«Самодельный» контроллер ЖКИ на микроконтроллере

Применение микроконтроллеров в различных измерительных устройствах и вывод значений измеряемых величин на цифровой индикатор стали уже делом привычным. Чаще всего цифровые индикаторы выполняют в виде строки из нескольких семисегментных знакомест, разделенных десятичной точкой. Наибольшее распространение по причинам общеизвестным получили светодиодные и жидкокристаллические индикаторы (ЖКИ). При проектировании портативной аппаратуры с питанием от гальванических элементов предпочтение отдают, как правило, последним ввиду их очень малого энергопотребления, хотя с их применением и связан ряд сложностей и ограничений. Одна из таких сложностей состоит в том, что для управления каждым сегментом знакоместа необходимо переменное напряжение. То есть, если для свечения светодиодного сегмента требуется небольшой постоянный ток, то для отображения жидкокристаллического сегмента на индикаторе на него необходимо подать переменное напряжение с частотой в пределах, как правило, от *30* до *60 Гц*. Таким образом изображение какого-либо числа на семисегментном ЖКИ требует своего рода «регенерации». Чтобы освободить программно-аппаратные ресурсы микроконтроллера от необходимости поддерживать регенерацию изображения на таком индикаторе применяют либо специально разработанные для этой цели микросхемы контроллеров (см. **таблицу 1**), либо индикаторы с уже встроенным контроллером. По видимому та же самая проблема привела такие фирмы как *Atmel* и *Microchip* к разработке и серийному выпуску особого класса микроконтроллеров со встроенной схемой управления ЖКИ (см. **таблицу 2**).

Таблица 1. *Специализированные ИС управления ЖКИ*

Наименование	К-во сегм.	Кол-во общих выв.	Интерфейс	Корпус
HOLTEK:				
HT1621	32	4	Serial	
NXP Semiconductors (Philips Semiconductors):				
OM4068	32	3	SPI	DIP-40,QFP-44
PCF2100C	20	2	SPI	DIP-28,SOIC-28
PCF2111C	32	2	SPI	DIP-40,VSO-40
PCF2112C	32	1	SPI	DIP-40,VSO-40
PCF8533	80	4	I^2C	Chip On Glass
PCF8562	32	4	I^2C	TSSOP-48
PCF8566	24	4	I^2C	DIP-40,VSO-40
PCF8576	40	4	I^2C	LQFP-64,VSO-56
PCF8577	32	2	I^2C	DIP-40,VSO-40
OKI Semiconductor:				
ML9060	160	2	SPI	Chip On Glass
MSM6779B	160	-	4-bit parallel	Chip On Glass
MSM9006-01	41	3	SPI	QFP-64
MSM9006-02	41	4	SPI	QFP-64
ROHM:				
BU9716BK	32	3	SPI	QFP-44
BU9716BKV	32	3	SPI	VQFP-48C
BU9718KV	32	3	SPI	VQFP-48C
BU9728AKV	32	4	SPI	VQFP-48C
BU9735K	18	4	SPI	QFP-32
MiniLogic:				
ML1001	40	2	SPI	Chip On Glass
ML2002	48	2	SPI	Chip On Glass

И тем не менее как радиолюбители, так и разработчики промышленной аппаратуры еще сталкиваются с трудностями применения в своих разработках ЖКИ, так как универсального решения не существует. Если, например, требуется управлять 6-разрядным семисегментным ЖКИ с только одним общим электродом, то даже без десятичной точки количество требующих управления сегментов составит 6 x 7 = 42. Очевидно, что из приведенных в **таблице 2** с этой задачей справится только два микроконтроллера, а из представленных в **таблице 1** специализированных микросхем – только пять, да и те недешевы, малодоступны и не всегда удобны в использовании. Номенклатура цифровых семисегментных ЖКИ со встроенной схемой управления крайне

ограничена, в то время как целый ряд фирм, таких как, например, минское *НПО «Интеграл»* или китайская *Intech LCD Group Ltd.* выпускают много типов удобных даже в макетировании выводных индикаторов.

Таблица 2. *Микроконтроллеры со схемой управления ЖКИ*

Наименование	К-во сегм.	Кол-во общих выводов	Корпус
Atmel:			
ATmega169	25	4	TQFP-64
ATmega329/649	25	4	TQFP-64
ATmega3290/6490	40	4	TQFP-100
Microchip:			
PIC18F8390/8490	48	4	TQFP-80
PIC18F6390/6490	32	4	TQFP-64
PIC16F946	42	4	TQFP-64
PIC16F914/917	24	4	DIP-40, TQFP-44
PIC16F913/916	16	4	DIP-28, SOIC-28, SSOP-28, QFP-28

Схему управления семисегментным ЖКИ можно сделать самостоятельно на основе недорогого современного микроконтроллера. Преимущества такого решения очевидны:

1) такой контроллер легко сконфигурировать под конкретный индикатор;
2) интерфейс загрузки данных легко адаптируется под ту схему, в составе которой призван работать контроллер;
3) дополнительные возможности, такие как, например, встроенная дешифрация из двоичного, двоично-десятичного или *ASCII*-кода в семисегментный, и пр.

Вариант построения такого «самодельного» контроллера ЖКИ приведен на **рис.1**.

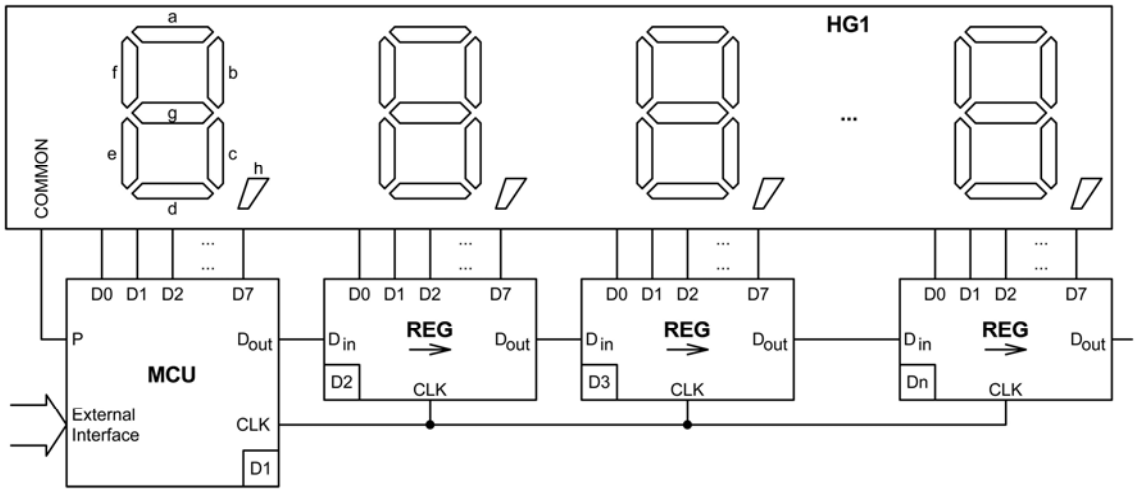

Рис.1. *Вариант схемы управления ЖКИ на основе микроконтроллера*

Микроконтроллер *D1*, работающий под управлением записанной в его память программы, обеспечивает как загрузку новых предназначенных для вывода на индикатор *HG1* данных, так и «регенерацию» изображения на индикаторе, управляя его общим электродом *COMMON* и выводами сегментов. Сдвиговые регистры *D2..Dn* позволяют наращивать количество управляемых контроллером знакомест и дополнительных вспомогательных сегментов.

На **рис.2** приведена принципиальная схема построенного подобным образом модуля индикации на основе микроконтроллера *ATtiny2313* фирмы *Atmel* и недорогого ЖКИ типа *ИЖЦ5-4/8*.

Данный модуль индикации функционирует под управлением микроконтроллера *D1*. Поэтому те или иные функциональные особенности модуля, необходимые для работы в составе того или иного устройства, реализуются путем соответствующего построения программного обеспечения микроконтроллера. В частности, если сама схема предусматривает только последовательный интерфейс загрузки предназначенных для вывода на индикатор данных, то способ загрузки, соответствующий тому или иному типу интерфейса, определяется алгоритмом, по которому микроконтроллер работает с интерфейсными входами модуля. Таким образом могут быть реализованы такие распространенные 2-проводные последовательные интерфейсы обмена данными, как *SPI* и I^2C. Интерфейсы этих типов являются синхронными, то есть последовательно передаваемые по линии *DATA/SDA* биты данных стробируются импульсами, передаваемыми по линии *SCK/SCL* (для *SPI* и I^2C соответственно). Используя же вход *RxD* (*XT2* на схеме) встроенного в микроконтроллер асинхронного приемо-передатчика можно реализовать совместимый с *RS-232C* асинхронный последовательный интерфейс. Для загрузки данных в этом случае требуется всего одна сигнальная линия! Скорость обмена данными может быть

выбрана из стандартного ряда и устанавливается на этапе разработки программного обеспечения микроконтроллера.

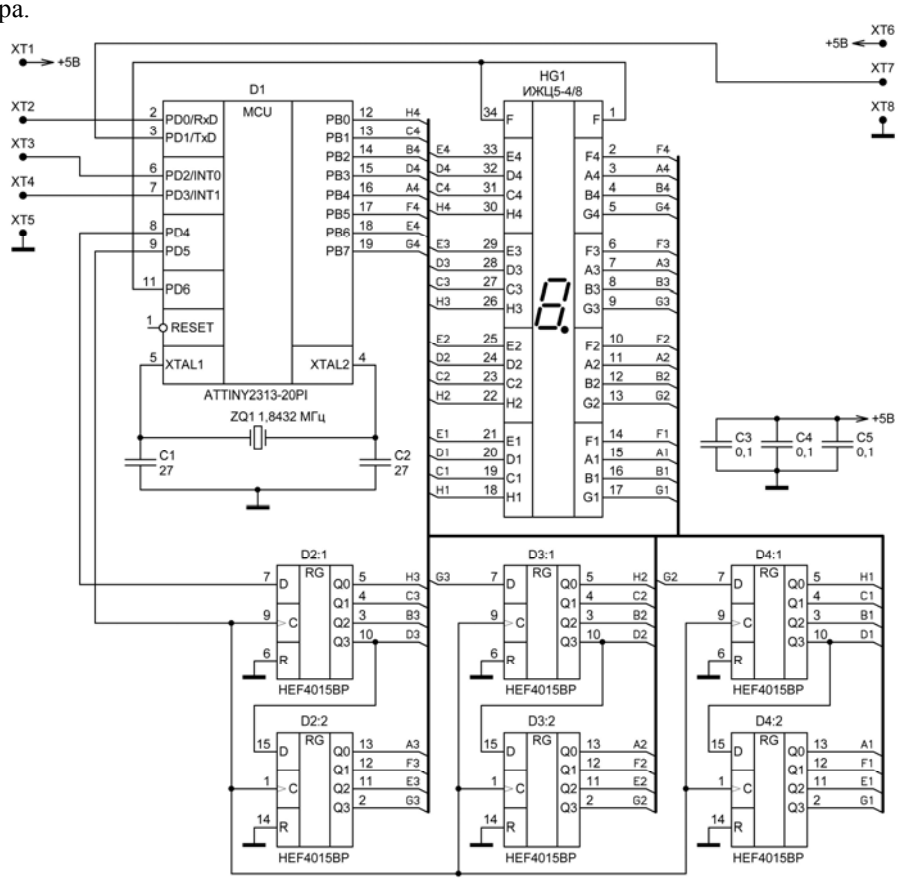

Рис.2. *Принципиальная схема модуля индикации с ЖКИ типа ИЖЦ5-4/8*

Плата модуля индикации – двухсторонняя. Трассировка печатных проводников и установка элементов по обе стороны платы изображены на **рис.3**. Модуль выполнен на печатной плате размером *66* х *35,6 мм*, что не намного больше размеров самого индикатора (см. фото на **рис.4**). Достигается это тем, что индикатор *ИЖЦ5-4/8* устанавливается на плату с одной стороны, а все остальные элементы – с другой.

ВНИМАНИЕ! Индикатор *HG1* устанавливается в последнюю очередь!

В качестве примера опишем алгоритмы разработанного автором программного обеспечения микроконтроллера *D1*, которое позволяет самостоятельно изготовить модуль индикации с интерфейсом типа *SPI*. В отличие от похожих контроллеров ЖКИ типа *OM4068* и *PCF2112C* фирмы *Philips Semiconductors* (теперь *NXP Semiconductors*) управление модулем намного проще и осуществляется только двумя сигналами. Особенностью разработки стала возможность последовательного каскадного подключения нескольких модулей индикации к одному интерфейсу. Функциональное соответствие выводов модуля и указанного интерфейса приведено в **таблице 3**, а способы подключения – на **рис.5** и **рис.6**.

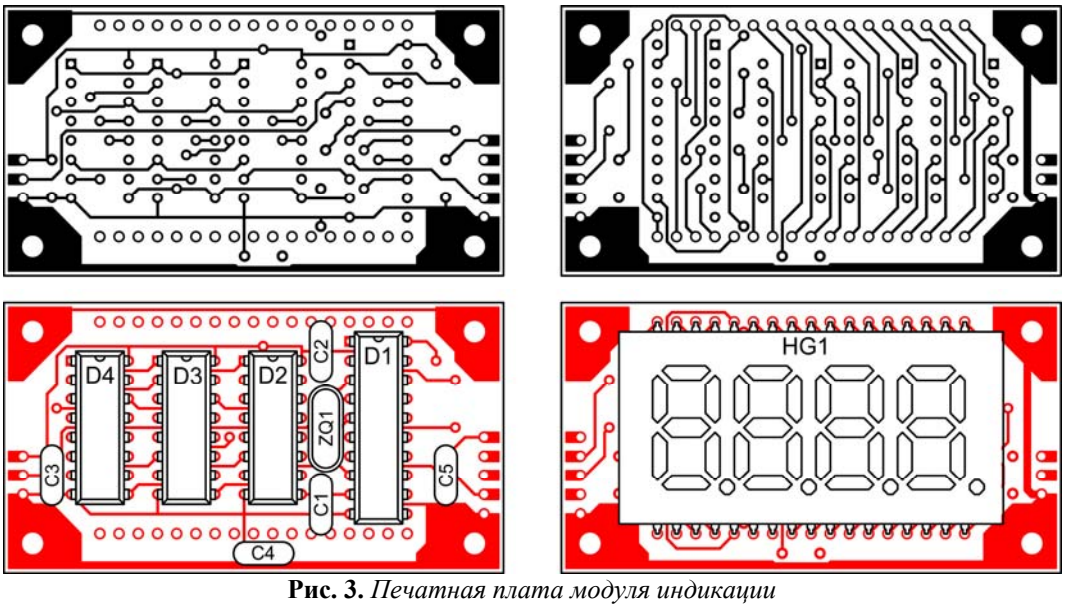

Рис. 3. *Печатная плата модуля индикации*

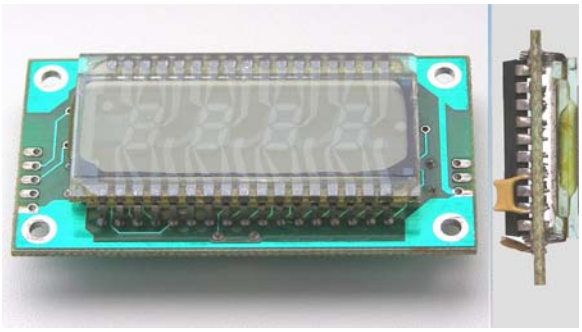

Рис.4. *Внешний вид модуля индикации*

Таблица 3. *Соответствие сигналов интерфейса SPI выводам модуля индикации*

Интерфейсный сигнал	Вывод модуля	Тип	Функция
DATA IN	XT4	вход	Подаваемая на модуль последовательность бит данных
SCK	XT3	вход	Подаваемые на модуль синхроимпульсы стробирования бит данных
DATA OUT	XT7	выход	Выдаваемая на следующий модуль последовательность бит ранее принятых данных (используется при последовательном каскадном подключении нескольких модулей индикации)

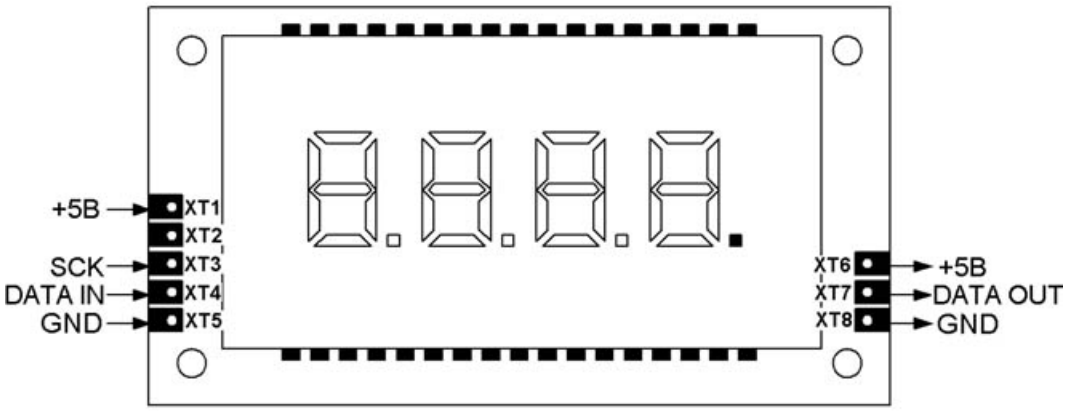

Рис.5. *Подключение модуля индикации к интерфейсу типа SPI*

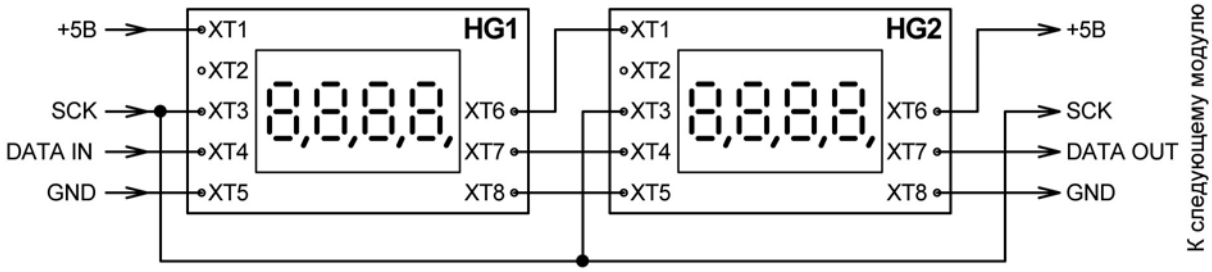

Рис.6. *Подключение нескольких модулей к одному интерфейсу типа SPI*

Управление модулем индикации осуществляется по двум линиям: *DATA IN* и *SCK*. Биты данных, последовательно поступающие на вход *DATA IN*, стробируются импульсами, поступающими на вход *SCK* (см. диаграмму на **рис.7**). По перепаду сигнала на входе *SCK* из состояния лог. «1» в состояние лог. «0» происходит ввод значения очередного бита данных с линии *DATA IN*. Если после ввода бита данных в течение 20 мс не произойдет очередного перепада сигнала *SCK* из состояния лог. «1» в состояние лог. «0» (не начнется ввод следующего бита), то есть если $t_{PERIOD} > 20$ *мс*, то принятые данные будут выведены на индикатор. Таким образом интервал $t_{PERIOD} > 20$ *мс* является признаком окончания процесса ввода данных и командой на обновление изображения на индикаторе в соответствии с принятыми данными. Другие временные соотношения сигналов управления модулем индикации приведены в **таблице 4**.

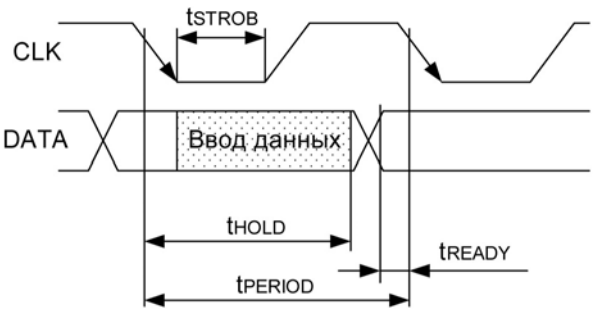

Рис.7. *Загрузка бит данных*

Таблица 4. *Временные соотношения сигналов управления*

Параметр	Значение, мкс	
	min	max
t_{STROB}	4	t_{PERIOD}-4
t_{PERIOD}	40	20000
t_{HOLD}	12	-
t_{READY}	1	-

Информация загружается в модуль блоками по 32 бита, каждый из которых соответствует определенному сегменту индикатора, одному из 32-х. Обозначение сегментов и разрядов индикатора приведено на **рис.8**, а соответствие между порядковым номером бита в блоке и сегментами индикатора – в **таблице 5**. Сегмент отображается на индикаторе если соответствующий бит данных установлен в состояние лог. «1».

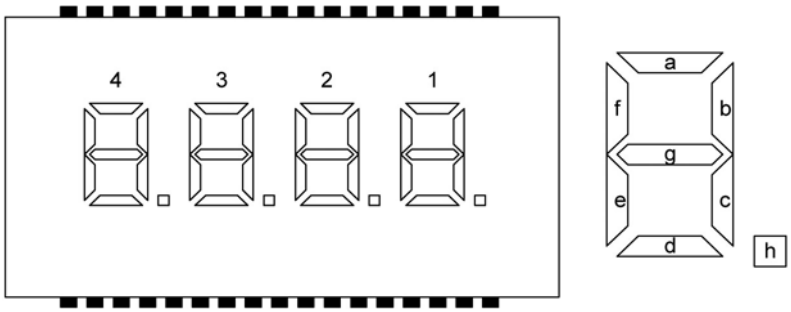

Рис.8. *Разряды и сегменты индикатора*

Таблица 5. *Соответствие очередности бит данных информационного блока сегментам индикатора*

Бит блока, № п.п.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Сегмент индикатора	4a	4b	4c	4d	4e	4f	4g	4h	3a	3b	3c	3d	3e	3f	3g	3h
Бит блока, № п.п.	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
Сегмент индикатора	2a	2b	2c	2d	2e	2f	2g	2h	1a	1b	1c	1d	1e	1f	1g	1h

Диаграмма на **рис.9** иллюстрирует порядок ввода данных при каскадном последовательном соединении нескольких модулей индикации. Предназначенные каждый для своего модуля индикации 32-битные блоки данных следуют один за другим. При этом биты предыдущего блока в том же порядке с выхода *DATA OUT (XT7)* подаются на вход *DATA IN* следующего модуля и стробируются импульсами *SCK*. Поэтому первым передается информационный блок, предназначенный для последнего модуля в цепочке, а последним – для первого.

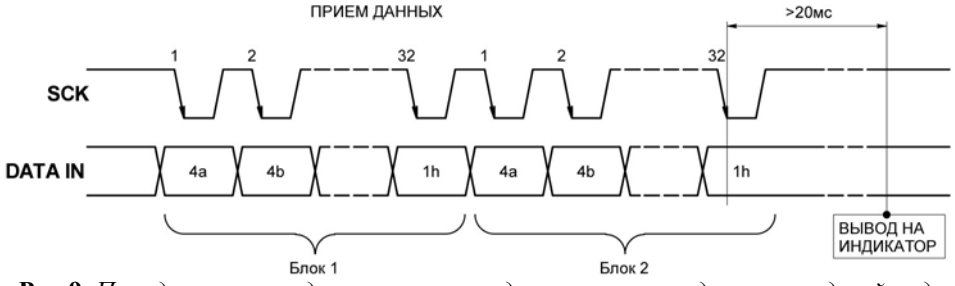

Рис.9. *Порядок загрузки данных при последовательном соединении модулей индикации*

Ввод очередного бита данных, поступившего на вход *DATA IN* модуля индикации, реализован в подпрограмме обработки прерываний по перепаду состояния входа *PD2/INT0* микроконтроллера *D1* из лог. «1» в лог. «0». Алгоритм подпрограммы приведен на **рис.10**, а ее исходный текст – в файле *STROBINT.ASM* приложения. В первую очередь фиксируем значение очередного стробируемого бита данных – состояние вывода *PD3/INT1 (DATA IN)* на момент прерывания. Это значение размещаем в бите переноса *C* статус-регистра *SREG* для последующего сдвига вправо через перенос организованного в оперативной памяти 4-

байтового «сдвигового регистра» принимаемых данных. После установки в лог. «1» признака активности процесса загрузки данных и перезапуска отсчета тайм-аута вывод микроконтроллера *PD1/TxD (DATA OUT)* устанавливаем в соответствии со значением младшего, то есть наиболее раннего принятого бита из 4-байтового «сдвигового регистра» принимаемых данных.

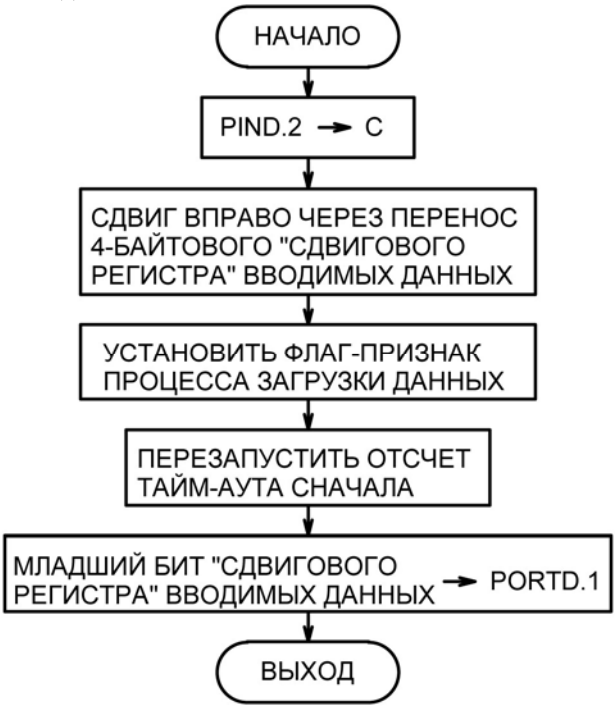

Рис.10. *Алгоритм ввода очередного бита данных*

Теперь рассмотрим порядок «регенерации» изображения на жидкокристаллическом индикаторе *HG1*. На его общий электрод (выводы 1 и 34, см. схему на **рис.2**) с вывода *PD6* микроконтроллера *D1* подается последовательность импульсов с частотой следования около *50 Гц* и скважностью около 2. Чтобы тот или иной сегмент отображался на индикаторе, текущий уровень сигнала на соответствующем ему выводе поддерживается противоположным состоянию сигнала на общем электроде индикатора, то есть в противофазе. Таким образом на сегменте появляется переменное напряжение, формируемое как разность потенциалов между его выводом и общим электродом индикатора. Амплитуда этого напряжения практически равна значению напряжения питания модуля индикации. На вывод неотображаемого сегмента сигнал подается синфазно с сигналом общего электрода, поэтому напряжение на сегменте практически отсутствует. С каждой сменой состояния сигнала на общем электроде соответственно обновляются сигналы на выводах сегментов. Сигналы на выводы сегментов старшего разряда индикатора подаются с выводов *PB0..PB7* микроконтроллера *D1*, а всех остальных разрядов – с выходов сдвиговых регистров *D2..D4*. Загрузка соответствующих данных в эти регистры производится сигналами с выводов *PD4* (данные) и *PD5* (строб). Поскольку загрузка регистров сдвига *D2..D4* производится очень быстро, то этот процесс занимает незначительный интервал в периоде «регенерации» (см. диаграмму на **рис.11**). На время действия этого интервала вывод *PD6* микроконтроллера *D1* переводится в высокоимпедансное состояние, отключая паразитную емкостную нагрузку выходов регистров и предотвращая нежелательное затягивание фронтов цифровых сигналов, а также некоторый рост энергопотребления.

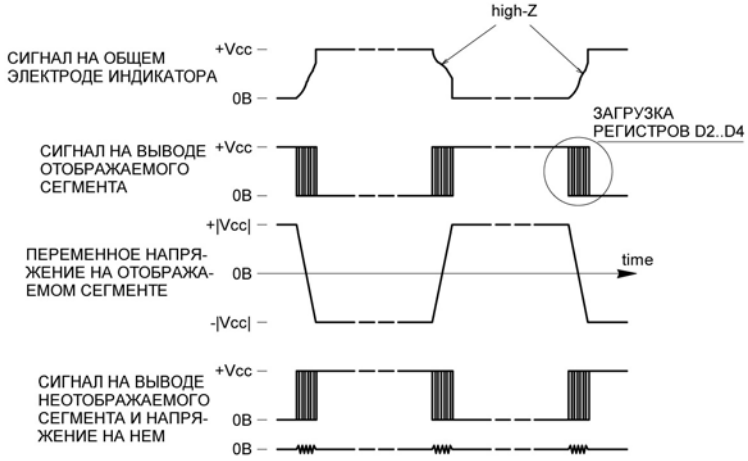

Рис.11 *Сигналы на выводах ЖКИ HG1*

Период регенерации задают прерывания по переполнению встроенного в микроконтроллер таймера-счетчика *T/C0*. Алгоритм подпрограммы обработки этих прерываний приведен на **рис.12**, а ее исходный текст – в файле *TOVFLINT.ASM* приложения. Сначала записываем в счетный регистр *TCNT0* таймера-счетчика *T/C0* константу, со значения которой таймер-счетчик продолжит свой счет. Этим обеспечивается необходимое значение периода прерываний (около *9,9 мс*). Затем, если идет процесс загрузки новых данных, следует анализ условия ее окончания с соответствующими установкой/сбросом флагов-признаков. Перед выходом из подпрограммы устанавливается флаг запроса на очередное обновление сигналов на общем электроде ЖКИ и на выводах его сегментов.

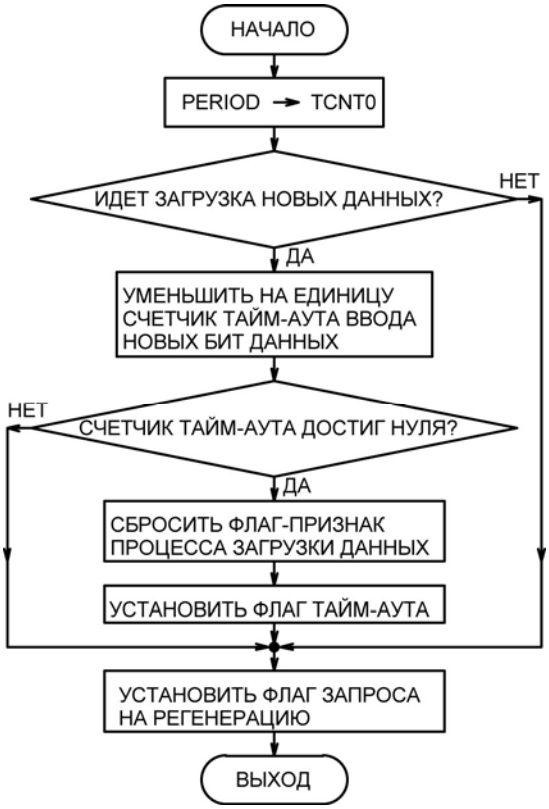

Рис.12. *Алгоритм обработки прерываний от T/C0*

Сохранение принятых данных, их дешифрация, а также «регенерация» изображения на ЖКИ реализованы в основном модуле программы, алгоритм работы которого представлен на **рис.13**. Работа основного модуля программы начинается по окончании действия внутреннего сигнала стартового сброса микроконтроллера. Первым делом производится стартовая инициализация, а именно: настройка режимов работы встроенных в микроконтроллер периферийных устройств (аналогового компаратора, таймера-счетчика, портов ввода-вывода и пр.), установка начальных значений регистров оперативной памяти, режимов прерываний и др. Перед началом главного цикла основного модуля программы следует разрешение прерываний. Далее, с целью снижения общего энергопотребления модуля индикации, программа переводит микроконтроллер в режим «холостого хода» (т.н. «idle-mode») и дальнейшая работа инициируется прерываниями, алгоритм обработки которых описан выше. Если прерывание было от импульса стробирования загружаемых в модуль индикации данных (по перепаду состояния входа *PD2/INT0* микроконтроллера *D1*), то никаких дальнейших действий от основного модуля программы не требуется и микроконтроллер снова переводится в режим «холостого хода». Если же в результате обработки очередного прерывания по переполнению таймера-счетчика *T/C0* был установлен флаг запроса на «регенерацию», то основной модуль программы последовательно осуществляет следующие действия:

- сбрасывает флаг запроса на «регенерацию», чтобы только следующее прерывание по переполнению таймера-счетчика *T/C0* вызвало следующие ниже действия;
- проверяет состояние флага тайм-аута ввода данных и, если он установлен (завершен прием новых данных для отображения на ЖКИ), то запрещает прерывания, копирует принятые данные в расположенный в другом месте оперативной памяти буфер индикации, сбрасывает флаг тайм-аута (во избежание повторения перечисляемых действий), разрешает прерывания и производит перекодировку принятых данных для их правильного отображения на индикаторе;
- производит «регенерацию» – обновляет текущее состояние сегментных выводов и общего электрода индикатора путем загрузки инверсных по отношению к предыдущим значений в регистры сдвига *D2..D4*, порт *PB0..PB7* микроконтроллера *D1* и его выход *PD6*;
- снова переводит микроконтроллер в режим «холостого хода» (главный цикл замыкается).

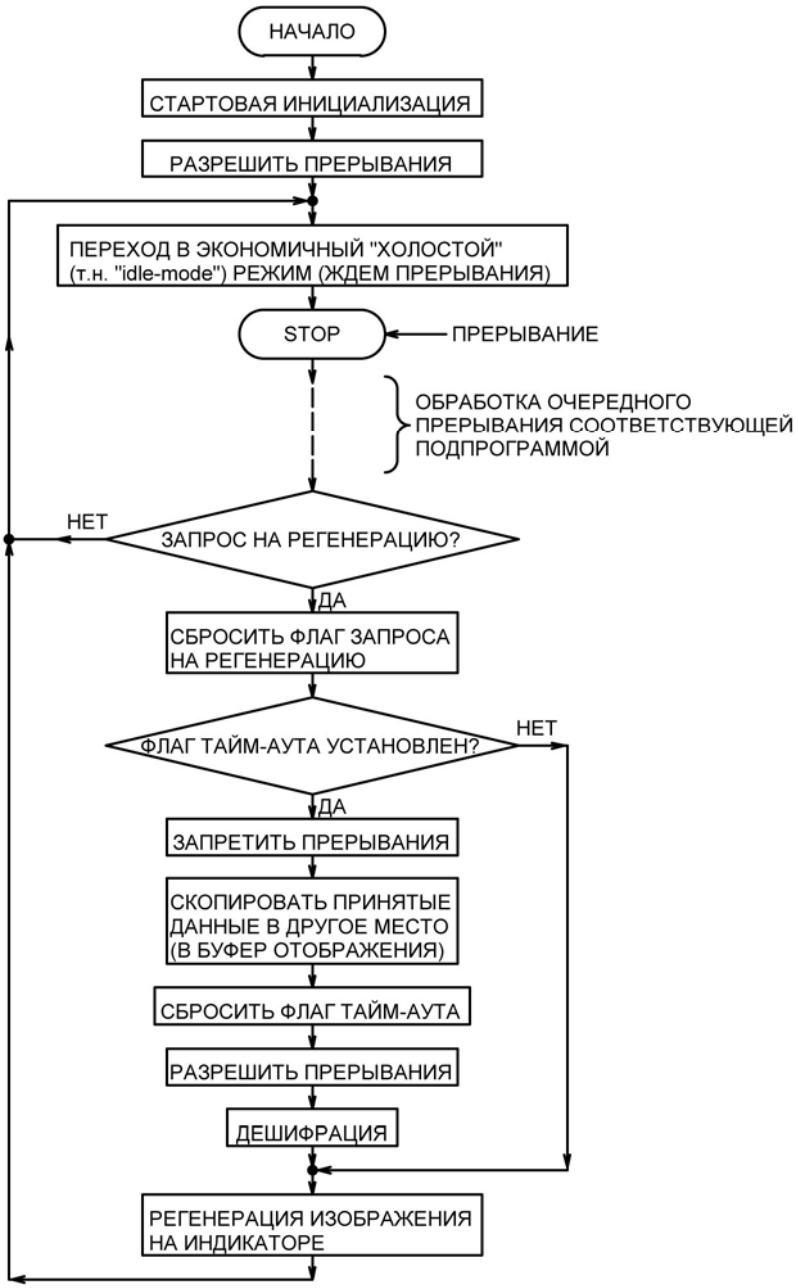

Рис.13. *Алгоритм работы основного модуля программы*

Необходимость дешифрации обусловлена несоответствием бит загружаемого в модуль индикации информационного блока порядку бит данных в сдвиговых регистрах *D2..D4*. Связано это с тем, что выходы этих регистров подсоединены к выводам индикатора *HG1* исходя из простоты трассировки печатных проводников на плате модуля.

Исходный текст основного модуля программы содержится в файле *LCDCTRL.ASM* приложения. Подробные комментарии, которыми снабжен текст, при сопоставлении с описанными алгоритмами позволяют легко разобраться в программе чтобы затем, если потребуется, уже самостоятельно модифицировать ее под свои нужды. Код для записи во *Flash*-память программ микроконтроллера *D1* содержится в файлах *LCDCTRL.HEX* (формат *Intel-HEX*) и *LCDCTRL.BIN* (двоичный образ памяти программ). Программа написана таким образом, что вместо микроконтроллера *ATtiny2313* вполне можно применить устаревшие и снимаемые с производства микроконтроллеры *AT90S1200* (потребуется предварительно запрограммировать т.н. *fuse*-бит *RCEN* чтобы разрешить работу встроенного *RC*-генератора на *1 МГц*) и *AT90S2313* (потребуется установка элементов *ZQ1*, *C1* и *C2*, т.к. этот микроконтроллер не имеет встроенного *RC*-генератора). Микроконтроллер *ATtiny2313* используется со своими начальными заводскими установками, поэтому в модуле индикации он работает на частоте *1 МГц* (встроенный *RC*-генератор на *8 МГц* + делитель частоты на 8). Если для работы микроконтроллера будет выбрана другая тактовая частота, то следует соответственно откорректировать значение константы *PERIOD* (см. исходный текст программы) чтобы частота «регенерации» ЖКИ оставалась в пределах *30..60 Гц*. Не забудьте в этом случае снова транслировать программу (транслятор ассемблера *AVRASM.EXE* есть в приложении, надо просто запустить командный файл *MAKE.BAT*) и, если нет ошибок, записать в память программ микроконтроллера уже новый *HEX*- или *BIN*-файл.

По включении питания, до загрузки в модуль какой-либо информации, на индикаторе отобразится десятичная точка в крайнем правом разряде. Работоспособность модуля сохраняется при снижении напряжения питания до *2,8 В*. Потребляемый ток при напряжении питания *5 В* не превышает *0,4 мА*.

Вместо сдвигового регистра *HEF4015BP* можно применить отечественный *К561ИР2*. Все конденсаторы – керамические. Если в качестве тактового генератора использовать встроенный в микроконтроллер калиброванный *RC*-генератор, то кварцевый резонатор *ZQ1* и конденсаторы *C1* и *C2* устанавливать незачем. Эти элементы устанавливаются только при использовании в качестве интерфейса для загрузки данных асинхронного приемо-передатчика микроконтроллера. Частота кварцевого резонатора, равная *1,8432 МГц*, позволяет более точно устанавливать скорость обмена данными из стандартного ряда интерфейса *RS-232C* в диапазоне от *600 Бод* до *115200 Бод* (см. описание микроконтроллера *ATtiny2313*).

Построенный на базе микроконтроллера модуль индикации может стать основой и для других разработок с отображением информации на ЖКИ, необходимо только разработать соответствующее программное обеспечение.

Подобным образом можно построить контроллер для работы с ЖКИ других типов. Для увеличения разрядности следует просто дополнить цепочку сдвиговых регистров *D2..D4* требуемым количеством аналогичных микросхем. В качестве сдвиговых регистров можно использовать и другие микросхемы. Например, микросхема *SN74HC595N* содержит не только 8-битовый сдвиговый регистр, но и параллельный регистр-защелку, загружаемые в такой регистр данные появляются на его выходах по отдельному сигналу, что гораздо удобнее, особенно при большой разрядности ЖКИ. Не забудьте при этом изменить соответствующим образом и программное обеспечение.