```
;   ==
;   KURSACH
;   PAMYATI LIL PIPA
;   YA NAKODIL ETO DLYA TEBYA BRAT
;   ==

;   PB7, PB6, PB5, PB4 - выходы, управляющие драйвером
;   PD0 - RXD0, PD1 - TXD0, PD2 - FLOW CONTROL
;   PA0 - ADC0

.include "m164Adef.inc"

.def tempL          = R16
.def tempH          = R17
.def rxBytes        = R18   ; Счетчик принятых байт
.def txBytes        = R19   ; Счетчик переданных байт
.def rxChecksum     = R20   ; Контрольная сумма принятых байт
.def flagReg        = R21   ; Регистр флагов
.def ConvCount      = R22   ; Счетчик измерений аналого-цифрового преобразователя
.def ADC1           = R23   ; Сумма значений АЦП
.def ADC2           = R24

.equ FLAG_RECEIVE   = 0     ; Флаг принятого запроса
.equ FLAG_TRANSMIT  = 1     ; Флаг завершения передачи
.equ FLAG_ADC_READY = 2     ; Флаг завершения замера АЦП
.equ VAL_TX         = 5     ; Количество байт для обмена
.equ VAL_RX         = 3
.equ VAL_TIMEOUT    = 35    ; Таймаут (35 переполнений - 1 секунда)
.equ VAL_CONVERTER  = 8     ; Количество измерений аналого-цифрового преобразователя
.equ VAL_CONV_SHIFT = 3     ; Сдвиг для среднего арифметического результатов (квадратный корень
 от VAL_CONVERTER)
.equ PIN_FLOW_CTRL  = 3     ; Пин flow control
.equ MCU_ADDRESS    = 1     ; Адрес микроконтроллера

.DSEG

varBuf_Rx:  .BYTE 8         ; Буфер приема
varBuf_Tx:  .BYTE 8         ; Буфер передачи

.CSEG

; Reset
.org $0000
    rjmp Init

; Output Compare1A Interrupt Vector Address
.org $001A
    rjmp  RX_Timeout

; UART Receive Complete Interrupt Vector Address
.org $0028
    rjmp  RX_complete

; UART Data Register Empty Interrupt Vector Address
.org $002A
    rjmp  UDR_empty

; UART Transmit Complete Interrupt Vector Address
.org $002C
    rjmp  TX_complete

; Окончание считывания АЦП
```

```
.org $0030
    rjmp ADC_complete

Init:
; Инициализация стека (выбор вершины)
    ldi     tempL, LOW(RAMEND)
    out     SPL, tempL
    ldi     tempL, HIGH(RAMEND)
    out     SPH, tempL

; Инициализация портов ввода/вывода B
    ; PB4-PB7 - выходы,
    ldi     tempL, 0b11110000
    out     DDRB, tempL
    ; Отключена подтяжка на всех выходах
    ldi     tempL, 0b00000000
    out     PORTB, tempL

; Инициализация портов ввода/вывода D
    ; PD1, PD2 - выходы (TXD0, FLOW CONTROL), PD0 - вход (RXD0)
    ldi     tempL, 0b00000110
    out     DDRD, tempL
    ; Вкл подтяжка на PD0 (RXD0)
    ldi     tempL, 0b00000001
    out     PORTD, tempL;
    ; Включение приема
    sbi     PORTD, PIN_FLOW_CTRL

; Инициализация портов ввода/вывода A
    ldi     tempL, 0b11111111           ; ?!
    out     DDRC, tempL

; Инициализация UART
    ; Разрешение прерывания по завершению приема
    ; Установка режима межпроцессорного обмена
    ldi     tempL, (1<<MPCM0)
    sts     UCSR0A, tempL
    ; Установка режима приема данных
    ldi     tempL, (1<<RXCIE0)|(1<<UDRIE0)|(1<<RXEN0)|(1<<MPCM0)
    sts     UCSR0B, tempL
    ; Асинхронный режим, бит четности, 8 бит информации
    ldi     tempL, (1<<UPM00)|(1<<UCSZ00)|(1<<UCSZ10)
    sts     UCSR0C, tempL
    ; Частота тактирования - 8 МГц
    ; Частота обмена - 19200 бод, одинарная скорость
    ldi     tempL, 25
    ldi     tempH, 00
    sts     UBRR0L, tempL;
    sts     UBRR0H, tempH

; Инициализация  таймера TCNT1
    ; Выходы OCnA и OCnB отключены
    ldi     tempL, 0
    sts     TCCR1A, tempL;
    ; Нет деления частоты, верхний предел счета - OCR1A
    ldi     tempL, (1<<WGM12)
    sts     TCCR1B, tempL
    ; Верхний предел счета = 0,006*8000000=48000(0xBB80)
    ldi     tempH, 0xBB
    ldi     tempL, 0x80
    sts     OCR1AH, tempH
    sts     OCR1AL, tempL
```

```
; Инициализация аналого-цифрового конвертера
    ; Вход - ADC0, AREF включен
    ldi     tempL, 0
    sts     ADMUX, tempL

; Обнуление участка буфера Rx в SRAM
    ldi     YL, LOW(varBuf_Rx)
    ldi     YH, HIGH(varBuf_Rx)
    ldi     tempL, VAL_RX
BufRxdNull:
    st      Y+, tempL
    dec     tempL
    cpi     tempL, 0x00
    brne    BufRxdNull

; Обнуление участка буфера Tx в SRAM
    ldi     YL, LOW(varBuf_Tx)
    ldi     YH, HIGH(varBuf_Tx)
    ldi     tempL, VAL_TX
BufTxdNull:
    st      Y+, tempL
    dec     tempL
    cpi     tempL, 0x00
    brne    BufTxdNull

; Обнуление переменных
    clr     tempL
    clr     tempH
    clr     rxBytes
    clr     txBytes
    clr     rxChecksum
    clr     flagReg

; Разрешаем прерывания
    sei

;=================================================
; Начало цикла программы
;=================================================
Start:
    ; Ожидание приема запроса
    sbrc    flagReg, FLAG_RECEIVE
    rjmp    Received
    rjmp    Start
; Запрос принят
Received:
    ; Загрузка адреса устройства из буфера приема
    ld      tempL, Y+
    ; Загружаем информационный байт из буфера приема
    ld      tempL, Y+
    ; Очищаем все кроме четырех младших битов
    andi    tempL, 0x0F
    ; Выводим на порт
    out     PORTB, tempL
    ; Включение АЦП, разрешение прерывания по окончанию считывания, начало замера
    ldi     tempL, (1<<ADEN)|(1<<ADIE)|(1<<ADSC)
    sts     ADCSRA, tempL
; Ожидание окончания замеров
Wait_ADC:
    sbrc    flagReg, FLAG_ADC_READY
    rjmp    Transmit
    rjmp    Wait_ADC
```

```
; Формирование посылки для отправления
Transmit:
    ; Загрузка адреса начала буфера отправки в адресный регистр
    ldi     YL, LOW(varBuf_Tx)
    ldi     YH, HIGH(varBuf_Tx)
    ; Запись адреса устройства в первый байт
    ldi     tempL, MCU_ADDRESS
    st      Y+, tempL
    ; Запись состояния портов ввода/вывода во второй байт
    in      tempL, PORTB
    st      Y+, tempL
    ; Запись результата измерения АЦП в третий и четвертый байты
    st      Y+, ADC2
    st      Y+, ADC1
    ; Вычисление контрольной суммы
    ldi     convCount, VAL_TX
    clr     tempH
Checksum_calc:
    ld      tempL, -Y
    add     tempH, tempL
    dec     convCount
    cpi     convCount, 1
    breq    Checksum_end
    rjmp    Checksum_calc
; Добавление контрольной суммы в последний байт
Checksum_end:
    com     tempH
    sts     varBuf_Tx + VAL_TX - 1, tempH
    ; Flow control на передачу
    cbi     PORTD, PIN_FLOW_CTRL
    ; Заполнение регистра данных USART
    ld      tempL, Y+
    sts     UDR0, tempL
    ; Разрешение прерывания по окончанию приема и опустошению регистра данных
    ldi    tempL, (1<<TXEN1)|(1<<UDRIE1)
    sts    UCSR0B, tempL
Wait_transmit:
    sbrc    flagReg, FLAG_TRANSMIT
    rjmp    End
    rjmp    Wait_transmit
End:
    clr     flagReg
    rjmp    Start
;=================================================
; Конец цикла программы
;=================================================

; Прерывание окончания приема по USART
RX_complete:
    ; Сохранение регистров в стэк
    push    tempL
    push    tempH
    in      tempL, SREG
    push    tempL
; Загрузка принятой информации из регистра данных
RX_UDR_in:
    lds     tempL, UDR1
; Проверка статусного регистра USART на наличие ошибок
RX_check:
    lds     tempH, UCSR1A
    andi    tempH, (1<<FE1)|(1<<DOR1)|(1<<UPE1)
    breq    RX_no_error
```

```
    rjmp    RX_regs_exit
; При отсутствии ошибок приема
RX_no_error:
    st      Y+, tempL
    inc     rxBytes
    cpi     rxBytes, 1
    breq    RX_first_byte
    cpi     tempH, VAL_RX
    breq    RX_checksum
    add     rxChecksum, tempL
    rjmp    RX_regs_exit
; При приеме первого байта
RX_first_byte:
    ; Проверка адреса
    cpi     tempL, MCU_ADDRESS
    brne    RX_regs_exit
    ; Сброс бита межпроцессорного обмена
    ldi     tempL, 0
    sts     UCSR0A, tempL
    ; Добавление значения байта в контрольную сумму
    add     rxChecksum, tempL
    lds     tempL, TIMSK1
    ; Сброс таймера
    clr     tempH
    sts     TCNT1H,tempH
    sts     TCNT1L,tempH
    ; Разрешить прерывание по каналу A
    bld     tempL, OCIE1A
    sts     TIMSK1, tempL
    rjmp    RX_regs_exit
; Проверка контрольной суммы
RX_checksum:
    ; Инверсия последнего принятого байта
    com     tempL
    ; Сравнение контрольных сумм
    cp      tempL, rxChecksum
    brne    RX_end
    ; Установка флага успешного приема данных
    ori     flagReg, (1<<FLAG_RECEIVE)
; Окончание приема данных
RX_end:
    ; Запрет прерывания таймера по переполнению канала A
    lds     tempL, TIMSK1
    andi    tempL, ~(1<<OCIE1A)
    sts     TIMSK1, tempL
    clr     rxBytes
    clr     rxChecksum
    ; Установка указателя на начало буфера отправки
    ldi     YL, LOW(varBuf_Tx)
    ldi     YH, HIGH(varBuf_Tx)
; Загрузка регистров из стэка
RX_regs_exit:
    pop     tempL
    out     SREG, tempL
    pop     tempH
    pop     tempL
    reti

; Прерывание по опустошению регистра данных
UDR_empty:
    ; Сохранение регистров в стэк
    push    tempL
```

```
    in      tempL, SREG
    push    tempL
    ; Разрешение прерывания по завершению приема
    ldi     tempL, (1<<TXEN1)|(1<<TXCIE1)
    sts     UCSR0B, tempL
    ; Загрузка адреса начала буфера передачи в адресный регистр
    ldi     YL, low(varBuf_Tx)
    ldi     YH, high(varBuf_Tx)
    ld      tempL, Y+
    sts     UDR0, tempL
    ; Загрузка регистров из стэка
    pop     tempL
    out     SREG, tempL
    pop     tempL
    reti

; Прерывание по завершению приема
TX_complete:
    ; Сохранение регистров в стэк
    push    tempL
    in      tempL, SREG
    push    tempL
    push    tempH
    ; Инкремент счетчика отправленных байтов
    inc     txBytes
    ; Проверка на последний байт
    cpi     txBytes, VAL_TX
    breq    TX_end
    ; Отправление данных
    ld      tempL, Y+
    sts     UDR1, tempL
    rjmp    TX_regs_exit
; Завершение отправки
TX_end:
    clr     txBytes
    ; Установка режима межпроцессорного обмена
    ldi     tempL, (1<<MPCM0)
    sts     UCSR0A, tempL
    ; Разрешение прерывания по завершению приема
    ; Установка режима приема данных
    ldi     tempL, (1<<RXCIE1)|(1<<RXEN1)
    sts     UCSR0B, tempL
    ; Загрузка адреса начала буфера приема в адресный регистр
    ldi     YL, LOW(varBuf_Rx)
    ldi     YH, HIGH(varBuf_Rx)
    ori     flagReg, (1<<FLAG_TRANSMIT)
    ; Flow control на прием
    sbi     PORTD, PIN_FLOW_CTRL
; Загрузка регистров из стэка
TX_regs_exit:
    pop     tempH
    pop     tempL
    out     SREG, tempL
    pop     tempL
    reti

; Прерывание таймера по совпадению канала A
RX_timeout:
    ; Сохранение регистров в стэк
    push    tempL
    push    tempH
    in      tempL, SREG
```

```
    push    tempL
    ; Запрет прерывания по совпадению канала A
    lds     tempL, TIMSK1
    andi    tempL, ~(1<<OCIE1A)
    sts     TIMSK1, tempL
    ; Обнуление регистров
    clr     rxBytes
    clr     rxChecksum
    ; Установка адресного регистра на адрес начала буфера приема
    ldi     YL, LOW(varBuf_Rx)
    ldi     YH, HIGH(varBuf_Rx)
    ; Загрузка регистров из стэка
    pop     tempL
    out     SREG, tempL
    pop     tempH
    pop     tempL
    reti

; Прерывание по завершению измерения АЦП
ADC_complete:
    ; Сохранение регистров в стэк
    push    tempL
    push    tempH
    in      tempL, SREG
    push    tempL
    ; Получение результатов измерения
    lds     tempL, ADCL
    lds     tempH, ADCH
    ; Сложение результатов с предыдущими результатами
    add     ADC1, tempL
    adc     ADC2, tempH
    ; Прибавление счетчика
    inc     ConvCount
    ; Достигнуто ли максимальное количество измерений
    cpi     ConvCount, VAL_CONVERTER
    breq    ADC_max_measures
    ; Начало следующего замера
    ldi     tempL, (1<<ADEN)|(1<<ADIE)|(1<<ADSC)
    sts     ADCSRA, tempL
; Загрузка регистров из стэка
ADC_regs_exit:
    pop     tempH
    pop     tempL
    out     SREG, tempL
    pop     tempL
    reti
; Достигнуто максимальное количество измерений
ADC_max_measures:
    ldi     ConvCount, VAL_CONV_SHIFT
    clr     tempL
; Вычисление среднего арифметического (деление на степень двойки сдвигом)
ADC_middle:
    lsr     ADC2
    brcc    ADC_mid_skip
    ldi     tempL, 0b10000000
ADC_mid_skip:
    lsr     ADC1
    or      ADC1, tempL
    dec     ConvCount
    cpi     ConvCount, 0
    breq    ADC_end
    rjmp    ADC_middle
```

```
ADC_end:
    ; Установка флага окончания вычисления значения АЦП
    ori     flagReg, (1<<FLAG_ADC_READY)
    rjmp    ADC_regs_exit
```