Кафедра электронно-вычислительной аппаратуры

Курсовая работа по дисциплине

«Микропроцессорные системы»

##### Тема “Счетчик обратного отсчета”

Москва – 2007

# Оглавление

Техническое задание

Описание работы программы

Текст программы

# Техническое задание

Разработать программу на языке Assembler, отсчитывающую время от заданного значения до 0. Значение может быть от 0001 до 9999.

2 режима работы:

1) режим установки значения

2) режим отсчета

Ввод необходимо осуществлять с помощью стандартной шестнадцатикнопочной терминальной клавиатуры. Переход в режим установки времени необходимо осуществлять нажатием клавиши 0. Выход из режима установки времени должен происходить автоматически по окончании ввода значений всех регистров.

# Описание работы программы

## 

## Логика построения программы

Исходя из ТЗ сформируем следующую блок-схему работы основной программы:



## Устройства ввода

Для ввода данных используется стандартная шестнадцатикнопочная клавиатура, подключаемая к параллельному порту (P1). Она имеет следующий вид:



Чтобы определить, нажата или нет какая-либо клавиша-ключ, сначала нужно выставить на линии регистра-защёлки P1, которые соединяет клавиша, разные логические значения. Затем необходимо считать из регистра значения этих линий: если оба бита стали равными «0», то значит, проверяемая клавиша была нажата.

Для сокращения процедуры опроса клавиатуры будем делать опрос по «строкам»:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Клавиша** | **Слово, выставленное на P1** | | | | | | | | **Ожидаемое на P1 слово** | | | | | | | | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | **‘1’** | 1 | 1 | **0** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |  | 1 | 1 | 0 | 1 | | **‘2’** | 1 | 1 | **0** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | | **‘3’** | 1 | 1 | **0** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | | **‘4’** | 1 | 1 | 1 | **0** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | | **‘5’** | 1 | 1 | 1 | **0** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | | **‘6’** | 1 | 1 | 1 | **0** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | | **‘7’** | **0** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | | **‘8’** | **0** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | | **‘9’** | **0** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | | **‘0’** | 1 | **0** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | |

Чтобы просто зафиксировать факт нажатия любой из цифровых клавиш, достаточно выставить в регистр P1 слово #00001111b и ожидать на P1 значение #0000**x**1**xx**b, где x∈{0;1}.

Особенностью данной клавиатуры, которую надо учитывать при программировании, является наличие переходного процесса (дребезга) с длительностью в несколько раз большей, чем время машинного цикла ОМЭВМ МК8051:



# Текст программы

ORG 8000h;

jmp lbMain

ORG 800Bh;

jmp intTF0

lbMain:

; конфигурируем таймер T0:

anl TMOD, #0F0h; обнуляем младшую тетраду

orl TMOD, #01h; младшую тетраду приводим к виду "0001"

clr TR0; остановка таймера T0 (таймер - не считает)

; загружаем старший и младший байты регистра таймера T0 нулями:

mov TH0, #0

mov TL0, #0

; настраиваем систему прерываний:

mov IP, #0; все прерывания (пока) - с одинаковым низким уровнем приоритета

mov IE, #10000010b; IEN0: выставили биты EA(7),ES(4),ET0(1)

;mov 0E8h, #00001011b; IEN1: выставили биты 0,1,3

mov P1, #00001111b

mov 0EBh,#0FFh;

;;;; ОПИСАНИЕ ПЕРЕМЕННЫХ: ;;;;

;регистр R0 - единицы

;регистр R1 - десятки

;регистр R2 - сотни

;регистр R3 - тысячи

;регистры R4, R5, R6 - для разных промежуточных значений

;регистр R7 - буфер для хранения значения нажатой клавиши [#0..#9] или

; значения #FFh, если клавиша не нажата

CR equ 0Dh; "возврат каретки"

LF equ 0Ah; "перевод строки"

ESC equ 1Bh; "конец сообщения"

Buf\_R0 equ 20h

Buf\_R1 equ 21h

Buf\_R2 equ 22h

Buf\_R3 equ 23h

;;;; ТЕЛО ПРОГРАММЫ: ;;;;

;задаём начальное время:

mov R0, #0;

mov R1, #2;

mov R2, #1;

mov R3, #0;

mov Buf\_R0, R0

mov Buf\_R1, R1

mov Buf\_R2, R2

mov Buf\_R3, R3

mov DPTR, #msgShowTime\_Mode

call prShowMessage; вывод сообщения msgShowTime\_Mode

setb TR0

mov R4, #13

lbMainLoop:

nop

clr A

add A, R0

add A, R1

add A, R2

add A, R3

jz lbFinal

call prIs\_SetTime\_Mode;

jmp lbMainLoop

;call prDelay;

;call prDisplay;

;call prIs\_SetTime\_Mode;

intTF0:

clr TF0

djnz R4, lbTF0\_End;

; реализация обратного отсчёта:

dec R0

cjne R0, #0FFh, lbTF0\_Next1;

mov R0, #9

dec R1

cjne R1, #0FFh, lbTF0\_Next1;

mov R1, #9

dec R2

cjne R2, #0FFh, lbTF0\_Next1;

mov R2, #9

dec R3

cjne R3, #0FFh, lbTF0\_Next1;

lbTF0\_Next1: mov R4, #13

lbTF0\_End: call prDisplay

reti

lbFinal:

clr TR0

mov DPTR, #msgFinal

call prShowMessage

;call prIs\_SetTime\_Mode;

jmp $

;;;; ОПИСАНИЕ ПРОЦЕДУР: ;;;;

;; процедура prDelay:

; lb5: mov R5, #1;

;

; lb3: djnz R6, lb3

; djnz R4, lb3

; djnz R5, lb3

;

; ret

;

; prDelay:

; mov R6, #0

; mov R4, #0

; mov R5, #6

; call lb3

;

; mov R4, #244; (!!!)

; call lb5

;

; mov R4, #1

; mov R6, #118; (!!!) Это следует менять для подстройки задержки!!!

; call lb5

;

; ret

;; процедура prSmartDelay:

;; небольшая задержка, за время которой на клавиатуре успевает

;; завершиться "переходный процесс"

prSmartDelay:

mov R6, #0

mov R5, #0

lbSmartDelay\_Loop:

djnz R5, $

djnz R6, lbSmartDelay\_Loop

ret

;; процедура prDisplay:

;; [2 на вызов]+[2 на возврат]+[7]+[1]+[1432]=[1444]

;; переводим значения R0, R1, R2 в кодировку ACSII:

lbDrawElement:

add A, #30h

call prWaitFor\_TI

mov SBUF, A

ret

prDisplay:

call prWaitFor\_TI

mov SBUF, #CR; переводим курсор в начало строки:

mov A, R3

call lbDrawElement

mov A, R2

call lbDrawElement

mov A, R1

call lbDrawElement

mov A, R0

call lbDrawElement

call prWaitFor\_TI; для выравнивания подождём передачи последнего символа

setb TI; [1] иначе - программа "зависнет"

ret; [2]

;; процедура prWaitFor\_TI:

;; ожидание "готовности передачи" по последовательному порту

prWaitFor\_TI:

jnb TI, $; зациклить, пока TI=0

clr TI ; устанавливаем "неготовность пердачи"

ret

;; процедура prIs\_SetTime\_Mode:

;; [2 на вызов]+[2+1+2+2]=[9]

;; проверяет, нажата ли клавиша входа в режим установки времени;

;; такой клавишей явл. '0'

prIs\_SetTime\_Mode:

mov P1, #00BFh

mov A, P1

cjne A, #00BEh, lbNot\_SetTime\_Mode

call prSetTime; переход в режим установки таймера

lbNot\_SetTime\_Mode:

cjne A, #00B7h, lbNotAnyMode

mov R0, Buf\_R0

mov R1, Buf\_R1

mov R2, Buf\_R2

mov R3, Buf\_R3

lbNotAnyMode: ret

;; процедура prExam\_NumKeys:

;; опрос цифровых клавиш

prExam\_NumKeys:

mov R7, #0FFh; допустим, что ничего не будет нажато

;опрос первого столбца клавиш:

lbKey\_1:

mov P1, #00DFh

mov A, P1

cjne A, #00DDh, lbKey\_2

mov R7, #1

ret

lbKey\_2:

cjne A, #00DEh, lbKey\_3

mov R7, #2

ret

lbKey\_3:

cjne A, #00D7h, lbKey\_4

mov R7, #3

ret

;опрос второго столбца клавиш:

lbKey\_4:

mov P1, #00EFh

mov A, P1

cjne A, #00EDh, lbKey\_5

mov R7, #4

ret

lbKey\_5:

cjne A, #00EEh, lbKey\_6

mov R7, #5

ret

lbKey\_6:

cjne A, #00E7h, lbKey\_7

mov R7, #6

ret

;опрос третьего столбца клавиш:

lbKey\_7:

mov P1, #007Fh

mov A, P1

cjne A, #007Dh, lbKey\_8

mov R7, #7

ret

lbKey\_8:

cjne A, #007Eh, lbKey\_9

mov R7, #8

ret

lbKey\_9:

cjne A, #0077h, lbKey\_0

mov R7, #9

ret

;опрос четвёртого столбца клавиш:

lbKey\_0:

mov P1, #00BFh

mov A, P1

cjne A, #00BDh, lbEnd\_Exam\_NumKeys

mov R7, #0

lbEnd\_Exam\_NumKeys: ret

;; процедура prWaitFor\_NextKey\_Pressed:

;; ждёт СЛЕДУЮЩЕГО НАЖАТИЯ, чтобы долгое нажатие не "флудило"

prWaitFor\_NextKey\_Pressed:

lbPrevKey\_Pressed: call prSmartDelay; чтоб избежать "дребезга" клавиатуры

call prExam\_NumKeys;

cjne R7, #0FFh, lbPrevKey\_Pressed;

call prSmartDelay; чтоб избежать "дребезга" клавиатуры

lbNo\_NextKey\_Pressed: call prExam\_NumKeys

cjne R7, #0FFh, lbNextKey\_Pressed

jmp lbNo\_NextKey\_Pressed

lbNextKey\_Pressed: ret

;; процедура prSetTime:

;; режим установки таймера:

prSetTime:

clr EA

mov DPTR, #msgSetTime\_Mode

call prShowMessage; вывод сообщения msgSetTime\_Mode

call prDisplay

;регистр R3:

call prWaitFor\_NextKey\_Pressed;

mov R3, 07

call prDisplay

;регистр R2:

call prWaitFor\_NextKey\_Pressed

mov R2, 07

call prDisplay

;регистр R1:

call prWaitFor\_NextKey\_Pressed

mov R1, 07

call prDisplay

;регистр R0:

call prWaitFor\_NextKey\_Pressed

mov R0, 07

call prDisplay

mov Buf\_R0, R0

mov Buf\_R1, R1

mov Buf\_R2, R2

mov Buf\_R3, R3

mov DPTR, #msgShowTime\_Mode

call prShowMessage; вывод сообщения msgShowTime\_Mode

setb EA

ret

;; процедуры prIs\_R0\_more\_59, prIs\_R1\_more\_59, prIs\_R2\_more\_23:

;; проверяют соответствующие регистры на корректность значений в них;

;; большие значения уменьшаются до ближайших верных

prCorrect\_R0:

cjne R0, #59, lbR0\_not\_59

ret

lbR0\_not\_59: jnc lbR0\_more\_59

ret

lbR0\_more\_59: mov R0, #59

ret

prCorrect\_R1:

cjne R1, #59, lbR1\_not\_59

ret

lbR1\_not\_59: jnc lbR1\_more\_59

ret

lbR1\_more\_59: mov R1, #59

ret

prCorrect\_R2:

cjne R2, #23, lbR2\_not\_23

ret

lbR2\_not\_23: jnc lbR2\_more\_23

ret

lbR2\_more\_23: mov R2, #23

ret

;; процедура prShowMessage:

;; выводит на экран текстовое сообщение; символ ESC - признак конца сообщения

prShowMessage:

clr A

movc A, @A + DPTR

lbNextSymbol: call prWaitFor\_TI

mov SBUF, A

inc DPTR

clr A

movc A, @A + DPTR

cjne A, #ESC, lbNextSymbol

ret

;;;; ОПИСАНИЕ ТЕКСТОВЫХ СООБЩЕНИЙ: ;;;;

msgSetTime\_Mode: db LF,CR, '[Regim ustanovki taymera:]', CR, LF, ESC

msgShowTime\_Mode: db LF,CR, 'OBRATNYI OTSCHET...', CR, LF, ESC

msgFinal: db CR, '!!!OBRATNYI OTSCHET ZAKONCHEN!!!:', CR, LF, ESC

END;;;; КОНЕЦ ПРОГРАММЫ