УДК 681.32

С.В. Березняков

ОАО «СТАР», Пермь, Россия

А.В. Греков

Пермский военный институт внутренних войск МВД России, Пермь, Россия

МОДЕЛИРОВАНИЕ МИКРОКОНТРОЛЛЕРА 80С51 В СИСТЕМЕ СХЕМОТЕХНИЧЕСКОГО МОДЕЛИРОВАНИЯ PROTEUS VSM

Proteus является одной из систем схемотехнического моделирования. Proteus VSM позволяет производить отладку не только простейших аналоговых устройств, но и сложных систем, созданных на микроконтроллерах. Основным преимуществом этого программного продукта перед другими симуляторами электронных устройств является то, что никакой другой симулятор не позволяет производить отладку такого количества микроконтроллеров и микропроцессоров. Intel 8051 – это однокристальный микроконтроллер гарвардской архитектуры, который был впервые произведен Intel в 1980 г., для использования во встраиваемых системах. В течение 1980-х и начале 1990-х гг. в был чрезвычайно популярен, однако позже устарел и был вытеснен более современными устройствами, также с 8051-совместимыми ядрами, производимыми более чем 20 независимыми производителями, такими как Atmel, Maxim IC (дочерняя компания Dallas Semiconductor), NXP, Winbond, Silicon Laboratories, Texas Instruments и Cypress Semiconductor). Официальное название 8051-семейства микроконтроллеров Intel – MCS 51. Существует также советский клон данной микросхемы – КР1816ВЕ51. Первые микроконтроллеры из 8051-семейства Intel производились с использованием n-МОП технологии, но следующие версии, содержащие символ «С» в названии, такие как 80С51, использовали КМОП-технологию и потребляли меньшую мощность, чем n-МОП-предшественники (это облегчало их применение для устройств с батарейным питанием). В статье рассматривается реализация временных задержек и прерываний микроконтроллера 80С51 в системе схемотехнического моделирования Proteus VSM. Подобное моделирование будет полезным и в контексте лабораторного занятия, и научно-технического семинара по перспективным направлениям развития элементной базы, особенно в условиях новых задач по импортозамещению.

Ключевые слова: микроконтроллер 80С51, временная задержка, прерывания, таймер, Proteus, моделирование, цикл.

S.V. Bereznyakov

JSC “STAR”, Perm, Russia Federation

A.V. Grekov

Perm military institute of the Internal Troops of the Ministry of the Interior of the Russian Federation, Perm, Russia Federation

SIMULATION OF 80C51 MICROCONTROLLER IN THE SYSTEM OF CIRCUIT SIMULATION PROTEUS VSM

Proteus is a system of circuit simulation. Proteus VSM allows you to debug not only simple analog devices, but also complex systems created on microcontrollers. The main advantage of this software over other simulators of electronic devices is that no other simulator does not allow debugging so many microcontrollers and microprocessors. Intel 8051 is a single chip Harvard architecture, which was first produced by Intel in 1980 for use in embedded systems. During the 1980s and early 1990s it was extremely popular, but later is obsolete and has been superseded by more modern devices, as with the 8051-compatible cores, produced more than 20 independent producers, such as Atmel, Maxim IC (a subsidiary of Dallas Semiconductor), NXP, Winbond, Silicon Laboratories, Texas Instruments and Cypress Semiconductor). The official name of the 8051 family of microcontrollers Intel is MCS 51. There is also a Soviet clone of the chip KR1816VE51. The first microcontrollers from 8051 of Intel produced using n-MOS technology, but the next version with the symbol «C» in the title, such as 80C51, used CMOS technology and consumes less power than the n-MOS predecessors (which facilitates its application for battery powered devices). The article discusses the implementation of the time delays and interruptions 80C51 microcontroller in the system of circuit simulation Proteus VSM. This modeling will be useful in the context of laboratory research, and scientific and technical workshop on promising directions of development of the element base, especially in the face of new challenges on import substitution.

Keywords: 80C51 microcontroller, time delay, interrupt, timer, Proteus, simulation, cycle.

Введение. Сегодня существует множество программ-симуляторов, заменяющих реальную радиоэлектронную аппаратуру (РЭА) виртуальными моделями [1, 2]. Такие программы позволяют без проведения сборки реального устройства выполнить отладку работы схемы [3], найти ошибки, допущенные на стадии проектирования, снять требуемые параметры и многое другое [4, 5].

Одной из таких систем схемотехнического моделирования является Proteus [6, 7]. Моделирование работы электронных компонентов – не единственная способность программы. Proteus VSM, созданная фирмой Labcenter Electronics на основе ядра SPICE3F5 университета Berkeley, представляет собой среду сквозного проектирования. Это означает создание устройства, начиная с графического изображения его принципи-

альной схемы и заканчивая изготовлением печатной платы устройства с возможностью контроля на каждом этапе производства [8, 9].

Пользователю доступна обширная библиотека моделей элементов, пополнять которую можно самостоятельно, естественно, для этого нужно досконально знать работу элемента и уметь программировать. Имеется достаточный набор инструментов и функций, среди которых вольтметр, амперметр, осциллограф, всевозможные генераторы, способность отлаживать программное обеспечение микроконтроллеров и многие другие средства.

Proteus VSM включает в себя более 6000 электронных компонентов со всеми справочными данными, а также демонстрационные ознакомительные проекты. Программа имеет инструменты USBCONN и COMPIM, которые позволяют подключить виртуальное устройство к портам USB и COM компьютера [10–14]. При подсоединении к этим портам любого внешнего прибора виртуальная схема будет работать с ним, как если бы она существовала в реальности. Proteus VSM поддерживает следующие компиляторы: CodeVisionAVR и WinAVR (AVR), ICC (AVR, ARM7, Motorola), HiTECH (8051, PIC Microchip) и Keil (8051, ARM) [15–21]. Существует возможность экспорта моделей электронных компонентов из программы PSpice.

Proteus VSM состоит из двух самостоятельных программ: ISIS – программа синтеза и моделирования непосредственно электронных схем и ARES – программа разработки печатных плат. Кроме того, в состав восьмой версии входит среда разработки VSM Studio, позволяющая быстро написать программу для микроконтроллера, используемого в проекте, и осуществить компиляцию.

2. Программная реализация временной задержки. Процедура реализуется методом программных циклов. В некоторый регистр загружается число, и оно инкрементируется (циклически уменьшается на единицу). Выход из цикла организуется при достижении нуля. Задержка определяется величиной этого числа с учетом времени выполнения команд. Необходимо знать время выполнения команд в машинных циклах – эта информация есть в таблице системы команд. Например, рассмотрим подпрограмму WAIT:

```
WAIT: MOV R6,#X
W1:       DJNZ R6,W1
RET
```

В регистр 6 загружается число *X*, затем командой DJNZ R6,W1 организуется цикл – декремент с проверкой нулевого содержимого регистра. Известно, что подпрограмма вызывается командой CALL, ее длительность 2 машинных цикла, длительность команды MOV – 1 машинный цикл, длительность команды DJNZ – 2 машинных цикла, RET – 2 машинных цикла.

При тактовой частоте 12 МГц один машинный цикл, состоящий из 12 периодов синхронизации, составляет:

$$\frac{12 \cdot 1}{12 \cdot 10^6 \frac{1}{\text{с}}} = 1 \cdot 10^{-6}\text{с} = 1 \text{ мкс}.$$

Таким образом, программа выполняется за $2 + 1 + 2X + 2$ мкс. Для временной выдержки, например 125 мкс, необходимое число циклов вычисляется так: $X = (125 - 5)/2 = 60$.

В данном случае для реализации требуемой временной выдержки в регистр 6 загружается десятичное число 60. Задержка реализуется точно. Если число *X* получается дробным, то для точной доводки можно применить холостые команды NOP.

Минимальная задержка, получаемая подпрограммой WAIT, равна: $2 + 1 + 2 + 2 = 7$ мкс, максимальная: $2 + 1 + 2 \cdot 255 + 2 = 515$ мкс.

Для увеличения временной задержки нужно организовать вложенные циклы. Например, следующим образом:

```
WAIT2:     MOV R6,#255
W1:        MOV R7,#0FFH
W2:        DJNZ R7,W2
DJNZ R6,W1
RET
```

Для еще большей задержки можно организовать еще один цикл, в котором вызывается подпрограмма WAIT2.

Программа выдачи кода с задержкой – программа PRG6.

```
BEGIN:
P1 EQU 90H ; определение порта P1
P2 EQU 0A0H     ; определение порта P2
P0 EQU 80H ; определение порта P0
P3 EQU 0B0H     ; определение порта P3
M1:   MOV P2,P1 ; ввод с порта P1, вывод на P2
MOV R7,#30 ; загрузка константы 30 в R7
```

```
    M3:   MOV R6,#255; загрузка константы 255 в R6
    M4:   MOV R5,#255; загрузка константы 255 в R5
    M5:   DJNZ R5,M5; декремент R5 и переход, если
не 0
    DJNZ R6,M4 ; декремент R6 и переход
    DJNZ R7,M3 ; декремент R7 и переход
    JMP M1          ; 4 переход на начало
    END
```

Необходимо помнить о том, что работа осуществляется не на реальном контроллере, а на модели, в которой время машинного цикла контроллера определяется моделирующей программой (рис. 1).

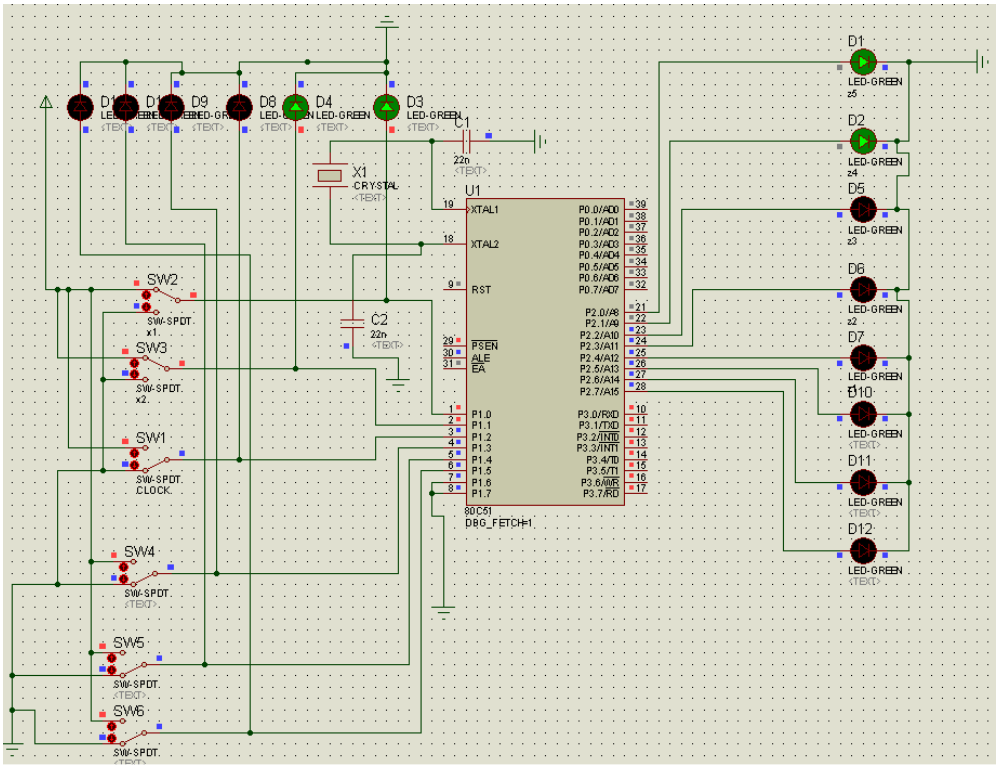

Рис. 1. Работа программы выдачи кода с задержкой

В рассматриваемом примере задержка составляет примерно 3–4 с.

2. Реализация временной задержки с помощью таймера. В МК51 два программируемых 16-битных таймера-счетчика T/C0, T/C1 могут быть использованы в качестве таймеров или счетчиков внешних событий. При работе в качестве таймеров содержимое инкрементируется

в каждом машинном цикле, т.е. есть каждые 12 периодов кварцевого резонатора, при работе в качестве счетчика – под воздействием перехода из 1 в 0 внешнего входного сигнала T0, T1. Оба режима могут быть использованы для реализации временных задержек. Счетчик T/C0 можно использовать для счета внешних сигналов, которые подаются на вход T0 порта P3 (рис. 2).

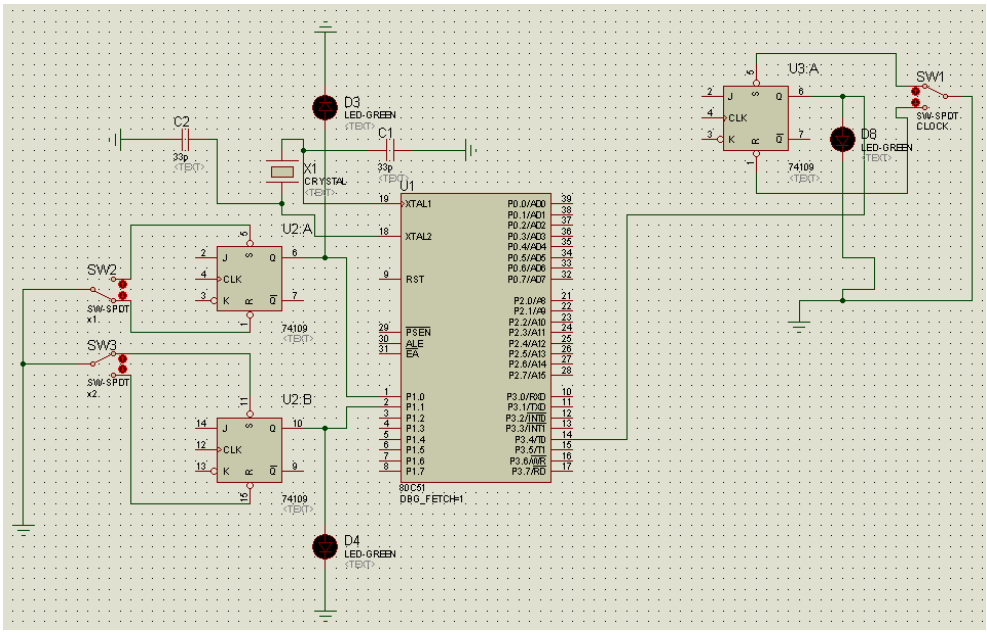

Рис. 2. Работа программы счетчика

Разработаем программу, в которой предусмотрим настройку счетчика T/C0 для счета. Это выполняется с помощью регистра режимов работы TMOD (табл. 1).

Таблица 1

Регистр режимов работы TMOD

№	Символ	Позиция	Имя и назначение
1	2	3	4
	GATE	TMOD.7	Управление блокировкой. Если бит установлен, то таймер/счетчик 1 разрешен до тех пор, пока на входе INT1 высокий уровень и бит управления TR1 установлен. Если бит сброшен, то T/C1 разрешается, как только бит управления TR1 устанавливается.

Окончание табл. 1

1	2	3	4
6	C/T	TMOD.6	Бит выбора режима таймера или счетчика событий 1. Если бит сброшен, то работает таймер от внутреннего генератора. Если бит установлен, то работает счетчик от внешних сигналов на входе T1.
5	M1	TMOD.5	Выбор режима работы T/C1.
4	M0	TMOD.4	Выбор режима работы T/C1.
3	GATE	TMOD.3	Управление блокировкой. Если бит установлен, то таймер/счетчик 0 разрешен до тех пор, пока на входе INT0 высокий уровень и бит управления TR0 установлен. Если бит сброшен, то T/C0 разрешается, как только бит управления TR0 устанавливается.
2	C/T	TMOD.2	Бит выбора режима таймера или счетчика событий 0. Если бит сброшен, то работает таймер от внутреннего генератора. Если бит установлен, то работает счетчик от внешних сигналов на входе T0.
1	M1	TMOD.1	Выбор режима работы T/C0.
0	M0	TMOD.0	Выбор режима работы T/C0.

Таблица режимов работы имеет вид (табл. 2).

Таблица 2

Режимы работы

M1	M0	Режим работы
0	0	Таймер MK48. TL работает как 5-битный предделитель.
0	1	16-битный таймер-счетчик. TH, TL включены последовательно.
1	0	8-битный автоперезагружаемый таймер-счетчик. TH хранит значение, которое должно быть перезагружено в TL каждый раз по переполнению.
1	1	Таймер-счетчик 1 останавливается. Таймер-счетчик 0: TL0 работает как 8-битный таймер-счетчик, и его режим определяется управляющими битами таймера 0. TH0 работает как 8-битный таймер-счетчик, и его режим определяется управляющими битами таймера 1.

Проанализировав эту информацию, можно сделать вывод, что необходимо всего лишь установить бит TMOD.2. Однако, как ни странно, TMOD не допускает адресацию отдельных бит, поэтому придется

использовать команду MOV TMOD, #00000100B, которая как раз и обеспечивает установку 2-го бита. Кроме того, если требуется установить режим, отличный от нулевого, то необходимо сформировать соответствующий непосредственный операнд. Например, для первого режима T/C0: 00000101B. Далее необходимо запустить счетчик. Для этого используется регистр управления/ таймеров-счетчиков TCON (табл. 3).

Таблица 3

Регистр TCON

№	Символ	Позиция	Имя и назначение
7	TF1	TCON.7	Флаг переполнения таймера 1. Устанавливается аппаратурно при переполнении. Сбрасывается аппаратурно при обслуживании прерывания
6	TR1	TCON.6	Бит управления таймера 1. Устанавливается / сбрасывается программно для пуска/останова
5	TF0	TCON.5	Флаг переполнения таймера 0. Устанавливается аппаратурно при переполнении. Сбрасывается аппаратурно при обслуживании прерывания
4	TR0	TCON.4	Бит управления таймера 0. Устанавливается / сбрасывается программно для пуска/останова
3	IE1	TCON.3	Флаг фронта прерывания 1. Устанавливается аппаратно по срезу INT1. Сбрасывается при обслуживании прерывания
2	IT1	TCON.2	Бит управления типом прерывания 1. Устанавливается / сбрасывается программно для спецификации запроса INT1 (срез/низкий уровень)
1	IE0	TCON.1	Флаг фронта прерывания 0. Устанавливается аппаратно по срезу INT0. Сбрасывается при обслуживании прерывания
0	IT0	TCON.0	Бит управления типом прерывания 0. Устанавливается/ сбрасывается программно для спецификации запроса INT0 (срез/низкий уровень)

Таким образом, для включения T/C0 необходимо установить бит TCON.4. Этот регистр, в отличие от предыдущего, допускает адресацию отдельных бит, поэтому нужна команда SETB TCON.4.

Для отображения состояния счетчика следует предусмотреть также вывод его содержимого на порты P2, P0 и возврат к началу.

В результате получим программу PRG8:

```
BEGIN:          // счет внешних сигналов с по-
мощью таймера T/C0
P1 EQU 90H              ; определение порта P1
P2 EQU 0A0H             ; определение порта P2
P0 EQU 80H              ; определение порта P0
P3 EQU 0B0H             ; определение порта P3
TCON EQU 88H                ; определение TCON
TMOD EQU 89H                ; определение TMOD
TL0 EQU 8AH                 ; определение TL0
TH0 EQU 8CH                 ; определение TH0
MOV TMOD,#00000101B  ; разрешение счета событий
SETB TCON.4                 ; старт T/C0
M1:   MOV P2,TL0                ; вывод TL0
MOV P0,TH0              ; вывод TH0
JMP M1
END
```

Переключая ключ SW1 и анализируя состояние порта P2, убедимся, что счетчик «считает» (рис. 3).

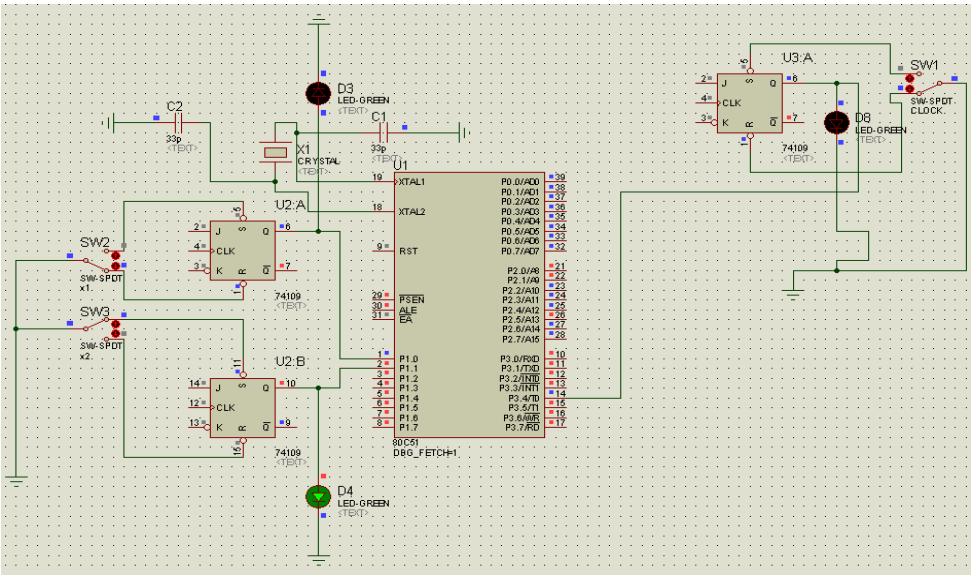

Рис. 3. Работа программы счетчика с отображением состояния

Используем таймеры в режиме счета от внутреннего генератора. Организуем «бегущую единицу» с задержкой 64 мс.

```
BEGIN:                  // T/C0 – задержка 64 мс
P1 EQU 90H          ; определение порта P1
P2 EQU 0A0H             ; определение порта P2
P0 EQU 80H          ; определение порта P0
P3 EQU 0B0H             ; определение порта P3
TCON EQU 88H            ; определение TCON
TMOD EQU 89H            ; определение TMOD
TL0 EQU 8AH             ; определение TL0
TH0 EQU 8CH             ; определение TH0
MOV R0,#1               ; загрузке 1 в R0
MOV P2, R0          ; вывод на порт P2
NEXT: MOV A, R0         ; передача R0 в A
RL A                ; сдвиг A влево циклический
MOV R0,A                ; возвращение R0
MOV P2,R0               ; вывод R0 в порт P2
CALL DELAY          ; вызов подпрограммы задержки
JMP NEXT                ; возврат
DELAY: MOV TMOD,#01H        ; настройка T/C0
MOV TL0,#LOW(NOT(64000-1)); загрузка таймера
MOV TH0,#HIGH(NOT(64000-1)); загрузка таймера
SETB TCON.4                     ; старт T/C
M1:   JNB TCON.5, M1 ; ожидание окончания счета
CLR TCON.4                  ; очистка T/C0
RET         ; выход из подпрограммы задержки
END
```

Получилась «бегущая единица» с аппаратной задержкой (рис. 4).

Для использования таймеров иногда необходимы прерывания. Рассмотрим порядок их реализации.

3. Реализация прерываний. Прерывания организуются от внешних входов INT0, INT1, от таймеров TF0, TF1 и от приемника T1 или передатчика R1.

Таблица векторов прерываний, переход на которые осуществляется аппаратно, имеет вид (табл. 4).

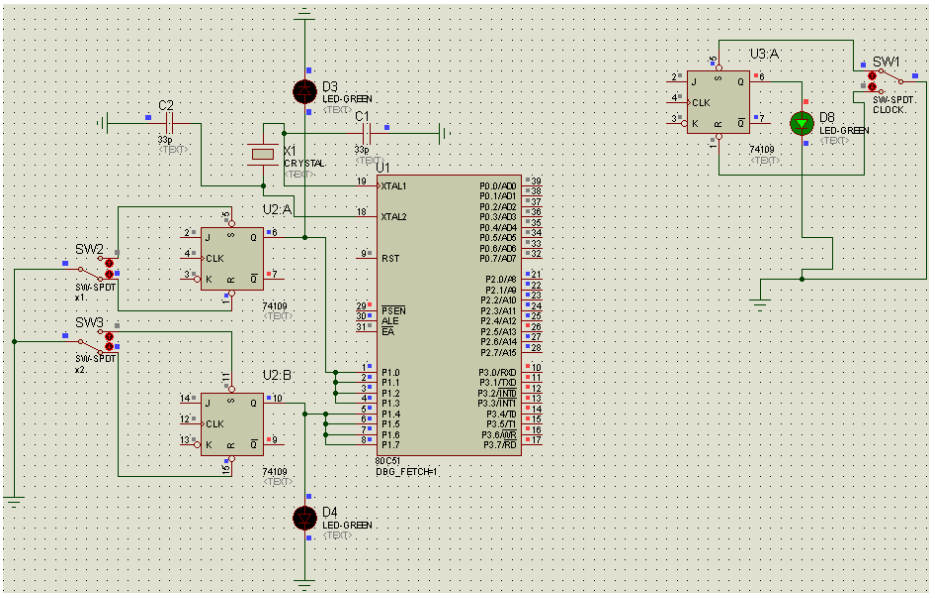

Рис. 4. Работа программы «бегущая единица»

Таблица 4

Векторы прерываний

Сигнал	Порядок опроса приравенстве приоритетов	Адрес вектора
INT0	1	0003H
TF0	2	000BH
INT1	3	0013H
TF1	4	001BH
T1 или R1	5	0023H

Разрешение прерываний устанавливает регистр масок прерываний IE (табл. 5).

Таблица 5

Регистр масок прерываний IE

Символ	Позиция	Назначение
EA	IE.7	Снятие блокировки прерываний. Сбрасывается программно независимо от IE.4 – IE.0
–	IE.6	Не используется
–	IE.5	Не используется
ES	IE.4	Разрешение прерывания от приемопередатчика – программно для разрешения прерывания от флагов T1, R1
ET1	IE.3	Разрешение прерывания от таймера 1. Программно
EX1	IE.2	Разрешение внешнего прерывания 1. Программно
ET0	IE.1	Разрешение прерывания от таймера 0. Программно
EX0	IE.0	Разрешение внешнего прерывания 0. Программно

Приоритеты прерываний могут изменяться в соответствии с регистром приоритетов IP (табл. 6).

Таблица 6

Регистр приоритетов IP

Символ	Позиция	Назначение
–	IP.7	Не используется
–	IP.6	Не используется
–	IP.5	Не используется
PS	IP.4	Приоритет приемопередатчика. Установка-сброс программно для установки высшего/низшего приоритета
PT1	IP.3	Приоритет таймера 1
PX1	IP.2	Приоритет внешнего прерывания 1
PT0	IP.1	Приоритет таймера 0
PX0	IP.0	Приоритет внешнего прерывания 0

Рассмотрим «бегущую единицу» с реализацией временной задержки 64 мс на основе прерываний.

```
    BEGIN:     // сдвиг бита P2 при переполнении
T/C0 – задержка 64 мс
    P1 EQU 90H      ; определение порта P1
    P2 EQU 0A0H          ; определение порта P2
    P0 EQU 80H      ; определение порта P0
    P3 EQU 0B0H          ; определение порта P3
    TCON EQU 88H         ; определение TCON
    TMOD EQU 89H         ; определение TMOD
    IE EQU 0A8H          ; определение IE
    TL0 EQU 8AH          ; определение TL0
    TH0 EQU 8CH          ; определение TH0
    ORG 0
    SETB IE.7  ; разрешение прерываний
    MOV R0,#1  ; загрузка 1 в регистр R0
    MOV P2,R0  ; вывод R0 в P2
    JMP NEXT
    ORG 0BH    ; адрес вектора прерываний от T/C0
    JMP MET
    ORG 100H        ; начальный адрес подпрограммы
    MET:  MOV A,R0       ; передача из R0 в A
    RL A            ; сдвиг A влево циклический
    MOV R0,A        ; передача из A в R0
```

```
    MOV P2,A          ; передача из A в P2
    MOV P2,R0
    MOV TMOD,#00000001B       ; настройка T/C0
    MOV TL0,#LOW(NOT(64000-1)); загрузка таймера
    MOV TH0,#HIGH(NOT(64000-1)); загрузка таймера
    SETB TCON.4                    ; старт T/C0
    RETI       ; выход из подпрограммы обработки
прерывания
    NEXT:MOV  TMOD,#00000001B; первоначальная на-
стройка T/C0
    MOV  TL0,#LOW(NOT(50000-1)); первоначальная за-
грузка таймера
    MOV TH0,#HIGH(NOT(50000-1)); первоначальная за-
грузка таймера
    SETB TCON.4     ; первоначальный старт T/C0
    SETB IE.1       ; разрешение прерывания от T/C0
    NEXT1:JMP NEXT1 ; переход, петля, ожидание пре-
рывания
    END
```

На рис. 5 проиллюстрирован результат работы программы «бегущая единица» с реализацией временной задержки 64 мс на основе прерываний.

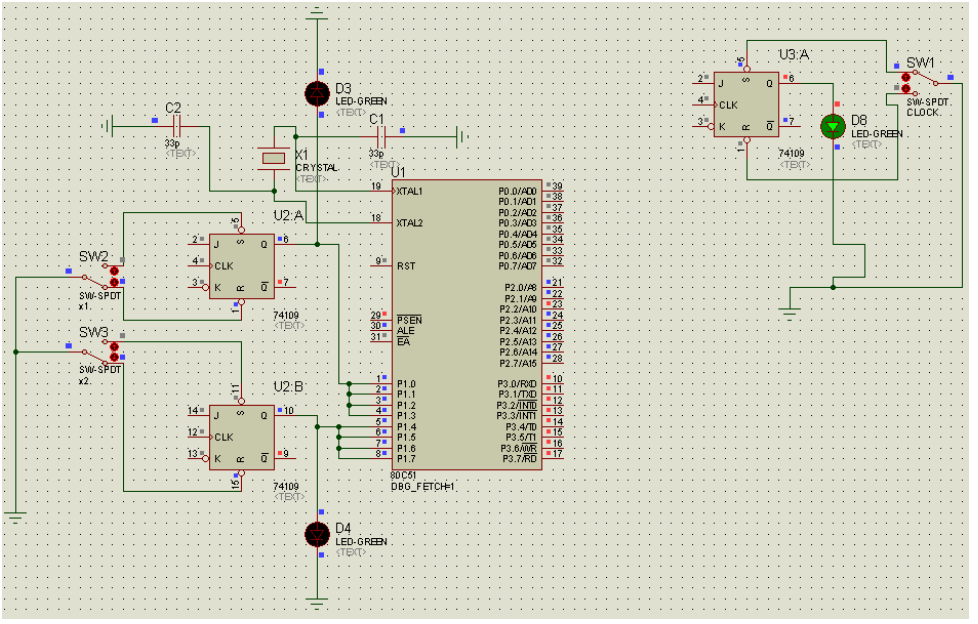

Рис. 5. Работа программы «бегущая единица» с реализацией временной задержки 64 мс на основе прерываний

Выводы

Таким образом, в системе схемотехнического моделирования Proteus VSM рассмотрена реализация временных задержек программно и с помощью таймера, а также прерываний для микроконтроллера 80С51. Результаты работы могут быть использованы при проведении практических и лабораторных занятий по моделированию компонентов радиоэлектронной аппаратуры.

Библиографический список

1. Угрюмов Е.П. Цифровая схемотехника: учеб. пособие для вузов. – 2-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2007. – 800 с.

2. Потемкин И.С. Функциональные узлы цифровой автоматики. – М.: Энергоатомиздат, 1988. – 320 с.

3. Тюрин С.Ф. Вычислительная техника и информационные технологии. Цифровая схемотехника: учеб. пособие. – Пермь: Изд-во Перм. гос. техн. ун-та, 2008. – 137 с.

4. Тюрин С.Ф., Гончаровский О.В., Громов О.А. Вычислительная техника и информационные технологии. Аппаратные средства вычислительной техники: консп. лекций. – Пермь: Изд-во Перм. гос. техн. ун-та, 2011. – 267 с.

5. Тюрин С.Ф., Греков А.В., Громов О.А. Реализация цифровых автоматов в системе Quartus фирмы Altera: учеб. пособие. – Пермь: Изд-во Перм. гос. техн. ун-та, 2011. – 148 с.

6. Proteus Schematic Capture [Электронный ресурс]. – URL: http://www.labcenter.com/products/pcb/schematic_intro.cfm (дата обращения: 05.01.2016).

7. Device Libraries [Электронный ресурс]. – URL: http://www.labcenter.com/products/pcb/schematic_libraries.cfm (дата обращения: 05.01.2016).

8. Hierarchical Design [Электронный ресурс]. – URL: http://www.labcenter.com/products/pcb/schematic_hierarchy.cfm (дата обращения: 05.01.2016).

9. Установка Proteus [Электронный ресурс]. – URL: http://hamlab.net/mcu/training/proteus.html (дата обращения: 05.01.2016).

10. Proteus VSM for 8051 [Электронный ресурс]. – URL: http://www.labcenter.com/products/vsm/8051.cfm (дата обращения: 05.01.2016).

11. Labcenter Electronics Proteus VSM [Электронный ресурс]. – URL: http://www.keil.com/uvision/agsi/193.htm (дата обращения: 05.01.2016).

12. Proteus VSM: digital vs analog resistor and LED [Электронный ресурс]. – URL: http://electronics.stackexchange.com/questions/ 172651/proteus- vsm-digital-vs-analog-resistor-and-led (дата обращения: 05.01.2016).

13. Система моделирования ISIS Proteus [Электронный ресурс]. – URL: http://easyelectronics.ru/sistema-modelirovaniya-isis-proteus-bystryj-start.html (дата обращения: 05.01.2016).

14. ProSPICE Advanced Simulation Option [Электронный ресурс]. – URL: http://www.labcenter.com/products/advancedsim.cfm (дата обращения: 05.01.2016).

15. ARMKeil Product Overview [Электронный ресурс]. – URL: http://www.keil.com/c51/ (дата обращения: 05.01.2016).

16. MCU 8051 IDE. Introduction [Электронный ресурс]. – URL: http://www.moravia-microsystems.com/mcu-8051-ide/ (дата обращения: 05.01.2016).

17. Архитектура микроконтроллера 8051 [Электронный ресурс]. – URL: http://www.atsv.ru/articles/8051.htm (дата обращения: 05.01.2016).

18. ARMKeil Instructions [Электронный ресурс]. – URL: http://www.keil.com/support/man/docs/is51/is51_instructions.htm (дата обращения: 05.01.2016).

19. Микроконтроллеры на базе архитектуры 8051 [Электронный ресурс]. – URL: http://www.atmel.com/ru/ru/products/microcontrollers/ 8051architecture/ (дата обращения: 05.01.2016).

20. 80C51 Datasheet [Электронный ресурс]. – URL: http://www.alldatasheet.com/view.jsp?Searchword=80c51 (дата обращения: 05.01.2016).

21. Texas Instruments 8051-Based MCUs [Электронный ресурс]. – URL: http://www.ti.com/mcu/docs/mcugeneralcontent.tsp?sectionId=98&tabId=1515 (дата обращения: 05.01.2016).

References

1. Ugriumov E.P. Tsifrovaia skhemotekhnika [Digital circuitry engineering]. Saint Petersburg: BKhV-Peterburg, 2007. 800 p.

2. Potemkin I.S. Funktsional'nye uzly tsifrovoi avtomatiki [The functional components of digital automation]. Moscow: Energoatomizdat, 1988. 320 p.

3. Tiurin S.F. Vychislitel'naia tekhnika i informatsionnye tekhnologii. Tsifrovaia skhemotekhnika [Computer science and information technologies. Digital circuitry engineering]. Permskii gosudarstvennyi tekhnicheskii universitet, 2008. 137 p.

4. Tiurin S.F., Goncharovskii O.V., Gromov O.A. Vychislitel'naia tekhnika i informatsionnye tekhnologii. Apparatnye sredstva vychislitel'noi tekhniki [Computer science and information technologies. Computer hardware]. Permskii gosudarstvennyi tekhnicheskii universitet, 2011. 267 p.

5. Tiurin S.F., Grekov A.V., Gromov O.A. Realizatsiia tsifrovykh avtomatov v sisteme Quartus firmy Altera [The implementation of digital machines in Altera Quartus system]. Permskii gosudarstvennyi tekhnicheskii universitet, 2011. 148 p.

6. Proteus Schematic Capture, available at: http://www.labcenter.com/products/pcb/schematic_intro.cfm (accessed 05 January 2016).

7. Device Libraries, available at: http://www.labcenter.com/products/pcb/schematic_libraries.cfm (accessed 05 January 2016).

8. Hierarchical Design, available at: http://www.labcenter.com/products/pcb/schematic_hierarchy.cfm (accessed 05 January 2016).

9. Ustanovka Proteus [Proteus installation], available at: http://hamlab.net/mcu/training/proteus.html (accessed 05 January 2016).

10. Proteus VSM for 8051, available at: http://www.labcenter.com/products/vsm/8051.cfm (accessed 05 January 2016).

11. Labcenter Electronics Proteus VSM, available at: http://www.keil.com/uvision/agsi/193.htm (accessed 05 January 2016).

12. Proteus VSM: digital vs analog resistor and LED, available at: http://electronics.stackexchange.com/questions/172651/proteus-vsm-digital-vs-analog-resistor-and-led (accessed 05 January 2016).

13. Sistema modelirovaniia ISIS Proteus [ISIS Proteus simulation system], available at: http://easyelectronics.ru/sistema-modelirovaniya-isis-proteus-bystryj-start.html (accessed 05 January 2016).

14. ProSPICE Advanced Simulation Option, available at: http://www.labcenter.com/products/advancedsim.cfm (accessed 05 January 2016).

15. ARMKeil Product Overview, available at: http://www.keil.com/c51/ (accessed 05 January 2016).

16. MCU 8051 IDE. Introduction, available at: http://www.moravia-microsystems.com/mcu-8051-ide/ (accessed 05 January 2016).

17. Arkhitektura mikrokontrollera 8051 [The microcontroller 8051 architecture], available at: http://www.atsv.ru/articles/8051.htm (accessed 05 January 2016).

18. ARMKeil Instructions, available at: http://www.keil.com/support/man/docs/is51/is51_instructions.htm (accessed 05 January 2016).

19. Mikrokontrollery na baze arkhitektury 8051 [Microcontrollers based on 8051 architecture], available at: http://www.atmel.com/ru/ru/products/microcontrollers/ 8051architecture/ (accessed 05 January 2016).

20. 80C51 Datasheet, available at: http://www.alldatasheet.com/view.jsp?Searchword=80c51 (accessed 05 January 2016).

21. Texas Instruments 8051-Based MCUs, available at: http://www.ti.com/mcu/docs/mcugeneralcontent.tsp?sectionId=98&tabId=1515 (accessed 05 January 2016).

Сведения об авторах

Березняков Сергей Вадимович (Пермь, Россия) – кандидат технических наук, начальник отдела ОАО «СТАР» (614990, Пермь, ул. Куйбышева, 140А, тел. +7-912-883-26-32; e-mail: berser22@mail.ru).

Греков Артем Владимирович (Пермь, Россия) – кандидат технических наук, доцент кафедры программного обеспечения вычислительной техники и автоматизированных систем Пермского военного института МВД РФ (614112, Пермь, ул. Гремячий Лог, 1, тел. +7-909-118-45-82; e-mail: grekartemvl@mail.ru).

About the authors

Bereznyakov Sergey Vadimovich (Perm, Russian Federation) is a Ph.D. in Technical Sciences, Head of Department JSC “Star” (614990, Perm, 140A, Kuibyshev St., tel.: +7-912-883-26-32; e-mail: berser22@mail.ru).

Grekov Artem Vladimirovich (Perm, Russian Federation) is a Ph.D. in Technical Sciences, Associate Professor at the Department of Software computer technology and automated systems, Perm military institute of the Internal Troops of the Ministry of the Interior of the Russian Federation (614112, Perm, Gremyachy Log St., 1, tel.: +7-909-118-45-82; e-mail: grekartemvl@mail.ru).

Получено 20.02.2016