 <- LCD_EN
                             // P3.6 <- LCD_RW
                             // P3.5 <- LCD_DC
}

//*****
void UART0_Init (void) // Инициализация последовательного порта
{
    SCON = 0x52;             // SCON: mode 1, 8-bit UART, enable RX
    TMOD = 0x20;             // TMOD: timer 1, mode 2, 8-bit reload
    TH1 = -(SYSCLK/BAUDRATE/16); // set Timer1 reload value for baudrate
    TR1 = 1;                 // start Timer1
    CKCON |= 0x10;           // Timer1 uses SYSCLK as time base
    PCON |= 0x80;            // SMOD = 1
    TI = 1;                  // Indicate TX ready
}

//*****
void WDT (void) // Перезапуск охранного таймера
{
    WDTCN = 0xA5;
}

//*****

```



```

// Инициализация ADC0. ADC0 настраивается на использование переполнения таймера 3 для формирования
// частоты преобразования и генерирования прерывания после завершения преобразования
void ADC0_Init (void)
{
    ADC0CN = 0x05;
    REF0CN = 0x07;           // Разрешить температурный датчик и встроенный
                             // источник опорного напряжения VREF
    AMX0SL = 0x0f;           // Выбрать TEMP датчик в мультиплексоре
    ADC0CF = 0x80;           // Задать частоту преобразования ADC = SYSCLK/16
    ADC0CF |= 0x01;          // Установить коэффициент усиления PGA = 2
    EIE2 |= 0x02;            // Разрешить прерывания ADC
}

//*****
// Инициализация таймера 3 в режим автозагрузки без генерации прерывания и использовать для счета
// SYSCLK
void Timer3_Init (int counts)
{
    TMR3CN = 0x02;           // Остановить таймер, стереть флаг TF3, использовать SYSCLK
    TMR3RL = -counts;        // Установить величину автозагрузки
    TMR3 = 0xffff;           // Перезагрузить
    EIE2 &= ~0x01;           // Запретить прерывания
    TMR3CN |= 0x04;          // Запустить таймер
}

//*****
// Подпрограмма прерывания завершения преобразования ADC0. По прерыванию берется текущий
// результат ADC0, прибавляется к десятичному значению <accumulator>, уменьшается локальный
// десятичный указатель <int_dec>. Когда он достигает 0, десятичному значению <accumulator>
// перезаписывается в глобальную переменную <REZ>.
void ADC0_ISR (void) interrupt 15 using 1
{
    static unsigned int_dec=INT_DEC;
    static long accumulator=0L;
    ADCINT = 0;              // Очистить индикатор завершения преобразования ADC
    accumulator += ADC0;      // Прибавить результат к полученному ранее ADC
    int_dec--;               // Изменить указатель
    if (int_dec == 0)
    {
        int_dec = INT_DEC;   // Переустановить счетчик
        REZ = accumulator >> 8;
        accumulator = 0L;    // Очистить аккумулятор
    }
}

//*****
void Start_Init (void) // Подпрограмма начальных установок
{
    SYSCLK_Init ();        // Инициализировать системный тактовый генератор
    PORT_Init ();           // Инициализировать порты и периферию
    LCD_Init ();            // Инициализировать LCD
    UART0_Init ();          // Инициализировать UART
                           // Инициализировать таймер 3

    Timer3_Init (SYSCLK/SAMPLE_RATE);
    ADC0_Init ();           // Инициализировать ADC
    ADCEN = 1;              // Разрешить ADC

    ON485=0;                // Закрыть передатчик RS485
}

```

```

    WDT();
    RI=0;
    TI=0;
    EA=1;                // Разрешить прерывания

    LCD_Reset ();        // Сбросить LCD
    LCD_ON ();           // Включить LCD
    LCD_Clear ();        // Очистить LCD
                        // Выдать приветствие
    LCD_XYString (0,1,"Start CI_LAN MISNET v.4.0");
    WDT();
}

/*****
void General_Routines (void)
{
// Пустая функция. Здесь могут находиться подпрограммы опроса портов, регистров, например, клавиатуры.
}

*****/

void main (void) // Главная функция
{
    Start_Init ();       // Начальная инициализация
    FRAME_RDY=0;         // Очистить флаг готовности фрейма
    FRAME_PTR=0;         // Очистить указатель фрейма
    ES=1;                // Разрешить прерывания UART
    while (1)
    {
        WDT();           // Перезапустить охранный таймер
        General_Routines (); // Подпрограммы общего назначения
        if (FRAME_RDY==1) // Если получен фрейм
        {
            Monitor ();   // Распознать и выполнить команду
            ES=1;         // Разрешить прерывания UART
            FRAME_RDY=0; // Очистить флаг готовности фрейма
            FRAME_PTR=0; // Очистить указатель фрейма
        }
    }
}

*****/

```

14.7. Области применения и примеры использования

Микроконтроллеры фирмы Cygnal обладают столь разнообразным набором периферии, что могут использоваться практически во всех областях микроконтроллерной техники. Базовые модели каждого из семейств, т.е. обладающие полным комплектом периферии, безусловно, являются наиболее мощными, но и наиболее дорогими. Их использование целесообразно при решении тех задач, в которых необходимо использование большинства имеющейся периферии, в первую очередь – аналоговой периферии. “Усеченные” модели семейств, т.е. модели с уменьшенным количеством портов или аналоговых узлов могут очевидно использоваться в более простых задачах. Вообще говоря, все микроконтроллеры Cygnal представляют собой мощные микросистемы сбора аналоговых данных. Поэтому, в первую очередь, они перспективны для применения в различных измерительных и управляющих технологических системах и сетях. Однако высокие показатели производительности этих микроконтроллеров, удобство программирования и отладки, обилие цифровой периферии, объемы памяти программ и данных, все это делает их привлекательными и для чисто цифровых систем, не связанных с измерением и обработкой аналоговых сигналов.

На сайте фирмы Cygnal выложено несколько pdf файлов с описаниями различных примеров проектов применения продукции (см. табл.14.17).

Примеры применения продукции фирмы Sygnal

Имя файла	Наименование	Пример использования
tb002.pdf	AC Power Meter	Измеритель активной мощности
tb003.pdf	Battery Charger	Система контроля заряда аккумуляторов
tb004.pdf	System Manager	Поддержание параметров среды в системе
tb005.pdf	Gas Detection Monitor	Газовый детектор
tb006.pdf	Uninterruptible Power Supply	Бесперебойный блок питания
tb007.pdf	Weigh Scale	Электронные весы
tb008.pdf	Position Sensor	Датчик положения

Все перечисленные проекты не используют в полную силу всех богатейших возможностей описанных микроконтроллеров. Пожалуй, в наибольшей степени все резервы микроконтроллеров могут быть использованы только в современных универсальных технологических контроллерах, ориентированных на работу в составе рассредоточенных систем управления и контроля, например в составе CI LAN. В качестве примера, приведем принципиальную схему одного мощного универсального технологического контроллера на базе микроконтроллера C8051F020 (см.рис.14.12).

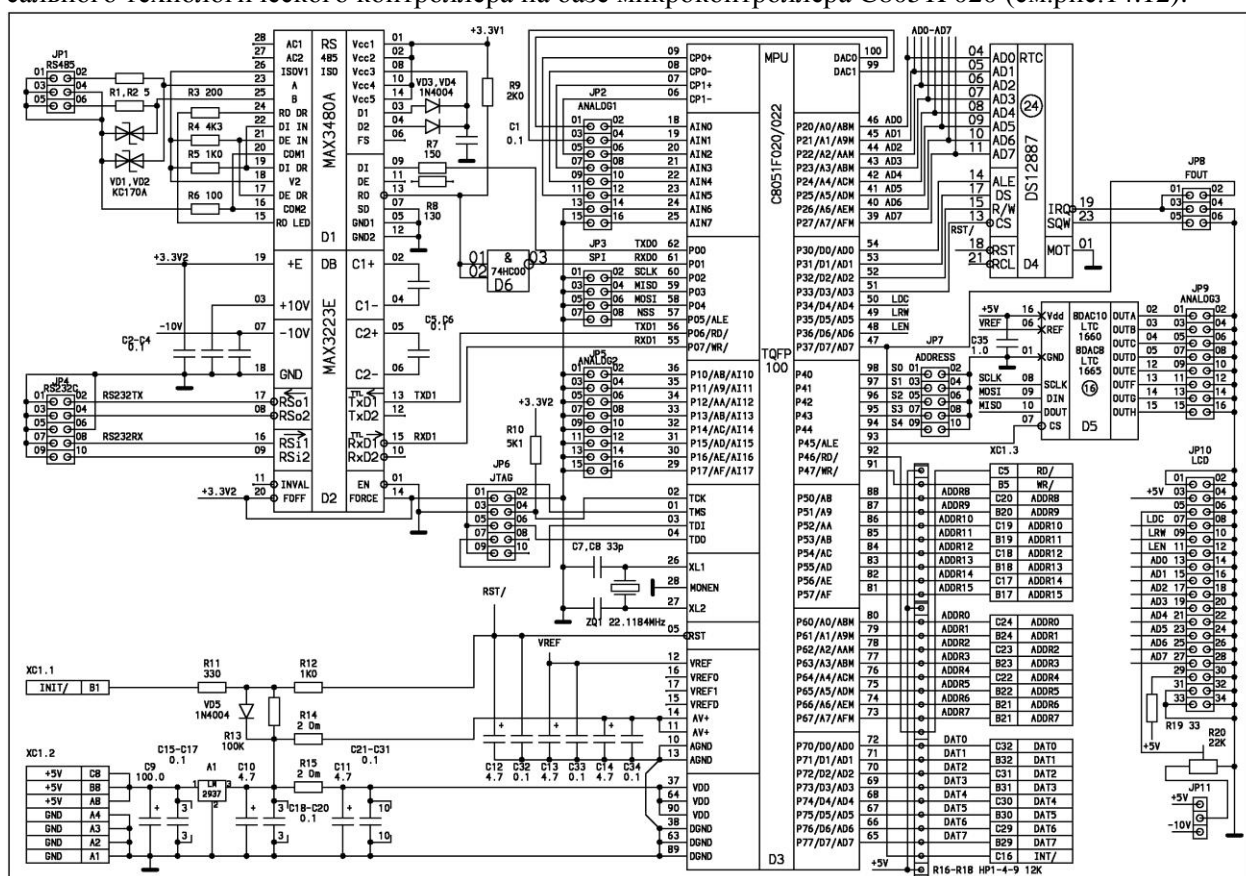


Рис.14.12. Универсальный технологический контроллер на базе микроконтроллера C8051F020

Описываемый контроллер выполнен в виде модуля магистрально-модульной системы с размерами 100x100 мм². Модуль оснащен системным разъемом СНР96 с модифицированной магистралью И41. Модуль представляет собой центральное контроллерное устройство. Подразумевается, что система содержит еще ряд модулей, на которых расположены следующие функциональные узлы:

- входные аналоговые усилители, необходимых для фильтрации и предварительного усиления аналоговых сигналов с датчиков перед подачей их на аналого-цифровой преобразователь;
- мощные выходные нормирующие усилители, необходимых для обеспечения требуемых уровней и мощности выходных сигналов цифро-аналоговых преобразователей;
- силовые и высоковольтные ключи с соответствующими регистрами;

- узлы оптической изоляции выходных сигналов.

Перейдем к рассмотрению собственно главного контроллера.

Используемый микроконтроллер фирмы Cygnal F8051F020 (D3) имеет 64K Flash памяти программ и 4.25K оперативной памяти. В состав цифровой периферии микроконтроллера входят аппаратный охранный таймер WDT, пять таймеров/счетчиков общего назначения, программируемый счетчик-массив, встроенный тактовый генератор до 16 MHz, подсистема сброса со встроенным монитором питания, развитый контроллер прерываний на 22 вектора, два универсальных высокоскоростных интерфейса UART и последовательный периферийный интерфейс SPI.

Приведенная схема мощного универсального контроллера содержит три подсистемы: подсистему аналогового ввода/вывода, подсистему интерфейсов и подсистему контроллера.

Подсистема интерфейсов содержит:

- драйвер интерфейса RS485 с оптической изоляцией и низковольтным питанием (3.3V) - D1 (MAX3480A), подключенный к первому последовательному интерфейсу UART0;
- драйвер интерфейса RS232C с низковольтным питанием D2 (MAX3223E), подключенный ко второму последовательному интерфейсу UART1;
- интерфейсный разъем JP10 для подключения "интеллектуального" алфавитно-цифрового жидкокристаллического индикатора LCD с интерфейсом L2462, например фирмы Seiko Instruments, подключенный к линиям портов P2 и P3.

Подсистема аналогового ввода/вывода содержит:

- Первый встроенный 12-разрядный аналого-цифровой преобразователь с входным мультиплексором. Мультиплексор может работать в двух режимах: однополярном восьмиканальном режиме или дифференциальным четырехканальным. Между выходом мультиплексора и входом аналого-цифрового преобразователя подключен масштабирующий усилитель с программируемым коэффициентом усиления, который может иметь значения 0.5, 1, 2, 4, 8 и 16. Входы мультиплексора первого аналого-цифрового преобразователя соединены с контактами штыревого двухрядного разъема JP2 (PLD16).
- Второй встроенный 8-разрядный быстродействующий аналого-цифровой преобразователь с входным мультиплексором. Мультиплексор может работать в двух режимах: однополярным восьмиканальным или дифференциальным четырехканальным. Между выходом мультиплексора и входом аналого-цифрового преобразователя подключен масштабирующий усилитель с программируемым коэффициентом усиления, который может иметь значения 0.5, 1, 2, 4. Входы мультиплексора второго аналого-цифрового преобразователя соединены с контактами штыревого двухрядного разъема JP5 (PLD16).
- Два встроенных 12-разрядных цифро-аналоговых преобразователя. Выходы обоих преобразователей соединены с контактами штыревого двухрядного разъема JP2 (PLD16).
- Два встроенных компаратора напряжения с возможностью независимой программной установки положительного и отрицательного порогов. Входы обоих компараторов также выведены на штыревой двухрядный разъем JP2 (PLD16).
- Встроенный источник опорного напряжения на 2.4V, от которого работает вся встроенная аналоговая периферия и внешний десятиразрядный восьмиканальный аналого-цифровой преобразователь D5 (LTC1660).
- Микросхема восьмиканального десятиразрядного цифро-аналогового преобразователя D5 (LTC1660). Эта микросхема использует интерфейс SPI микроконтроллера, который также выведен на штыревой двухрядный разъем JP3 (PLD8).
- В аналоговую подсистему также входит преобразователь питания 5V в 3.3V, выполненный на микросхеме LM2937-3.3 (A1), предназначенный для питания микроконтроллера D3 и микросхем с низковольтным питанием D1, D2.

Подсистема микроконтроллера включает в себя собственно микроконтроллер D3 и микросхему таймера реального времени D4 (DS12887). Микроконтроллер имеет восемь портов. На выходы первого порта P0 с помощью коммутатора ресурсов Crossbar выведены сигналы интерфейсов UART0 (P0.0, P0.1), SPI (P0.2-P0.5) и UART1 (P0.6, P0.7). На выводы второго порта P1 настроены

входы второго 8-разрядного аналого-цифрового преобразователя. Третий и четвертый порты (P2 и P3) используются для программного доступа к таймеру реального времени D4 и интерфейсу LCD. Пять младших разрядов пятого порта P4 соединены со штыревым разъемом PLD10, который используется для задания (путем установок перемычек) сетевого адреса станции (контроллера). Линия P4.5 используется для программной выборки внешней микросхемы цифро-аналогового преобразователя. Остальные порты P7-P5 и P4.6, P4.7 настроены на работу в режиме открытого истока (т.е. внутренние подтягивающие резисторы выключены) для обеспечения 5-вольтового режима работы и функционирования в качестве не мультиплексированного интерфейса внешних ресурсов (памяти и устройств ввода/вывода). Подтяжку уровней до 5V обеспечивают внешние резисторные матрицы R16-R18. Все линии интерфейса внешних ресурсов (т.е. магистрали) соединены с контактами системного разъема XC1. С этого же разъема поступает питания. Штырьковый двухрядный разъем JP6 (PLD10) предназначен для подключения JTAG адаптера-программатора (EC2). Штырьковый двухрядный разъем JP8 (PLD6) предназначен для коммутации выходных сигналов таймера реального времени. Весь контроллер выполнен всего на 6 микросхемах. Потребление от источника питания +5V составляет 295mA, из которых 280mA приходится на микросхему D1.

14.8. Новая микросхема - USB-UART мост

В августе 2003 года американская фирма Cygnal, известная в настоящее время как производитель самых высокопроизводительных x51 совместимых микроконтроллеров, опубликовала предварительную информацию на новый класс изделий - "интеллектуальный" USB-UART мост CP2101[42]. По всей видимости, микросхема построена на базе микроконтроллера семейства F32x с интерфейсом USB, о чем свидетельствует совпадение корпусов, выводов питания и сигналов USB. Скорее всего, микросхема представляет собой микроконтроллера с оригинальным Firmware.

Перечислим подробно основные анонсируемые характеристики изделия. Как уже было сказано выше, микросхема CP2101 производится в малогабаритном корпусе MLP-25, таком же, как микроконтроллер C8051F321. Напомним, что корпус имеет размеры всего 5x5 мм. Диапазон рабочих температур изделия - от -40°C до +85°C.

Микросхема CP2101 представляет собой преобразователь сигналов USB в сигналы UART, не требующая никаких внешних элементов, кроме фильтрующих конденсаторов. Микросхема содержит: встроенный тактовый генератор, не требующий внешних элементов (кварцевого или кристаллического резонатора); встроенную Flash память объемом 512 байтов для хранения: кода производителя (Vendor ID), кода продукта (Product ID), серийного номера, описания потребляемой мощности, версии изделия и строки описания изделия. Кроме этого, микросхема содержит встроенную подсистему сброса и регулятор напряжения с выходом 3.3 В.

Собственно сам интерфейс USB соответствует спецификации 2.0. Он обеспечивает передачу данных на скорости до 12 Мбит/с. Интерфейс содержит встроенную систему защиты и не требует внешних элементов.

Выходной последовательный интерфейс UART вырабатывает полный комплект сигналов интерфейса RS232C. Формат передачи данных - восемь бит + один стоп-бит. Интерфейс может работать с контролер на четность, на нечетность или без контроля. Интерфейс обеспечивает работу на скоростях от 300 бит/с до 921,6 Кбит/с. Имеются два внутренних буфера оперативной памяти по 512 байт каждый на прием и передачу соответственно. Обеспечивается также аппаратный контроль передачи.

По утверждению производителя, имеются поддерживающие драйверы для следующих операционных систем: Windows 98/Me/2000/XP, MAC OS-9, MAC OS-X, Windows CE и Linux версии 2.40 и выше.

Питание микросхемы может обеспечиваться от шины USB с напряжением от 4,0 В до 5,25 В или от батареи с напряжением от 3,0 В до 3,6 В.

Микросхема ориентирована на модернизацию существующих микроконтроллерных изделий, оснащенных интерфейсом UART с целью оснащения их интерфейсом USB. Кроме того, микросхема может найти широкое применение в различных измерительных и бытовых приборах, мобильных телефонах, интерфейсных кабелях PDA USB, конверторах USB-RS232.

Функциональная схема микросхемы CP2101 представлена на рис.14.13.

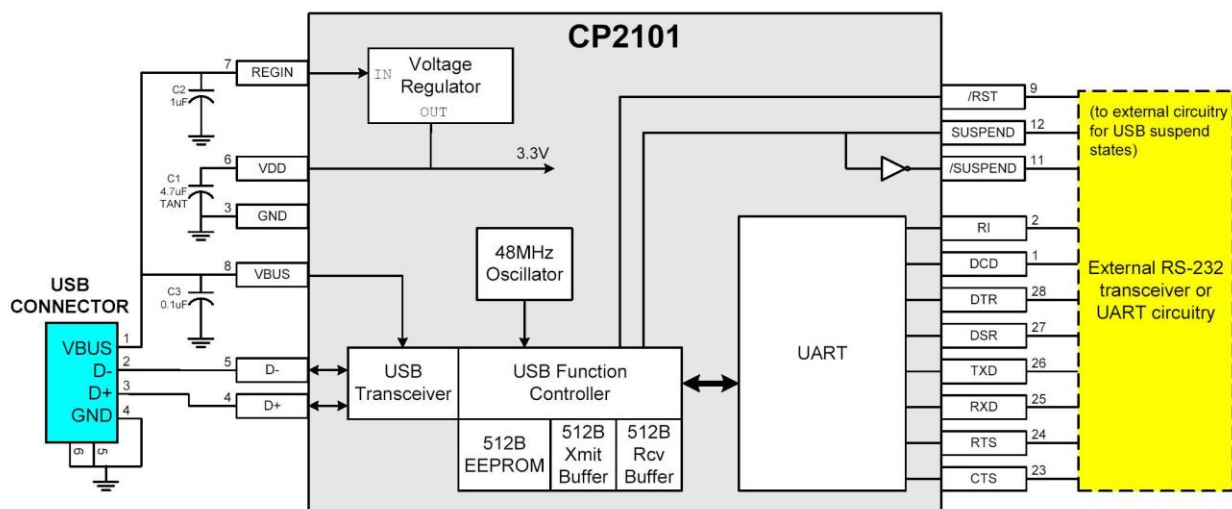


Рис.14.13. Функциональная схема микросхемы USB-UART моста CP2101.

Анализ приведенной функциональной схемы микросхемы и схемы ее включения показывает, что микросхема CP2101 позволяет создавать интерфейс USB в микроконтроллерных системах, имеющих интерфейс RS232 с минимумом элементов обвязки. Встроенная Flash память с идентификационной информацией может быть модифицирована внутрисхемно через интерфейс USB в фазе программирования. Фирма Cygnal предоставляет бесплатный виртуальный драйвер (Virtual COM Port device drivers – VCP) для использования интерфейса USB в качестве стандартного COM порта через микросхему CP2101, при этом последняя формирует все сигналы стандартного интерфейса RS232. При этом, никакой модификации устройств, сопрягаемых со стандартным COM портом не требуется. Это позволяет легко модифицировать ранее разработанные микроконтроллерные системы путем простой замены на печатной плате выходной интерфейсной микросхемы интерфейса RS232 (например, MAX232 или аналогичных микросхем) на микросхему CP2101. При этом имеется возможность существенного уменьшения размеров печатной платы за счет того, что микросхема CP2101 имеет значительно меньшие размеры и не требует дополнительных элементов обвязки, тогда как интерфейсные микросхемы RS232, как правило, используют до четырех дополнительных конденсаторов для формирования необходимых напряжений.

Кроме собственно микросхемы CP2101 фирма Cygnal предлагает эволюционный комплект CP2101EK, который содержит собственно эволюционную плату на базе микросхемы CP2101, полный комплект VCP драйверов для различных операционных систем, а также кабели USB и RS232 и полный комплект документации.

Описание выводов микросхемы CP2101 представлено в таблице 14.18.

Таблица 14.18

Название вывода	Номер вывода	Тип вывода	Описание
VDD	6	Вход питания	Вход питания при батарейном питании 3.0-3.6 В.
		Выход питания	Выход питания 3.3 В при использовании встроенного регулятора
GND	3		Общий
RST/	9	Цифровой вход/выход	Вход/выход сброса (Reset). Выход с открытым стоком встроенного монитора питания. Внешний источник сброса может осуществлять сброс микросхемы путем подачи на вход низкого уровня на время не менее 15 мкс.
REGIN	7	Вход питания	Вход встроенного 5-вольтового регулятора (стабилизатора) напряжения.
VBUS	8	Цифровой вход	Тестовый вход VBUS, соединенный с соответствующей линией интерфейса USB. Наличие на этом входе напряжения 5 В свидетельствует о том, что микросхема подключена к сети (интерфейсу) USB.
D+	4	Цифровой вход/выход	Сигнальный вход/выход USB D+ интерфейса USB.

D-	5	Цифровой вход/выход	Сигнальный вход/выход USB D- интерфейса USB.
TXD	26	Цифровой выход	Асинхронный выход данных (Transmit) интерфейса UART.
RXD	25	Цифровой вход	Асинхронный вход данных (Receive) интерфейса UART.
CTS	23	Цифровой вход	Вход ЗАПРОС НА ПЕРЕДАЧУ (активный - низкий уровень)
RTS	24	Цифровой выход	Выход ГОТОВНОСТЬ К ПЕРЕДАЧЕ (активный - низкий уровень)
DSR	27	Цифровой вход	Вход ГОТОВНОСТЬ АДАПТЕРА К РАБОТЕ (активный - низкий уровень)
DTR	28	Цифровой выход	Выход ГОТОВНОСТЬ ТЕРМИНАЛА К РАБОТЕ (активный - низкий уровень)
DCD	1	Цифровой вход	Вход ПЕРЕДАЧА ОБНАРУЖЕНА (активный - низкий уровень)
RI	2	Цифровой вход	Вход ВЫЗОВ (активный - низкий уровень)
SUSPEND	12	Цифровой выход	Выход устанавливается в высокое логическое состояние при нахождении интерфейса USB в состоянии ожидания.
SUSPEND/	11	Цифровой выход	Выход устанавливается в низкое логическое состояние при нахождении интерфейса USB в состоянии ожидания.
NC	10, 13-22	Не используются	Эти выводы должны быть либо свободны, либо соединены с выводом питания VDD.

Сигналы SUSPEND и SUSPEND/, индицирующие нахождение интерфейса USB в состоянии ожидания могут использоваться для управления электропитанием как самой микросхемы CP2101, так и возможной схемы обвязки. Эти сигналы устанавливаются в двух случаях: при нахождении интерфейса USB (сети USB) в состоянии ожидания, а также после сброса микросхемы CP2101 пока идет идентификация узлов USB, подключенных к ведущему (host) устройству. После завершения состояния ожидания сигналы SUSPEND и SUSPEND/ переходят в неактивное состояние. Следует также учитывать, что эти сигналы находятся в высокоимпедансном состоянии во время активного сигнала сброса. Если такая ситуация нежелательно, рекомендуется подтягивать сигнал SUSPEND/ через резистор 10K к общей шине питания

Интерфейс UART генерирует как основные сигналы: передачи (TX - transmit) и приема (RX - receive), так и все вспомогательные сигналы RTS, CTS, DSR, DTR, DCD и RI. Интерфейс может быть запрограммирован на различные режимы работы (контроля передачи с помощью вспомогательных сигналов), форматов передачи данных, а также контроля четности. Программирование осуществляется через оригинальные драйверы USB путем установки виртуального COM порта аналогично программированию штатных COM портов персонального компьютера.

Как уже упоминалось выше, в состав микросхемы входит Flash память объемом 512 байтов для хранения: кода производителя (Vendor ID), кода продукта (Product ID), серийного номера, описания потребляемой мощности, версии изделия и строки описания изделия, как того требует спецификация сети USB. Поскольку все эти данные модифицируемые, а настройка USB интерфейса host устройства (персонального компьютера) производится именно на основе этих данных, следует помнить, что пока данные не установлены пользователем, во Flash памяти содержатся заводские данные фирмы Cygnal (см. табл.2) и, соответственно, первоначальная настройка интерфейса USB будет производиться по ним. Кроме того, пользователь должен помнить, что уникальные серийные номера необходимы для правильной работы нескольких USB устройств, подключенных к персональному компьютеру.

Идентификационная Flash память программируется пользователем через интерфейс USB. Заводские данные записываются при производстве для проведения тестирования. Фирмой Cygnal предоставляется динамическая библиотека (DLL) и утилиты для самостоятельной модификации идентификационной памяти. Отметим также, что количество циклов перезаписи не менее 100 тысяч, а срок гарантированного хранения данных - 100 лет.

В таблице 14.19 показано содержание идентификационной памяти после заводского тестирования при поставке микросхем.

Таблица 14.19

Наименование поля	Название	Содержимое
Vendor ID	Код производителя	10C4h
Product ID	Код продукта	EA60h
Power Descriptor (Attributes)	Атрибуты питания	80h
Power Descriptor (Max. Power)	Потребление	32h
Release Number	Версия	0100h
Serial Number	Серийный номер	0001 (максимально 63 символа)
Product Description String	Описания изделия	"Cygnal CP2101 USB to UART Bridge Controller" (максимально 126 символов)

Как уже упоминалось выше, фирма Cygnal предоставляет бесплатный виртуальный драйвер (VCP) для использования интерфейса USB в качестве стандартного COM порта. При установке драйвера операционная система персонального компьютера обнаруживает дополнительный виртуальный COM порт, полностью идентичный аппаратным COM портам. После этого, приложение пользователя может работать с этим COM портом также, как и при работе со штатными аппаратными COM портами, но связь будет осуществляться через интерфейсы USB и микросхему CP2101.

В заключении коротко остановимся на электрических параметрах..

Питание микросхемы может осуществляться либо от соответствующей линии USB интерфейса через встроенный регулятор (стабилизатор) напряжения со входным рабочим диапазоном от 4,0 В до 5,25 В, либо от батарейного или иного источника питания с напряжением 3,3 В (от 3,0 В до 3,6 В). Встроенный регулятор напряжения имеет выход, совмещенный со входом батарейного питания, через который могут питаться и внешние цепи. Выходной ток встроенного регулятора от 1 мА до 100 мА.

В заключении отметим, что пока производится опытная продукция, производитель особо отмечает, что она не предназначена в изделиях, отвечающих за жизнеобеспечение систем или других критических приложениях. Производитель оставляет за собой право на дальнейшую модернизацию изделия с целью улучшения эксплуатационных характеристик.

14.9. Выводы

В заключение, перечислим еще раз общие достоинства микроконтроллеров фирмы Cygnal:

1. Все микроконтроллеры фирмы Cygnal имеют пиковую производительность, как минимум в 12 раз превосходящую производительность стандартных микроконтроллеров 8051, работающих на той же частоте.
2. Большинство микроконтроллеров работает на частотах до 25 МГц, при этом их пиковая производительность достигает 25 MIPS.
3. Одно из восьми выпускаемых семейств F12x может работать на частотах в 50 и 100 МГц, при этом их пиковая производительность достигает 50 и 100 MIPS соответственно.
4. Все микроконтроллеры выпускаются с Flash памятью большого объема:
 - 16К (8К, 4К, 2К) для сверхмалых микроконтроллеров семейств F30х-33х;
 - 32К, 16К, 8К для средних микроконтроллеров семейств F0хх, F018-9, F2хх;
 - 64К и 128К для мощных микроконтроллеров семейств F02х, F04х, F06х и F12х.
5. Важной особенностью Flash памяти микроконтроллеров Cygnal является возможность чтения, записи или стирания Flash памяти непосредственно из исполняемой программы. Это позволяет

- на только создавать самозагружаемые системы, но и хранить в Flash памяти редко изменяющиеся данные и настройки.
6. Все микроконтроллеры имеют основную оперативную память размером 256 байт, а многие имеют еще и дополнительную оперативную память, расположенную в адресном пространстве внешней памяти объемом от 1К до 8К.
 7. Микропроцессоры семейств большой мощности F02x, F04x, F06x и F12x оснащены интерфейсом внешней памяти, способным работать в мультиплексированном и немultipлексированном режимах.
 8. Практически все базовые модели микроконтроллеров средней мощности имеют 4 однобайтных порта ввода/вывода. Усеченные модели этих семейств имеют меньшее количество портов. Базовые микроконтроллеры большой мощности имеют 8 портов, а усеченные модели этих семейств – 4 порта. Сверхмалые микроконтроллеры семейства F30x-F33x имеют 3-1 порт.
 9. Все микроконтроллеры имеют 3 и более таймеров общего назначения.
 10. Почти все микроконтроллеры имеют аппаратный охранный таймер WDT.
 11. Многие микроконтроллеры оснащены встроенным монитором питания и развитой системой двунаправленного сброса.
 12. Все микроконтроллеры имеют усовершенствованный обработчик прерываний, обрабатывающий в среднем 20 векторов прерываний.
 13. Все микроконтроллеры имеют встроенный системный генератор и расширенный внешний генератор, способный работать с внешним кварцевым или пьезокерамическим резонатором, RC-сепочкой и отдельным конденсатором. Некоторые семейства имеют прецизионный калибруемый встроенный генератор с поддержкой работы последовательного порта.
 14. Практически все семейства имеют в составе цифровой периферии программируемый счетчик-массив, который может работать, как таймер общего назначения, 8- и 16-битный широтно-импульсный модулятор и высокоскоростной генератор импульсов. В некоторых моделях на одном из таймеров общего назначения и на одном из модулей программируемого массива счетчика возможна организация таймера реального времени.
 15. Практически все семейства имеют интерфейсы SMBus, совместимый с I2C, и SPI.
 16. В семействе F04x имеется интерфейс CAN 2.0B.
 17. В семействе F32x имеется интерфейс USB.
 18. Все семейства имеют один, а некоторые - два высокоскоростных последовательных интерфейса UART. Следует отметить, что все они поддерживают безошибочную скорость передачи данных 115200 при большом наборе стандартных кварцевых резонаторов.
 19. Все перечисленные интерфейсы поддерживают многопроцессорный режим обмена.
 20. Все семейства микроконтроллеров имеют различные варианты коммутаторов ресурсов Crossbar, которые позволяют оптимально ассоциировать входы/выходы используемой периферии и занимать ими физические выводы портов начиная с младшего P0 до P2. В некоторых семействах Crossbar коммутирует также и аналоговые входы.
 21. Все микроконтроллеры имеют встроенный интерфейс внутрисистемного программирования, с возможностью отладки JTAG или C2.
 22. Все микроконтроллеры имеют развитые средства защиты Flash памяти от чтения и записи.
 23. Большинство микроконтроллеров имеет встроенный источник опорного напряжения или могут использовать напряжение питания для этих целей.
 24. Практически все семейства содержат два аналоговых компаратора с возможностью программирования величины положительного и отрицательного гистерезиса и времени срабатывания. Как правило, один из компараторов позволяет вызывать сброс микроконтроллера. В некоторых семействах количество компараторов изменено на 1 или 3.
 25. Многие микроконтроллеры имеют на кристалле два 12-битных цифро-аналоговых преобразователя с выходом по напряжению, функцией сдвига входного кода и возможностью обновления состояния через заданные интервалы времени.
 26. Многие микроконтроллеры имеют среднескоростной 12- (10-) битный аналого-цифровой преобразователь с предварительным программируемым усилителем (коэффициенты усиления - 0.5, 1, 2, 4, 8 и 16) и входным мультиплексором с различной архитектурой и количеством входов от 8 до 32. Мультиплексор обеспечивает как однополярный режим, так и дифференциальный при уменьшении количества входов мультиплексора в 2 раза.

27. Некоторые микроконтроллеры имеют также второй высокоскоростной 8-битный аналого-цифровой преобразователь с предварительным программируемым усилителем (коэффициенты усиления - 0.5, 1, 2 и 4) и входным мультиплексором с различной архитектурой и количеством входов до 8. Мультиплексор обеспечивает как однополярный режим, так и дифференциальный при уменьшении количества входов мультиплексора в 2 раза.
28. Практически все микроконтроллеры имеют встроенный датчик температуры.
29. Все микроконтроллеры работают при напряжениях питания от 2.7 до 3.6V, при этом, содержащееся во внутренней оперативной памяти сохраняется при снижении напряжения до 1.5V.
30. Потребляемый ток практически всех семейств составляет 10-12mA в рабочем режиме, и не более 2 mA в режиме Idle. Исключением является сверхминиатюрное семейство C8051F30x, которое потребляет ток менее 5.8mA, и сверхбыстродействующее семейство C8051F12x, которое на частоте 100MHz потребляет не более 50mA.
31. Все микроконтроллеры работают в промышленном диапазоне температур: от -40°C до +85°C.

Перечисленные достоинства более чем достаточны, для того, чтобы заявить, что микроконтроллеры фирмы Cygnal на сегодняшний день не имеют достойных конкурентов.

Однако анализ технической и справочной документации, а также опыт разработки и эксплуатации ряда макетов и изделий позволяет сформировать несколько замечаний и пожеланий:

1. Документация на отдельные семейства микроконтроллеров написана не достаточно четко. Многие из вопросов настройки новых узлов (PLL, СНН) описаны недостаточно подробно и последовательно.
2. В документации имеется некоторое количество незначительных ошибок.
3. Похвальное стремление разработчиков фирмы Cygnal к совершенствованию коммутатора ресурсов Crossbar и структуры регистров специального назначения SFR привело к большому разнообразию реализаций коммутаторов ресурсов Crossbar, а также к тому, что многие, вновь введенные регистры SFR или отдельные биты в различных моделях называются по-разному и располагаются по разным адресам. Очевидно, что это отрицательно сказывается на сроках освоения изделий и снижает программную совместимость семейств.
4. Использование в продукции фирмы Cygnal корпусов TQFP с шагом выводов 0.5mm и шириной вывода в 0.3mm часто затрудняет создание новых разработок на этих микроконтроллерах в связи с отсутствием необходимой технологической базы производства печатных плат и опыта проведения точных монтажных работ.
5. Поскольку наиболее широкое распространение в ближайшее время, по мнению автора, получат семейства C8051F0xx, C8051F02x, C8051F2xx и C8051F33x, хотелось бы, чтобы была произведена модернизация или дальнейшее развитие этих семейств. При этом основными задачами модернизации или развития должны быть:
 - повышение производительности за счет повышения рабочей тактовой частоты;
 - оснащение семейств вторым последовательным портом и, соответственно, дополнительным таймером;
 - оснащение их прецизионным внутренним генератором с поддержкой UART и более высокой тактовой частотой.
 - оснащение их аппаратным таймером реального времени.