

Применение ЖК-модулей МТ10Т7–7 в измерительных приборах на основе микроконтроллеров серии AVR AT90S8535

В журнале "Схемотехника" №2/2000 была опубликована статья, которая знакомила читателей с довольно удобным модулем ЖК МТ10Е7-7. Она показалась очень интересной, но смутил тот факт, что описано было подключение и работа модуля с микроконтроллером серии MCS-51. Безусловно, эта серия еще достаточно широко используется, однако, по мнению автора, новые разработки целесообразнее делать на базе более современных микроконтроллеров. В частности, в настоящей статье читателю предлагается для рассмотрения вариант работы модуля МТ10Т7–7 с микроконтроллером AT90S8535. Здесь не будет рассматриваться структура самого ЖК-модуля, поскольку это достаточно подробно было сделано Владимиром Уголевым в вышеупомянутой статье. Вместо этого более детально будет рассказано о схеме подключения и программе работы с модулем.

Модуль подключается к порту В микроконтроллера (рис. 1). Выбор порта – произвольный. При желании его легко заменить на любой другой порт данного микроконтроллера, изменив, соответственно, программу управления модулем.

Для удобства работы составим набор подпрограмм для выполнения основных операций с модулем:

```

;*****
;* Подпрограмма: STROB_WR1
;* Описание:
;* Подпрограмма формирования
;stroba по линии WR1
;* WR1: —/—/—
;* Применяется при записи знако-
;места
;*****
STROB_WR1:
    sbi    PORTB,WR1
; Установить "1" на линии WR1
    rcall  delay
; Задержка для установления напряжения
    cbi    PORTB,WR1
; Установить "0" на линии WR1
    ret

;*****
;* Функция: STROB_ADR
;* Описание:
;* Подпрограмма формирования
;stroboв по линиям WR1 и ADR
;* A0 —/—/—
;* WR1 —/—/—
;* Применяется при записи адре-
;са
;*****
STROB_ADR:
    cbi    PORTB,A0
    rcall  delay
    sbi    PORTB,WR1
    rcall  delay
    cbi    PORTB,WR1
    rcall  delay
    sbi    PORTB,A0
    ret

;*****
;* Функция Set_Bus
;* Описание
;* Вывод младшей тетрады
;*****

```

```

;* A на шину DB
;*****
Set_Bus:
; Bit 0
    sbrc    A,0
; Пропустить, если бит 0 в A = '0'
    rjmp    Set1
; Переходим на установку в '1'
    cbi     PORTB,DB0
; Установка DB0='0'
    rjmp    t0
; Пропускаем установку в '1'
    Set1:  sbi     PORTB,DB0
; Установка DB0='1'
    t0:
; Bit 1
    sbrc    A,1
; Пропустить, если бит 1 в A = '0'
    rjmp    Set2
; Переходим на установку в '1'

```

```

    cbi     PORTB,DB1
; Установка DB1='0'
    rjmp    t1
; Пропускаем установку в '1'
    Set2:  sbi     PORTB,DB1
; Установка DB1='1'
    t1:
; Bit 2
    sbrc    A,2
; Пропустить, если бит 2 в A = '0'
    rjmp    Set3
; Переходим на установку в '1'
    cbi     PORTB,DB2
; Установка DB2='0'
    rjmp    t2
; Пропускаем установку в '1'
    Set3:  sbi     PORTB,DB2
; Установка DB2='1'
    t2:
; Bit 3
    sbrc    A,3
; Пропустить, если бит 3 в A = '0'
    rjmp    Set4
; Переходим на установку в '1'
    cbi     PORTB,DB3
; Установка DB3='0'
    rjmp    t3
; Пропускаем установку в '1'
    Set4:  sbi     PORTB,DB3
; Установка DB3='1'
    t3:    ret

```

Так как после включения питания состояние модуля не определено, необходимо произвести начальную инициализацию модуля. Для этого применяем подпрограмму Init_LCD:

```

;*****
;* Функция Init_LCD
;* Описание
;* Инициализация
;* Запись в триггер блокировки
;шины
;* BLK (по адресу 0Fh) значения
;01h
;*****

```

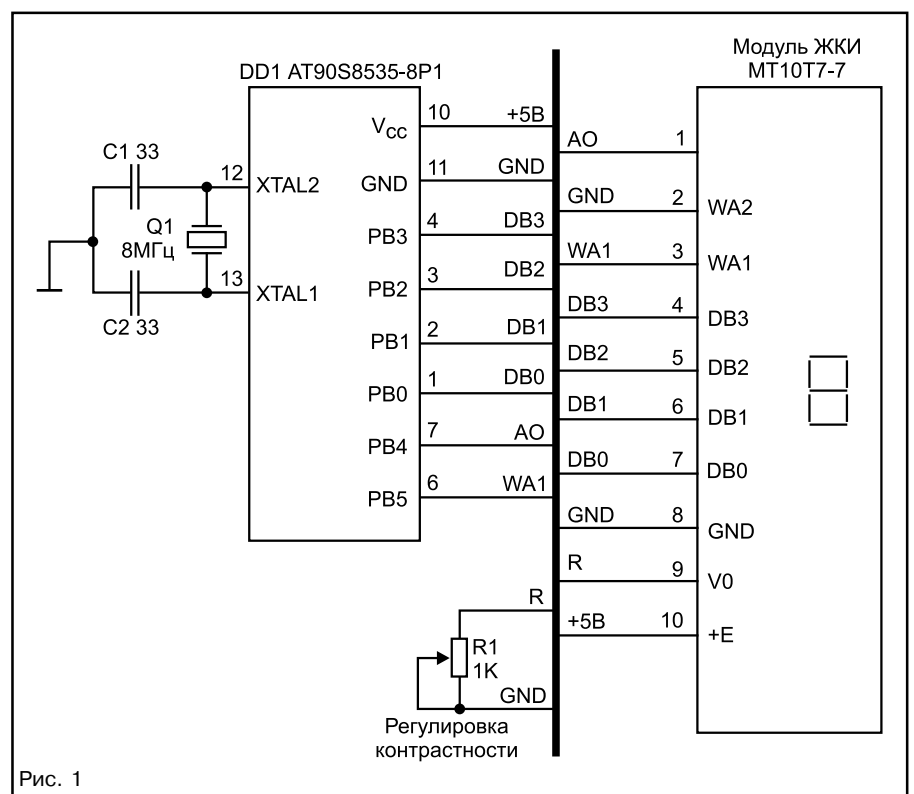


Рис. 1

```

;* и в регистры всех знакомест Sgx
(адреса 0h-9h) нули
;
Init_LCD:
    ldi    A,0x0f
    rcall  Set_Bus    ; Ус-
тановить адрес 0x0f
    rcall  STROB_ADR
    ldi    A,0x01
    rcall  Set_Bus    ; Установить
данные 0x01
    rcall  STROB_WR1
    clr    A          ;
сброс всех знакомест
    rcall  Set_Bus    ; вы-
ставить на шину 0
    rcall  STROB_ADR    ;
формируем строб записи адреса
; На шине уже выставлен 0
; формируем 20 раз строб данных
STROB_WR1
    clr    temp
c20: rcall  STROB_WR1    ;
строб данных
    inc    temp        ;
temp = temp + 1
    cpi    temp,20      ;
сравниваем temp с числом 20
    brne   lhl_int      ;
если не равно, переходим на c20
    ret

```

Поскольку модуль MT10T7-7 имеет четырехразрядную шину данных, для работы с ним потребуется подпрограмма вывода в порт микроконтроллера по тетрадам:

```

*****
;*      Функция out_a
;*      Описание
;*      Вывод в порт по тетрадам
*****
out_a:
    rcall  Set_Bus    ; вы-
ставляем в порт младшую тетраду
    rcall  STROB_WR1    ; ге-
нерируем первый строб записи данных
    rcall  delay        ;
временная задержка
    swap   A           ; пе-
реставление местами тетрад в A
    rcall  Set_Bus    ; вы-
ставляем в порт старшую тетраду
    rcall  STROB_WR1    ; ге-
нерируем второй строб записи данных
    rcall  delay
    ret

```

Для преобразования одноразрядного двоично-десятичного числа в код, который необходимо подать для индикации на модуле данного числа, используем подпрограмму Coder:

```

*****
;*      Функция Code_7
;*      Описание
;*      Преобразование в семисегментный код. Вход: temp; Выход: temp
*****
Code_7:
    ;fcbh, adeg 0xee, 0x2f,
0x60,0x6d,0xe1    ; 0,1,2,3,4
;0xcd,0xcf,0x68,0xef,0xed
; 5,6,7,8,9
    Coder: cpi    temp,0x0
            brne   q1
; если не равно 0
            ldi    temp,0xee

```

```

            rjmp   rend
q1:  cpi    temp,0x1
            brne   q2
; если не равно 0
            ldi    temp,0x60
            rjmp   rend
q2:  cpi    temp,0x2
            brne   q3
; если не равно 0
            ldi    temp,0x2f
            rjmp   rend
q3:  cpi    temp,0x3
            brne   q4
; если не равно 0
            ldi    temp,0x6d
            rjmp   rend
q4:  cpi    temp,0x4
            brne   q5
; если не равно 0
            ldi    temp,0xe1
            rjmp   rend
q5:  cpi    temp,0x5
            brne   q6
; если не равно 0
            ldi    temp,0xcd
            rjmp   rend
q6:  cpi    temp,0x6
            brne   q7
; если не равно 0
            ldi    temp,0xcf
            rjmp   rend
q7:  cpi    temp,0x7
            brne   q8
; если не равно 0
            ldi    temp,0x68
            rjmp   rend
q8:  cpi    temp,0x8
            brne   q9
; если не равно 0
            ldi    temp,0xef
            rjmp   rend
q9:  cpi    temp,0x9
            brne   rend
; если не равно 0
            ldi    temp,0xed
            rjmp   rend
rend: ret

```

Для подготовки числа к индикации используем функцию Prepare:

```

*****
;*      Функция: prepare
;*      Описание
;*      Подпрограмма подготовки чис-
ла к индикации
;*      b20, b21, ..., b29 – ячейки опера-
тивной памяти, содержащие коды цифр
;*      содержащие двоично-десятич-
ные коды цифр
*****
prepare:
; Заменяем двоичный код на семисег-
ментный
    lds    temp,b20
    rcall  Coder
    sts    b20,temp
    lds    temp,b21
    rcall  Coder
    sts    b21,temp
    lds    temp,b22
    rcall  Coder
    sts    b22,temp
    lds    temp,b23
    rcall  Coder
    sts    b23,temp
    lds    temp,b24
    rcall  Coder
    sts    b24,temp
    lds    temp,b25

```

```

    rcall  Coder
    sts    b25,temp
    lds    temp,b26
    rcall  Coder
    sts    b26,temp
    lds    temp,b27
    rcall  Coder
    sts    b27,temp
    lds    temp,b28
    rcall  Coder
    sts    b28,temp
    lds    temp,b29
    rcall  Coder
    sts    b29,temp

```

ret

Функция вывода цифр на индикатор имеет вид:

```

*****
;*      Функция: Display
;*      Описание
;*      Вывод числа на индикатор
*****
Display:
; Подготовка числа к индикации
    rcall  Prepar
    rcall  Set_Bus
    rcall  STROB_ADR
; Выводим последовательно все 10
цифр
    lds    A,b20
    rcall  Out_a
    lds    A,b21
    rcall  Out_a
    lds    A,b22
    rcall  Out_a
    lds    A,b23
    rcall  Out_a
    lds    A,b24
    rcall  Out_a
    lds    A,b25
    rcall  Out_a
    lds    A,b26
    rcall  Out_a
    lds    A,b27
    rcall  Out_a
    lds    A,b28
    rcall  Out_a
    lds    A,b29
    rcall  Out_a
    ret

```

```

*****
;*      Функция: delay
;*      Описание:
;*      Формирование небольшой за-
держки
*****
delay: ldi    temp2,0x0f
dl:    dec    temp2
        brne   dl
        dec    temp1
        brne   delay
        ret

```

Итак, наконец, мы подошли к завершающей части – подпрограмме вывода заданных в 5 байтах оперативной памяти в виде 10 четырехразрядных двоично-десятичных кодов цифр на модуль MT10T7-7:

```

*****
;*      Функция Output_Digit
;*      Описание
;*      подпрограмма вывода заданных
в 5 байтах оперативной памяти
;*      в виде 10 четырехразрядных

```

двоично-десятичных кодов цифр

```
;*      на модуль MT10T7-7
;*      Внимание: для нормальной ра-
боты требует чтобы был установлен стек,
;*      так как используются подпрог-
раммы
```

```
*****
```

```
Output_Digit:
    ser        temp1
```

```
; Инициализация порта B
    out        DDRB,temp1
```

```
; для работы на вывод
    out        PORTB,temp1
```

```
; по всем линиям
```

```
    rcall     Init_LCD
    rcall     Display
```

```
; Собственно, вывод на индикатор
    ret
```

Пример использования данной под-
программы:

```
*****
```

```
;* PROGRAM
*****
```

```
.include "8535def.inc"
.DSEG          ; Буфер для
```

символьного значения для вывода на ЖКИ

```
b20: .BYTE 1
b21: .BYTE 1
b22: .BYTE 1
b23: .BYTE 1
b24: .BYTE 1
b25: .BYTE 1
b26: .BYTE 1
b27: .BYTE 1
```

```
b28: .BYTE 1
b29: .BYTE 1
```

; Описание номеров битов порта B, под-
ключенных к индикатору

```
.equ WR1      = 5
.equ A0        = 4
.equ DB0       = 0
.equ DB1       = 1
.equ DB2       = 2
.equ DB3       = 3
```

```
; Описание переменных
.def A          =r16
.def temp       =r17
.def temp1      =r18
```

```
.def temp2     =r19
```

```
main:
.CSEG
```

; Инициализация стека

```
ldi r16,high(RAMEND)
```

; Старший байт

```
out SPH,r16
```

; Младший байт

```
ldi r16,low(RAMEND)
out SPL,r16
```

```
rcall Init_LCD ;
```

Инициализация индикатора

; Зададим тестовое задание. В ячейках
b20...b29 находятся коды цифр, которые

хотим вывести на индикатор.

```
ld temp,0x0
sts b20,temp
ld temp,0x1
sts b21,temp
ld temp,0x2
sts b22,temp
```

```
ld temp,0x3
sts b23,temp
```

```
ld temp,0x4
sts b24,temp
```

```
ld temp,0x5
sts b25,temp
```

```
ld temp,0x6
sts b26,temp
```

```
ld temp,0x7
sts b27,temp
```

```
ld temp,0x8
sts b28,temp
```

```
ld temp,0x9
sts b29,temp
```

```
rcall OutputDigit
```

; Вызов процедуры индикации

```
end: rjmp end
```

; Зацикливание программы

В результате выполнения программы
на индикаторе должны отобразиться
цифры от 0 до 9.

Следует отметить, что автор не ставил
целью написать оптимальный вариант
данной подпрограммы. В частности,
оптимизировать ее можно следующими
способами:

1 оформив повторяющиеся фраг-
менты в виде циклов или
макросов;

- модифицировав программу для
использования исходных данных в
виде стандартного BCD формата,
т. е. когда в каждом байте
находятся коды двух цифр.

Исходный текст данной программы
читатели могут найти по адресу:

www.platan.ru/shem/