```
;*****************************************************************************
;* U S B   S T A C K   F O R   T H E   A V R   F A M I L Y
;*
;* File Name            :"USB90S2313.asm"
;* Title                :USB stack + Infrared remote control to nonUSB MCU
;* Date                 :5.4.2003
;* Version              :1.6
;* Target MCU           :AT90S2313-10,AT90S2323-10,AT90S2343-10
;* AUTHOR               :Ing. Igor Cesko
;*                       Slovakia
;*                       cesko@internet.sk
;*                       http://www.cesko.host.sk
;*
;* DESCRIPTION:
;*  USB protocol implementation into MCU with noUSB interface:
;*  Device:
;*  Infrared remote control with TSOP17xx/SFH511x sensor and USB connection
;*  + Universal USB interface (8-bit I/O port + RS232 serial line + EEPROM)
;*
;* The timing is adapted for 12 MHz crystal (overclocked MCU !!!)
;*
;* Copyright (c) Ing. Igor Cesko 2002-2003
;* All rights reserved
;*****************************************************************************
.include "2313def.inc"

.equ    inputport               =PINB
.equ    outputport              =PORTB
.equ    USBdirection            =DDRB
.equ    DATAplus                =1              ;signal D+ na PB1
.equ    DATAminus               =0              ;signal D- na PB0 - treba dat na tento pin pull-up 1.5kOhm
.equ    USBpinmask              =0b11111100     ;mask low 2 bits (D+,D-) on PB
.equ    USBpinmaskDplus         =~(1<<DATAplus) ;mask D+ bit on PB
.equ    USBpinmaskDminus        =~(1<<DATAminus);mask D- bit on PB

.equ    TSOPPort                =PINB
.equ    TSOPpullupPort          =PORTB
.equ    TSOPPin                 =2              ;signal OUT z IR senzora TSOP1738 na PB2

.equ    LEDPortLSB              =PORTD          ;pripojenie LED diod LSB
.equ    LEDPinLSB               =PIND           ;pripojenie LED diod LSB (vstup)
.equ    LEDdirectionLSB         =DDRD           ;vstup/vystup LED LSB
.equ    LEDPortMSB              =PORTB          ;pripojenie LED diod MSB
.equ    LEDPinMSB               =PINB           ;pripojenie LED diod MSB  (vstup)
.equ    LEDdirectionMSB         =DDRB           ;vstup/vystup LED MSB
.equ    LEDlsb0                 =3              ;LED0 na pin PD3
.equ    LEDlsb1                 =5              ;LED1 na pin PD5
.equ    LEDlsb2                 =6              ;LED2 na pin PD6
.equ    LEDmsb3                 =3              ;LED3 na pin PB3
.equ    LEDmsb4                 =4              ;LED4 na pin PB4
.equ    LEDmsb5                 =5              ;LED5 na pin PB5
.equ    LEDmsb6                 =6              ;LED6 na pin PB6
.equ    LEDmsb7                 =7              ;LED7 na pin PB7

.equ    SOPbyte                 =0b10000000     ;Start of Packet byte
.equ    DATA0PID                =0b11000011     ;PID pre DATA0 pole
.equ    DATA1PID                =0b01001011     ;PID pre DATA1 pole
.equ    OUTPID                  =0b11100001     ;PID pre OUT pole
.equ    INPID                   =0b01101001     ;PID pre IN pole
.equ    SOFPID                  =0b10100101     ;PID pre SOF pole
.equ    SETUPPID                =0b00101101     ;PID pre SETUP pole
.equ    ACKPID                  =0b11010010     ;PID pre ACK pole
.equ    NAKPID                  =0b01011010     ;PID pre NAK pole
.equ    STALLPID                =0b00011110     ;PID pre STALL pole
.equ    PREPID                  =0b00111100     ;PID pre PRE pole

.equ    nSOPbyte                =0b00000001     ;Start of Packet byte - opacne poradie
.equ    nDATA0PID               =0b11000011     ;PID pre DATA0 pole - opacne poradie
.equ    nDATA1PID               =0b11010010     ;PID pre DATA1 pole - opacne poradie
.equ    nOUTPID                 =0b10000111     ;PID pre OUT pole - opacne poradie
.equ    nINPID                  =0b10010110     ;PID pre IN pole - opacne poradie
.equ    nSOFPID                 =0b10100101     ;PID pre SOF pole - opacne poradie
.equ    nSETUPPID               =0b10110100     ;PID pre SETUP pole - opacne poradie
.equ    nACKPID                 =0b01001011     ;PID pre ACK pole - opacne poradie
.equ    nNAKPID                 =0b01011010     ;PID pre NAK pole - opacne poradie
.equ    nSTALLPID               =0b01111000     ;PID pre STALL pole - opacne poradie
.equ    nPREPID                 =0b00111100     ;PID pre PRE pole - opacne poradie

.equ    nNRZITokenPID           =~0b10000000    ;PID maska pre Token paket (IN,OUT,SOF,SETUP) - opacne poradie NRZI
.equ    nNRZISOPbyte            =~0b10101011    ;Start of Packet byte - opacne poradie NRZI
.equ    nNRZIDATA0PID           =~0b11010111    ;PID pre DATA0 pole - opacne poradie NRZI
.equ    nNRZIDATA1PID           =~0b11001001    ;PID pre DATA1 pole - opacne poradie NRZI
.equ    nNRZIOUTPID             =~0b10101111    ;PID pre OUT pole - opacne poradie NRZI
.equ    nNRZIINPID              =~0b10110001    ;PID pre IN pole - opacne poradie NRZI
.equ    nNRZISOFPID             =~0b10010011    ;PID pre SOF pole - opacne poradie NRZI
.equ    nNRZISETUPPID           =~0b10001101    ;PID pre SETUP pole - opacne poradie NRZI
.equ    nNRZIACKPID             =~0b00100111    ;PID pre ACK pole - opacne poradie NRZI
.equ    nNRZINAKPID             =~0b00111001    ;PID pre NAK pole - opacne poradie NRZI
.equ    nNRZISTALLPID           =~0b00000111    ;PID pre STALL pole - opacne poradie NRZI
.equ    nNRZIPREPID             =~0b01111101    ;PID pre PRE pole - opacne poradie NRZI
.equ    nNRZIADDR0              =~0b01010101    ;Adresa = 0 - opacne poradie NRZI

                                                ;stavove byty - State
.equ    BaseState               =0              ;
.equ    SetupState              =1              ;
.equ    InState                 =2              ;
.equ    OutState                =3              ;
.equ    SOFState                =4              ;
.equ    DataState               =5              ;

                                                ;Flagy pozadovanej akcie
.equ    DoNone                                  =0
.equ    DoReceiveOutData                        =1
.equ    DoReceiveSetupData                      =2
.equ    DoPrepareOutContinuousBuffer            =3
.equ    DoReadySendAnswer                       =4


.equ    CRC5poly                =0b00101                ;CRC5 polynom
.equ    CRC5zvysok              =0b01100                ;CRC5 zvysok po uspesnpm CRC5
.equ    CRC16poly               =0b1000000000000101     ;CRC16 polynom
.equ    CRC16zvysok             =0b1000000000001101     ;CRC16 zvysok po uspesnom CRC16

.equ    MAXUSBBYTES             =14                     ;maximum bytes in USB input message
.equ    MAXINFRALENGTH          =36                     ;maximalna dlzka Infra kodu (pocet jednotiek a nul spolu) (pozor: MAXINFRALENGTH musi
  byt parne cislo !!!)
.equ    NumberOfFirstBits       =10                     ;kolko prvych bitov moze byt dlhsich
.equ    NoFirstBitsTimerOffset  =256-12800*12/1024      ;Timeout 12.8ms (12800us) na ukoncenie prijmu po uvodnych bitoch (12Mhz:clock,
  1024:timer predivider, 256:timer overflow value)
.equ    BaudRate                =12000000/16/57600-1    ;nastavit vysielaciu rychlost UART-u na 57600 (pre 12MHz=12000000Hz)

.equ    InputBufferBegin        =RAMEND-127                             ;zaciatok prijimacieho shift buffera
.equ    InputShiftBufferBegin   =InputBufferBegin+MAXUSBBYTES           ;zaciatok prijimacieho buffera
.equ    InfraBufferBegin        =InputShiftBufferBegin+MAXUSBBYTES      ;zaciatok buffera pre Infra prijem

.equ    OutputBufferBegin       =RAMEND-MAXUSBBYTES-2   ;zaciatok vysielacieho buffera
.equ    AckBufferBegin          =OutputBufferBegin-3    ;zaciatok vysielacieho buffera Ack
.equ    NakBufferBegin          =AckBufferBegin-3       ;zaciatok vysielacieho buffera Nak

```

```
128 .equ    StackBegin              =NakBufferBegin-1       ;spodok zasobnika
129
130 .def    ConfigByte              =R1             ;0=unconfigured state
131 .def    backupbitcount          =R2             ;zaloha bitcount registra v INT0 preruseni
132 .def    RAMread                 =R3             ;ci sa ma citat zo SRAM-ky
133 .def    backupSREGTimer         =R4             ;zaloha Flag registra v Timer interrupte
134 .def    backupSREG              =R5             ;zaloha Flag registra v INT0 preruseni
135 .def    ACC                     =R6             ;accumulator
136 .def    lastBitstufNumber       =R7             ;pozicia bitstuffingu
137 .def    OutBitStuffNumber       =R8             ;kolko bitov sa ma este odvysielat z posledneho bytu - bitstuffing
138 .def    BitStuffInOut           =R9             ;ci sa ma vkladat alebo mazat bitstuffing
139 .def    TotalBytesToSend        =R10            ;kolko sa ma poslat bytov
140 .def    TransmitPart            =R11            ;poradove cislo vysielacej casti
141 .def    InputBufferLength       =R12            ;dlzka pripravena vo vstupnom USB bufferi
142 .def    OutputBufferLength      =R13            ;dlzka odpovede pripravena v USB bufferi
143 .def    MyUpdatedAddress        =R14            ;moja USB adresa na update
144 .def    MyAddress               =R15            ;moja USB adresa
145
146
147 .def    ActionFlag              =R16            ;co sa ma urobit v hlavnej slucke programu
148 .def    temp3                   =R17            ;temporary register
149 .def    temp2                   =R18            ;temporary register
150 .def    temp1                   =R19            ;temporary register
151 .def    temp0                   =R20            ;temporary register
152 .def    bitcount                =R21            ;counter of bits in byte
153 .def    ByteCount               =R22            ;pocitadlo maximalneho poctu prijatych bajtov
154 .def    inputbuf                =R23            ;prijimaci register
155 .def    shiftbuf                =R24            ;posuvny prijimaci register
156 .def    State                   =R25            ;byte stavu stavoveho stroja
157 .def    InfraBufptrX            =R26            ;XL register - pointer do buffera prijatych IR kodov
158 .def    InfraBufferFull         =R27            ;XH register - priznak plneho Infra Buffera
159 .def    USBBufptrY              =R28            ;YL register - pointer do USB buffera input/output
160 .def    ROMBufptrZ              =R30            ;ZL register - pointer do buffera ROM dat
161
162 ;poziadavky na deskriptory
163 .equ    GET_STATUS              =0
164 .equ    CLEAR_FEATURE           =1
165 .equ    SET_FEATURE             =3
166 .equ    SET_ADDRESS             =5
167 .equ    GET_DESCRIPTOR          =6
168 .equ    SET_DESCRIPTOR          =7
169 .equ    GET_CONFIGURATION       =8
170 .equ    SET_CONFIGURATION       =9
171 .equ    GET_INTERFACE           =10
172 .equ    SET_INTERFACE           =11
173 .equ    SYNCH_FRAME             =12
174
175 ;typy deskriptorov
176 .equ    DEVICE                  =1
177 .equ    CONFIGURATION           =2
178 .equ    STRING                  =3
179 .equ    INTERFACE               =4
180 .equ    ENDPOINT                =5
181
182
183 ;-----------------------------------------------------------------------------------------
184 ;*****************************************************************
185 ;* Interrupt table
186 ;*****************************************************************
187 .cseg
188 ;-----------------------------------------------------------------------------------------
189 .org 0                                      ;po resete
190                 rjmp    reset
191 ;-----------------------------------------------------------------------------------------
192 .org INT0addr                               ;externe prerusenie INT0
193                 rjmp    INT0handler
194 ;-----------------------------------------------------------------------------------------
195 .org INT1addr                               ;externe prerusenie INT1 (pre AT89S2323 je tu Timer0 prerusenie)
196                 sei                         ;povol interrupty na obsluhu USB
197                 in      backupSREGTimer,SREG
198                 rjmp    OVF0handler2323     ;na kompatibilitu s AT89S2323 sa skoci na pretecenie casovaca T0
199 ;-----------------------------------------------------------------------------------------
200 ;*****************************************************************
201 ;* Timer0 interrupt handler
202 ;*****************************************************************
203 .org OVF0addr                              ;citac/casovac 0
204 OVF0handler:
205                 sei                         ;povol interrupty na obsluhu USB
206                 in      backupSREGTimer,SREG
207 OVF0handler2323:
208                 push    temp0
209                 ldi     temp0,0<<TOIE0      ;zakazat interrupt od casovaca
210                 out     TIMSK,temp0
211                 ldi     temp0,5             ;nastav clock input na CLK/1024 = tick kazdych 0.0853ms
212                 out     TCCR0,temp0
213                 push    temp1
214                 push    temp2
215                 push    temp3
216
217                 clr     temp2               ;vynulovanie offsetu v bufferi (a flagu pretecenia)
218                 ldi     temp3,1             ;timeout - plnenie citaca (pre prve bity iny ako pre dalsie bity: aby sa nikdy nevyslala dlzka
    kodu 255, tak sa musi plnit aspon na 1)
219 StartTSOPsampling:
220                 ldi     InfraBufptrX,InfraBufferBegin+3 ;nastavenie sa na zaciatok buffera Infra kodu + 3byty hlavicky (dlzka kodu + counter +
    offset)
221                 clr     temp1               ;vynulovanie dlzky Infra kodu
222 WaitForTOSOP1:
223                 sbis    TSOPPort,TSOPPin    ;ak je TSOPPin v nule
224                 rjmp    CheckWaitForTOSOP1  ;tak cakaj na jednotku
225                 rjmp    StoreToInfraBuffer
226 WaitForTOSOP0:
227                 sbic    TSOPPort,TSOPPin    ;ak je TSOPPin v jednotke
228                 rjmp    CheckWaitForTOSOP0  ;tak cakaj na nulu
229 StoreToInfraBuffer:
230                 in      temp0,TCNT0         ;odchyt citac
231                 sub     temp0,temp3         ;koriguj odchyteny citac vzhladom na jeho plnenie
232                 cpi     temp1,NumberOfFirstBits ;ak este neboli prve bity
233                 brne    WasNotFirstBits     ;tak nechaj plnenie citaca na nule
234                 ldi     temp3,NoFirstBitsTimerOffset    ;inak zmen plnenie citaca
235 WasNotFirstBits:
236                 out     TCNT0,temp3         ;nastav citac aby bol ako casovac
237                 out     UDR,temp0           ;vysli data na UART
238                 cpi     InfraBufferFull,0   ;ak je infra buffer plny
239                 brne    NoStoreToInfraBuffer    ;tak neukladaj data
240                 sbrs    RAMread,0           ;ak RAMread=1 cita sa z RAM - potom neukladaj data
241                 st      X+,temp0            ;a uloz ho do buffera
242 NoStoreToInfraBuffer:
243                 inc     temp1               ;zvys dlzku Infra Buffera
244                 cpi     temp1,MAXINFRALENGTH    ;ak sa nedosiahol maximum Infra buffera
245                 brne    NoTSOPoverflow      ;tak pokracuj
246                 inc     temp2               ;inak si poznac ze pretiekol
247                 cpi     temp2,3             ;ak boli uz 3 pretecenia
248                 dec     temp1               ;tak uber posledny bit
249                 breq    EndInfraSampling    ;a skonci so samplovanim
250                 rjmp    StartTSOPsampling   ;a zacni odznova (pozor: MAXINFRALENGTH musi byt parne cislo !!!)
251 NoTSOPoverflow:
252                 sbic    TSOPPort,TSOPPin    ;ak je TSOPPin v jednotke
253                 rjmp    WaitForTOSOP0       ;tak cakaj na nulu
254                                             ;inak cakaj na jednotku
```

```
255 CheckWaitForTOSOP1:
256                 in      temp0,TIFR              ;odchyt flag pretecenia citaca
257                 sbrc    temp0,TOV0              ;ak pretiekol
258                 rjmp    EndInfraSampling        ;tak skonci so samplovanim
259                 rjmp    WaitForTOSOP1           ;inak cakaj na zmenu stavu
260 CheckWaitForTOSOP0:
261                 in      temp0,TIFR              ;odchyt flag pretecenia citaca
262                 sbrc    temp0,TOV0              ;ak pretiekol
263                 rjmp    EndInfraSampling        ;tak skonci so samplovanim
264                 rjmp    WaitForTOSOP0           ;inak cakaj na zmenu stavu
265 EndInfraSampling:
266                 sbrs    InfraBufferFull,0       ;ak je infra buffer plny - tak neukladaj hlavicku
267                 sbrc    RAMread,0               ;ak RAMread=1 cita sa z RAM
268                 rjmp    EndTSOPSampling         ;potom neukladaj hlavicku
269                 cpi     temp2,0                 ;ak nebolo nejake pretecenie
270                 breq    NoOverflow              ;tak nic nerob
271                 mov     temp2,temp1             ;inak zaznamenaj offset
272                 ldi     temp1,MAXINFRALENGTH    ;a maximalnu dlzku buffera
273 NoOverflow:
274                 sts     InfraBufferBegin,temp1  ;dlzku kodu na zaciatok Infra buffera
275                 cpi     temp1,3                 ;zisti ci je dlzka kodu aspon 3
276                 lds     temp1,InfraBufferBegin+1;nacitaj pocitadlo infra kodov
277                 brcs    NoIncrementCodeCounter  ;ak je dlzka kodu vacsia alebo rovna 3
278                 inc     temp1                   ;tak zvys pocitadlo infra kodov
279                 ldi     InfraBufferFull,1       ;a nastav plnost buffera
280 NoIncrementCodeCounter:
281                 sts     InfraBufferBegin+1,temp1;uloz nove pocitadlo infra kodov
282                 sts     InfraBufferBegin+2,temp2;uloz offset zaciatku buffera
283 EndTSOPSampling:
284                 pop     temp3
285                 pop     temp2
286                 pop     temp1
287                 ldi     temp0,0xFF              ;napln temp0 hodnotou 0xFF (ukoncovaci znak)
288                 out     UDR,temp0               ;a vysli ukoncovaci znak na UART
289                 out     SREG,backupSREGTimer    ;obnova SREG (a sucasne povolenie interruptu)
290                 rcall   InitCounter             ;restartni citac na TSOP zmenu a zakaz interrupt
291                 pop     temp0
292                 reti
293 ;***************************************************************
294 ;* Init program
295 ;***************************************************************
296 ;-------------------------------------------------------------------------------------------
297 reset:                  ;inicializacia procesora a premennych na spravne hodnoty
298                 ldi     temp0,StackBegin        ;inicializacia stacku
299                 out     SPL,temp0
300
301                 clr     XH                      ;Infra pointer
302                 clr     YH                      ;USB pointer
303                 clr     ZH                      ;ROM pointer
304                 sts     InfraBufferBegin,YH     ;znuluj dlzky Infra kodu v bufferi
305                 sts     InfraBufferBegin+1,YH   ;znuluj pocitadlo infra kodov v bufferi
306
307                 clr     MyUpdatedAddress        ;nova adresa USB -  nedekodovana
308                 rcall   InitACKBufffer          ;inicializacia ACK buffera
309                 rcall   InitNAKBufffer          ;inicializacia NAK buffera
310
311                 rcall   USBReset                ;inicializacia USB adresy
312
313                 sbi     TSOPpullupPort,TSOPpin  ;nahodit pull-up na TSOP vstupe
314
315                 ldi     temp0,(1<<LEDlsb0)+(1<<LEDlsb1)+(1<<LEDlsb2)
316                 out     LEDPortLSB,temp0        ;nahodit pull-up na vsetkych LED vstupoch LSB
317                 ldi     temp0,(1<<LEDmsb3)+(1<<LEDmsb4)+(1<<LEDmsb5)+(1<<LEDmsb6)+(1<<LEDmsb7)
318                 out     LEDPortMSB,temp0        ;nahodit pull-up na vsetkych LED vstupoch MSB
319
320                 sbi     PORTD,0                 ;nahodit pull-up na RxD vstupe
321                 ldi     temp0,BaudRate          ;nastavitvysielaciu rychlost UART-u
322                 out     UBRR,temp0
323                 sbi     UCR,TXEN                ;povolit vysielanie UART-u
324                 sbi     UCR,RXEN                ;povolit prijimanie UART-u
325
326                 ldi     temp0,0x0F              ;INT0 - reagovanie na nabeznu hranu
327                 out     MCUCR,temp0             ;
328                 ldi     temp0,1<<INT0           ;povolit externy interrupt INT0
329                 out     GIMSK,temp0
330                 rcall   InitCounter             ;inicializacia citaca na TSOP zmenu
331 ;-------------------------------------------------------------------------------------------
332 ;***************************************************************
333 ;* Main program
334 ;***************************************************************
335                 sei                                     ;povolit interrupty globalne
336 Main:
337                 sbis    inputport,DATAminus     ;cakanie az sa zmeni D- na 0
338                 rjmp    CheckUSBReset           ;a skontroluj, ci to nie je USB reset
339
340                 cpi     ActionFlag,DoReceiveSetupData
341                 breq    ProcReceiveSetupData
342                 cpi     ActionFlag,DoPrepareOutContinuousBuffer
343                 breq    ProcPrepareOutContinuousBuffer
344                 rjmp    Main
345
346 CheckUSBReset:
347                 ldi     temp0,255               ;pocitadlo trvania reset-u (podla normy je to cca 10ms - tu je to cca 100us)
348 WaitForUSBReset:
349                 sbic    inputport,DATAminus     ;cakanie az sa zmeni D+ na 0
350                 rjmp    Main
351                 dec     temp0
352                 brne    WaitForUSBReset
353                 rcall   USBReset
354                 rjmp    Main
355
356 ProcPrepareOutContinuousBuffer:
357                 rcall   PrepareOutContinuousBuffer      ;priprav pokracovanie odpovede do buffera
358                 ldi     ActionFlag,DoReadySendAnswer
359                 rjmp    Main
360 ProcReceiveSetupData:
361                 ldi     USBBufptrY,InputBufferBegin     ;pointer na zaciatok prijimacieho buffera
362                 mov     ByteCount,InputBufferLength     ;dlzka vstupneho buffera
363                 rcall   DecodeNRZI              ;prevod kodovania NRZI na bity
364                 rcall   MirrorInBufferBytes     ;prehodit poradie bitov v bajtoch
365                 rcall   BitStuff                ;odstranenie bit stuffing
366                 rcall   CheckCRCIn              ;kontrola CRC
367                 rcall   PrepareUSBOutAnswer     ;pripravenie odpovede do vysielacieho buffera
368                 ldi     ActionFlag,DoReadySendAnswer
369                 rjmp    Main
370 ;***************************************************************
371 ;* Main program END
372 ;***************************************************************
373 ;-------------------------------------------------------------------------------------------
374 ;***************************************************************
375 ;* Interrupt0 interrupt handler
376 ;***************************************************************
377 INT0Handler:                                    ;prerusenie INT0
378                 in      backupSREG,SREG
379                 push    temp0
380                 push    temp1
381
382                 ldi     temp0,3                 ;pocitadlo trvania log0
383                 ldi     temp1,2                 ;pocitadlo trvania log1
```

```
384                 ;cakanie na zaciatok paketu
385 CheckchangeMinus:
386                 sbis    inputport,DATAminus     ;cakanie az sa zmeni D- na 1
387                 rjmp    CheckchangeMinus
388 CheckchangePlus:
389                 sbis    inputport,DATAplus      ;cakanie az sa zmeni D+ na 1
390                 rjmp    CheckchangePlus
391 DetectSOPEnd:
392                 sbis    inputport,DATAplus
393                 rjmp    Increment0              ;D+ =0
394 Increment1:
395                 ldi     temp0,3                 ;pocitadlo trvania log0
396                 dec     temp1                   ;kolko cyklov trvala log1
397                 nop
398                 breq    USBBeginPacket          ;ak je to koniec SOP - prijimaj paket
399                 rjmp    DetectSOPEnd
400 Increment0:
401                 ldi     temp1,2                 ;pocitadlo trvania log1
402                 dec     temp0                   ;kolko cyklov trvala log0
403                 nop
404                 brne    DetectSOPEnd            ;ak nenastal SOF - pokracuj
405                 rjmp    EndInt0Handler2
406 EndInt0Handler:
407                 pop     ACC
408                 pop     InfraBufptrX
409                 pop     temp3
410                 pop     temp2
411 EndInt0HandlerPOP:
412                 pop     USBBufptrY
413                 pop     ByteCount
414                 mov     bitcount,backupbitcount ;obnova bitcount registra
415 EndInt0Handler2:
416                 pop     temp1
417                 pop     temp0
418                 out     SREG,backupSREG
419                 ldi     shiftbuf,1<<INTF0       ;znulovat flag interruptu INTF0
420                 out     GIFR,shiftbuf
421                 reti                            ;inak skonci (bol iba SOF - kazdu milisekundu)
422
423 USBBeginPacket:
424                 mov     backupbitcount,bitcount ;zaloha bitcount registra
425                 in      shiftbuf,inputport      ;ak ano nacitaj ho ako nulty bit priamo do shift registra
426 USBloopBegin:
427                 push    ByteCount               ;dalsia zaloha registrov (setrenie casu)
428                 push    USBBufptrY
429                 ldi     bitcount,6              ;inicializacia pocitadla bitov v bajte
430                 ldi     ByteCount,MAXUSBBYTES   ;inicializacia max poctu prijatych bajtov v pakete
431                 ldi     USBBufptrY,InputShiftBufferBegin        ;nastav vstupny buffer
432 USBloop1_6:
433                 in      inputbuf,inputport
434                 cbr     inputbuf,USBpinmask     ;odmaskovat spodne 2 bity
435                 breq    USBloopEnd              ;ak su nulove - koniec USB packetu
436                 ror     inputbuf                ;presun Data+ do shift registra
437                 rol     shiftbuf
438                 dec     bitcount                ;zmensi pocitadlo bitov
439                 brne    USBloop1_6              ;ak nie je nulove - opakuj naplnanie shift registra
440                 nop                             ;inak bude nutne skopirovat shift register bo buffera
441 USBloop7:
442                 in      inputbuf,inputport
443                 cbr     inputbuf,USBpinmask     ;odmaskovat spodne 2 bity
444                 breq    USBloopEnd              ;ak su nulove - koniec USB packetu
445                 ror     inputbuf                ;presun Data+ do shift registra
446                 rol     shiftbuf
447                 ldi     bitcount,7              ;inicializacia pocitadla bitov v bajte
448                 st      Y+,shiftbuf             ;skopiruj shift register bo buffera a zvys pointer do buffera
449 USBloop0:                                      ;a zacni prijimat dalsi bajt
450                 in      shiftbuf,inputport      ;nulty bit priamo do shift registra
451                 cbr     shiftbuf,USBpinmask     ;odmaskovat spodne 2 bity
452                 breq    USBloopEnd              ;ak su nulove - koniec USB packetu
453                 dec     bitcount                ;zmensi pocitadlo bitov
454                 nop                             ;
455                 dec     ByteCount               ;ak sa nedosiahol maximum buffera
456                 brne    USBloop1_6              ;tak prijimaj dalej
457
458                 rjmp    EndInt0HandlerPOP       ;inak opakuj od zaciatku
459
460 USBloopEnd:
461                 cpi     USBBufptrY,InputShiftBufferBegin+3      ;ak sa neprijali aspon 3 byte
462                 brcs    EndInt0HandlerPOP       ;tak skonci
463                 lds     temp0,InputShiftBufferBegin+0   ;identifikator paketu do temp0
464                 lds     temp1,InputShiftBufferBegin+1   ;adresa do temp1
465                 brne    TestDataPacket          ;ak je dlzka ina ako 3 - tak to moze byt iba DataPaket
466 TestIOPacket:
467                 cp      temp1,MyAddress         ;ak to nie je urcene (adresa) pre mna
468                 brne    TestDataPacket          ;tak to moze byt este Data Packet
469 TestSetupPacket:;test na SETUP paket
470                 cpi     temp0,nNRZISETUPPID
471                 brne    TestOutPacket           ;ak nie je Setup PID - dekoduj iny paket
472                 ldi     State,SetupState
473                 rjmp    EndInt0HandlerPOP       ;ak je Setup PID - prijimaj nasledny Data paket
474 TestOutPacket:  ;test na OUT paket
475                 cpi     temp0,nNRZIOUTPID
476                 brne    TestInPacket            ;ak nie je Out PID - dekoduj iny paket
477                 ldi     State,OutState
478                 rjmp    EndInt0HandlerPOP       ;ak je Out PID - prijimaj nasledny Data paket
479 TestInPacket:   ;test na IN paket
480                 cpi     temp0,nNRZIINPID
481                 brne    TestDataPacket          ;ak nie je In PID - dekoduj iny paket
482                 rjmp    AnswerToInRequest
483 TestDataPacket: ;test na DATA0 a DATA1 paket
484                 cpi     temp0,nNRZIDATA0PID
485                 breq    Data0Packet             ;ak nie je Data0 PID - dekoduj iny paket
486                 cpi     temp0,nNRZIDATA1PID
487                 brne    NoMyPacked              ;ak nie je Data1 PID - dekoduj iny paket
488 Data0Packet:
489                 cpi     State,SetupState        ;ak bol stav Setup
490                 breq    ReceiveSetupData        ;prijmi ho
491                 cpi     State,OutState          ;ak bol stav Out
492                 breq    ReceiveOutData          ;prijmi ho
493 NoMyPacked:
494                 ldi     State,BaseState         ;znuluj stav
495                 rjmp    EndInt0HandlerPOP       ;a prijimaj nasledny Data paket
496
497 AnswerToInRequest:
498                 push    temp2                   ;zazalohuj dalsie registre a pokracuj
499                 push    temp3
500                 push    InfraBufptrX
501                 push    ACC
502                 cpi     ActionFlag,DoReadySendAnswer    ;ak nie je pripravena odpoved
503                 brne    NoReadySend             ;tak posli NAK
504                 rcall   SendPreparedUSBAnswer   ;poslanie odpovede naspat
505                 and     MyUpdatedAddress,MyUpdatedAddress       ;ak je MyUpdatedAddress nenulova
506                 brne    SetMyNewUSBAddress      ;tak treba zmenit USB adresu
507                 ldi     State,InState
508                 ldi     ActionFlag,DoPrepareOutContinuousBuffer
509                 rjmp    EndInt0Handler          ;a opakuj - cakaj na dalsiu odozvu z USB
510 ReceiveSetupData:
511                 push    temp2                   ;zazalohuj dalsie registre a pokracuj
512                 push    temp3
```

```
513                 push    InfraBufptrX
514                 push    ACC
515                 rcall   SendACK                 ;akceptovanie Setup Data paketu
516                 rcall   FinishReceiving         ;ukonci prijem
517                 ldi     ActionFlag,DoReceiveSetupData
518                 rjmp    EndInt0Handler
519 ReceiveOutData:
520                 push    temp2                   ;zazalohuj dalsie registre a pokracuj
521                 push    temp3
522                 push    InfraBufptrX
523                 push    ACC
524                 cpi     ActionFlag,DoReceiveSetupData   ;ak sa prave spracovava prikaz Setup
525                 breq    NoReadySend             ;tak posli NAK
526                 rcall   SendACK                 ;akceptovanie Out paketu
527                 clr     ActionFlag
528                 rjmp    EndInt0Handler
529 NoReadySend:
530                 rcall   SendNAK                 ;este nie som pripraveny s odpovedou
531                 rjmp    EndInt0Handler          ;a opakuj - cakaj na dalsiu odozvu z USB
532 ;-----------------------------------------------------------------------------------------
533 SetMyNewUSBAddress:            ;nastavi novu USB adresu v NRZI kodovani
534                 clr     MyAddress               ;vychodzi stav odpovede - mojej nNRZI USB adresy
535                 ldi     temp2,0b00000001        ;maska na xorovanie
536                 ldi     temp3,8                 ;pocitadlo bitov
537 SetMyNewUSBAddressLoop:
538                 mov     temp0,MyAddress         ;zapamatat si koncovu odpoved
539                 ror     MyUpdatedAddress        ;do carry vysielany bit LSB (v smere naskor LSB a potom MSB)
540                 brcs    NoXORBit                ;ak je jedna - nemen stav
541                 eor     temp0,temp2             ;inak sa bude stav menit podla posledneho bitu odpovede
542 NoXORBit:
543                 ror     temp0                   ;posledny bit zmenenej odpovede do carry
544                 rol     MyAddress               ;a z carry do koncovej odpovede na miesto LSB (a sucasne prehodenie LSB a MSB poradia)
545                 dec     temp3                   ;zmensi pocitadlo bitov
546                 brne    SetMyNewUSBAddressLoop  ;ak pocitadlo bitov nie je nulove opakuj vysielanie s dalsim bitom
547                 clr     MyUpdatedAddress        ;znulovanie adresy ako priznak jej buduceho nemenenia
548                 rjmp    EndInt0Handler
549 ;-----------------------------------------------------------------------------------------
550 FinishReceiving:               ;korekcne akcie na ukoncenie prijmu
551                 cpi     bitcount,7              ;prenes do buffera aj posledny necely byte
552                 breq    NoRemainingBits         ;ak boli vsetky byty prenesene, tak neprenasaj nic
553                 inc     bitcount
554 ShiftRemainingBits:
555                 rol     shiftbuf                ;posun ostavajuce necele bity na spravnu poziciu
556                 dec     bitcount
557                 brne    ShiftRemainingBits
558                 st      Y+,shiftbuf             ;a skopiruj shift register bo buffera - necely byte
559 NoRemainingBits:
560                 mov     ByteCount,USBBufptrY
561                 subi    ByteCount,InputShiftBufferBegin-1       ;v ByteCount je pocet prijatych byte (vratane necelych byte)
562
563                 mov     InputBufferLength,ByteCount             ;a uchovat pre pouzitie v hlavnom programe
564                 ldi     USBBufptrY,InputShiftBufferBegin        ;pointer na zaciatok prijimacieho shift buffera
565                 ldi     InfraBufptrX,InputBufferBegin+1         ;data buffer (vynechat SOP)
566 MoveDataBuffer:
567                 ld      temp0,Y+
568                 st      X+,temp0
569                 dec     ByteCount
570                 brne    MoveDataBuffer
571
572                 ldi     ByteCount,nNRZISOPbyte
573                 sts     InputBufferBegin,ByteCount              ;ako keby sa prijal SOP - nekopiruje sa zo shift buffera
574                 ret
575 ;-----------------------------------------------------------------------------------------
576 ;*****************************************************************
577 ;* Other procedures
578 ;*****************************************************************
579 ;-----------------------------------------------------------------------------------------
580 USBReset:               ;inicializacia USB stavoveho stroja
581                 ldi     temp0,nNRZIADDR0        ;inicializacia USB adresy
582                 mov     MyAddress,temp0
583                 clr     State                   ;inicializacia stavoveho stroja
584                 clr     BitStuffInOut
585                 clr     OutBitStuffNumber
586                 clr     ActionFlag
587                 clr     RAMread                 ;bude sa vycitavat z ROM-ky
588                 clr     ConfigByte              ;nenakonfiguravany stav
589                 ret
590 ;-----------------------------------------------------------------------------------------
591 InitCounter:            ;inicializacia countera na prijem zmeny z TSOP
592                 ldi     temp0,6                 ;na zostupnu hranu casovaca - externy pin
593                 out     TCCR0,temp0
594                 ldi     temp0,0xFF              ;napln citac naplno - aby hned na nasledujucu hranu pretiekol
595                 out     TCNT0,temp0
596                 ldi     temp0,1<<TOV0           ;znuluj pripadny cakajuci flag pretecenia citaca
597                 out     TIFR,temp0
598                 ldi     temp0,1<<TOIE0          ;povolit interrupt od casovaca
599         ;cli                            ;zakaz interrupt kvoli zacykleniu
600                 out     TIMSK,temp0
601                 ret
602 ;-----------------------------------------------------------------------------------------
603 SendPreparedUSBAnswer:  ;poslanie kodovanim NRZI OUT buffer s dlzkou OutputBufferLength do USB
604                 mov     ByteCount,OutputBufferLength            ;dlzka odpovede
605 SendUSBAnswer:  ;poslanie kodovanim NRZI OUT buffer do USB
606                 ldi     USBBufptrY,OutputBufferBegin            ;pointer na zaciatok vysielacieho buffera
607 SendUSBBuffer:  ;poslanie kodovanim NRZI dany buffer do USB
608                 ldi     temp1,0                 ;zvysovanie pointra (pomocna premenna)
609                 mov     temp3,ByteCount         ;pocitadlo bytov: temp3 = ByteCount
610                 ldi     temp2,0b00000011        ;maska na xorovanie
611                 ld      inputbuf,Y+             ;nacitanie prveho bytu do inputbuf a zvys pointer do buffera
612                                                 ;USB ako vystup:
613                 cbi     outputport,DATAplus     ;zhodenie DATAplus : kludovy stav portu USB
614                 sbi     outputport,DATAminus    ;nahodenie DATAminus : kludovy stav portu USB
615                 sbi     USBdirection,DATAplus   ;DATAplus ako vystupny
616                 sbi     USBdirection,DATAminus  ;DATAminus ako vystupny
617
618                 in      temp0,outputport        ;kludovy stav portu USB do temp0
619 SendUSBAnswerLoop:
620                 ldi     bitcount,7              ;pocitadlo bitov
621 SendUSBAnswerByteLoop:
622                 nop                             ;oneskorenie kvoli casovaniu
623                 ror     inputbuf                ;do carry vysielany bit (v smere naskor LSB a potom MSB)
624                 brcs    NoXORSend               ;ak je jedna - nemen stav na USB
625                 eor     temp0,temp2             ;inak sa bude stav menit
626 NoXORSend:
627                 out     outputport,temp0        ;vysli von na USB
628                 dec     bitcount                ;zmensi pocitadlo bitov - podla carry flagu
629                 brne    SendUSBAnswerByteLoop   ;ak pocitadlo bitov nie je nulove - opakuj vysielanie s dalsim bitom
630                 sbrs    inputbuf,0              ;ak je vysielany bit jedna - nemen stav na USB
631                 eor     temp0,temp2             ;inak sa bude stav menit
632 NoXORSendLSB:
633                 dec     temp3                   ;zniz pocitadlo bytov
634                 ld      inputbuf,Y+             ;nacitanie dalsieho bytu a zvys pointer do buffera
635                 out     outputport,temp0        ;vysli von na USB
636                 brne    SendUSBAnswerLoop       ;opakuj pre cely buffer (pokial temp3=0)
637
638                 mov     bitcount,OutBitStuffNumber      ;pocitadlo bitov pre bitstuff
639                 cpi     bitcount,0              ;ak nie je potrebny bitstuff
640                 breq    ZeroBitStuf
641 SendUSBAnswerBitstuffLoop:
```

```
642                 ror     inputbuf                ;do carry vysielany bit (v smere naskor LSB a potom MSB)
643                 brcs    NoXORBitstuffSend       ;ak je jedna - nemen stav na USB
644                 eor     temp0,temp2             ;inak sa bude stav menit
645 NoXORBitstuffSend:
646                 out     outputport,temp0        ;vysli von na USB
647                 nop                             ;oneskorenie kvoli casovaniu
648                 dec     bitcount                ;zmensi pocitadlo bitov - podla carry flagu
649                 brne    SendUSBAnswerBitstuffLoop       ;ak pocitadlo bitov nie je nulove - opakuj vysielanie s dalsim bitom
650                 ld      inputbuf,Y              ;oneskorenie 2 cykly
651 ZeroBitStuf:
652                 nop                             ;oneskorenie 1 cyklus
653                 cbr     temp0,3
654                 out     outputport,temp0        ;vysli EOP na USB
655
656                 ldi     bitcount,5              ;pocitadlo oneskorenia: EOP ma trvat 2 bity (16 cyklov pri 12MHz)
657 SendUSBWaitEOP:
658                 dec     bitcount
659                 brne    SendUSBWaitEOP
660
661                 sbi     outputport,DATAminus    ;nahodenie DATAminus : kludovy stav na port USB
662                 sbi     outputport,DATAminus    ;oneskorenie 2 cykly: Idle ma trvat 1 bit (8 cyklov pri 12MHz)
663                 cbi     USBdirection,DATAplus   ;DATAplus ako vstupny
664                 cbi     USBdirection,DATAminus  ;DATAminus ako vstupny
665                 cbi     outputport,DATAminus    ;zhodenie DATAminus : treti stav na port USB
666                 ret
667 ;---------------------------------------------------------------------------------------
668 ToggleDATAPID:
669                 lds     temp0,OutputBufferBegin+1       ;nahraj posledne PID
670                 cpi     temp0,DATA1PID                  ;ak bolo posledne DATA1PID byte
671                 ldi     temp0,DATA0PID
672                 breq    SendData0PID                    ;tak posli nulovu odpoved s DATA0PID
673                 ldi     temp0,DATA1PID                  ;inak posli nulovu odpoved s DATA1PID
674 SendData0PID:
675                 sts     OutputBufferBegin+1,temp0       ;DATA0PID byte
676                 ret
677 ;---------------------------------------------------------------------------------------
678 ComposeZeroDATA1PIDAnswer:
679                 ldi     temp0,DATA0PID                  ;DATA0 PID - v skutocnosti sa stoggluje na DATA1PID v nahrati deskriptora
680                 sts     OutputBufferBegin+1,temp0       ;nahraj do vyst buffera
681 ComposeZeroAnswer:
682                 ldi     temp0,SOPbyte
683                 sts     OutputBufferBegin+0,temp0       ;SOP byte
684                 rcall   ToggleDATAPID                   ;zmen DATAPID
685                 ldi     temp0,0x00
686                 sts     OutputBufferBegin+2,temp0       ;CRC byte
687                 sts     OutputBufferBegin+3,temp0       ;CRC byte
688                 ldi     ByteCount,2+2                   ;dlzka vystupneho buffera (SOP a PID + CRC16)
689                 ret
690 ;---------------------------------------------------------------------------------------
691 InitACKBufffer:
692                 ldi     temp0,SOPbyte
693                 sts     ACKBufferBegin+0,temp0          ;SOP byte
694                 ldi     temp0,ACKPID
695                 sts     ACKBufferBegin+1,temp0          ;ACKPID byte
696                 ret
697 ;---------------------------------------------------------------------------------------
698 SendACK:
699                 push    USBBufptrY
700                 push    bitcount
701                 push    OutBitStuffNumber
702                 ldi     USBBufptrY,ACKBufferBegin       ;pointer na zaciatok ACK buffera
703                 ldi     ByteCount,2                     ;pocet vyslanych bytov (iba SOP a ACKPID)
704                 clr     OutBitStuffNumber
705                 rcall   SendUSBBuffer
706                 pop     OutBitStuffNumber
707                 pop     bitcount
708                 pop     USBBufptrY
709                 ret
710 ;---------------------------------------------------------------------------------------
711 InitNAKBufffer:
712                 ldi     temp0,SOPbyte
713                 sts     NAKBufferBegin+0,temp0          ;SOP byte
714                 ldi     temp0,NAKPID
715                 sts     NAKBufferBegin+1,temp0          ;NAKPID byte
716                 ret
717 ;---------------------------------------------------------------------------------------
718 SendNAK:
719                 push    OutBitStuffNumber
720                 ldi     USBBufptrY,NAKBufferBegin       ;pointer na zaciatok ACK buffera
721                 ldi     ByteCount,2                     ;pocet vyslanych bytov (iba SOP a NAKPID)
722                 clr     OutBitStuffNumber
723                 rcall   SendUSBBuffer
724                 pop     OutBitStuffNumber
725                 ret
726 ;---------------------------------------------------------------------------------------
727 ComposeSTALL:
728                 ldi     temp0,SOPbyte
729                 sts     OutputBufferBegin+0,temp0       ;SOP byte
730                 ldi     temp0,STALLPID
731                 sts     OutputBufferBegin+1,temp0       ;STALLPID byte
732                 ldi     ByteCount,2                     ;dlzka vystupneho buffera (SOP a PID)
733                 ret
734 ;---------------------------------------------------------------------------------------
735 DecodeNRZI:     ;enkodovanie buffera z NRZI kodu do binarneho
736                 push    USBBufptrY              ;zalohuj pointer do buffera
737                 push    ByteCount               ;zalohuj dlzku buffera
738                 add     ByteCount,USBBufptrY    ;koniec buffera do ByteCount
739                 ser     temp0                   ;na zabezpecenie jednotkoveho carry (v nasledujucej rotacii)
740 NRZIloop:
741                 ror     temp0                   ;naplnenie carry z predchadzajuceho byte
742                 ld      temp0,Y                 ;nahraj prijaty byte z buffera
743                 mov     temp2,temp0             ;posunuty register o jeden bit vpravo a XOR na funkciu NRZI dekodovania
744                 ror     temp2                   ;carry do najvyssieho bitu a sucasne posuv
745                 eor     temp2,temp0             ;samotne dekodovanie NRZI
746                 com     temp2                   ;negovanie
747                 st      Y+,temp2                ;ulozenie spat ako dekodovany byte a zvys pointer do buffera
748                 cp      USBBufptrY,ByteCount    ;ak este neboli vsetky
749                 brne    NRZIloop                ;tak opakuj
750                 pop     ByteCount               ;obnov dlzku buffera
751                 pop     USBBufptrY              ;obnov pointer do buffera
752                 ret                             ;inak skonci
753 ;---------------------------------------------------------------------------------------
754 BitStuff:       ;odstranenie bit-stuffingu v buffri
755                 clr     temp3                   ;pocitadlo vynechanych bitov
756                 clr     lastBitstufNumber       ;0xFF do lastBitstufNumber
757                 dec     lastBitstufNumber
758 BitStuffRepeat:
759                 push    USBBufptrY              ;zalohuj pointer do buffera
760                 push    ByteCount               ;zalohuj dlzku buffera
761                 mov     temp1,temp3             ;pocitadlo vsetkych bitov
762                 ldi     temp0,8                 ;spocitat vsetky bity v bufferi
763 SumAllBits:
764                 add     temp1,temp0
765                 dec     ByteCount
766                 brne    SumAllBits
767                 ldi     temp2,6                 ;inicializuj pocitadlo jednotiek
768                 pop     ByteCount               ;obnov dlzku buffera
769                 push    ByteCount               ;zalohuj dlzku buffera
770                 add     ByteCount,USBBufptrY    ;koniec buffera do ByteCount
```

```
771                 inc     ByteCount               ;a pre istotu ho zvys o 2 (kvoli posuvaniu)
772                 inc     ByteCount
773 BitStuffLoop:
774                 ld      temp0,Y                 ;nahraj prijaty byte z buffera
775                 ldi     bitcount,8              ;pocitadlo bitov v byte
776 BitStuffByteLoop:
777                 ror     temp0                   ;naplnenie carry z LSB
778                 brcs    IncrementBitstuff       ;ak LSB=0
779                 ldi     temp2,7                 ;inicializuj pocitadlo jednotiek +1 (ak bola nula)
780 IncrementBitstuff:
781                 dec     temp2                   ;zniz pocitadlo jednotiek (predpoklad jednotkoveho bitu)
782                 brne    NeposunBuffer           ;ak este nebolo 6 jednotiek za sebou - neposun buffer
783                 cp      temp1,lastBitstufNumber ;
784                 ldi     temp2,6                 ;inicializuj pocitadlo jednotiek (ak by sa nerobil bitstuffing tak sa musi zacat odznova)
785                 brcc    NeposunBuffer           ;ak sa tu uz robil bitstuffing - neposun buffer
786
787                 dec     temp1
788                 mov     lastBitstufNumber,temp1 ;zapamataj si poslednu poziciu bitstuffingu
789                 cpi     bitcount,1              ;aby sa ukazovalo na 7 bit (ktory sa ma vymazat alebo kde sa ma vlozit nula)
790                 brne    NoBitcountCorrect
791                 ldi     bitcount,9
792                 inc     USBBufptrY              ;zvys pointer do buffera
793 NoBitcountCorrect:
794                 dec     bitcount
795                 bst     BitStuffInOut,0
796                 brts    CorrectOutBuffer        ;ak je Out buffer - zvys dlzku buffera
797                 rcall   PosunDeleteBuffer       ;posun In buffer
798                 dec     temp3                   ;zniz pocitadlo vynechani
799                 rjmp    CorrectBufferEnd
800 CorrectOutBuffer:
801                 rcall   PosunInsertBuffer       ;posun Out buffer
802                 inc     temp3                   ;zvys pocitadlo vynechani
803 CorrectBufferEnd:
804                 pop     ByteCount               ;obnov dlzku buffera
805                 pop     USBBufptrY              ;obnov pointer do buffera
806                 rjmp    BitStuffRepeat          ;a restartni od zaciatku
807 NeposunBuffer:
808                 dec     temp1                   ;ak uz boli vsetky bity
809                 breq    EndBitStuff             ;ukonci cyklus
810                 dec     bitcount                ;zniz pocitadlo bitov v byte
811                 brne    BitStuffByteLoop        ;ak este neboli vsetky bity v byte - chod na dalsi bit
812                                                 ;inak nahraj dalsi byte
813                 inc     USBBufptrY              ;zvys pointer do buffera
814                 rjmp    BitStuffLoop            ;a opakuj
815 EndBitStuff:
816                 pop     ByteCount               ;obnov dlzku buffera
817                 pop     USBBufptrY              ;obnov pointer do buffera
818                 bst     BitStuffInOut,0
819                 brts    IncrementLength         ;ak je Out buffer - zvys dlzku Out buffera
820 DecrementLength:                               ;ak je In buffer - zniz dlzku In buffera
821                 cpi     temp3,0                 ;bolo aspon jedno znizenie
822                 breq    NoChangeByteCount       ;ak nie - nemen dlzku buffera
823                 dec     ByteCount               ;ak je In buffer - zniz dlzku buffera
824                 subi    temp3,256-8             ;ak nebolo viac ako 8 bitov naviac
825                 brcc    NoChangeByteCount       ;tak skonci
826                 dec     ByteCount               ;inak este zniz dlzku buffera
827                 ret                             ;a skonci
828 IncrementLength:
829                 mov     OutBitStuffNumber,temp3 ;zapamataj si pocet bitov naviac
830                 subi    temp3,8                 ;ak nebolo viac ako 8 bitov naviac
831                 brcs    NoChangeByteCount       ;tak skonci
832                 inc     ByteCount               ;inak zvys dlzku buffera
833                 mov     OutBitStuffNumber,temp3 ;a zapamataj si pocet bitov naviac (znizene o 8)
834 NoChangeByteCount:
835                 ret                             ;skonci
836 ;---------------------------------------------------------------------------------------
837 PosunInsertBuffer:      ;posuv buffera o jeden bit vpravo od konca az po poziciu: byte-USBBufptrY a bit-bitcount
838                 mov     temp0,bitcount          ;vypocet: bitcount= 9-bitcount
839                 ldi     bitcount,9
840                 sub     bitcount,temp0          ;do bitcount poloha bitu, ktory treba nulovat
841
842                 ld      temp1,Y                 ;nahraj byte ktory este treba posunut od pozicie bitcount
843                 rol     temp1                   ;a posun vlavo cez Carry (prenos z vyssieho byte a LSB do Carry)
844                 ser     temp2                   ;FF do masky - temp2
845 HalfInsertPosuvMask:
846                 lsl     temp2                   ;nula do dalsieho spodneho bitu masky
847                 dec     bitcount                ;az pokial sa nedosiahne hranica posuvania v byte
848                 brne    HalfInsertPosuvMask
849
850                 and     temp1,temp2             ;odmaskuj aby zostali iba vrchne posunute bity v temp1
851                 com     temp2                   ;invertuj masku
852                 lsr     temp2                   ;posun masku vpravo - na vlozenie nuloveho bitu
853                 ld      temp0,Y                 ;nahraj byte ktory este treba posunut od pozicie bitcount do temp0
854                 and     temp0,temp2             ;odmaskuj aby zostali iba spodne neposunute bity v temp0
855                 or      temp1,temp0             ;a zluc posunutu a neposunutu cast
856
857                 ld      temp0,Y                 ;nahraj byte ktory este treba posunut od pozicie bitcount
858                 rol     temp0                   ;a posun ho vlavo cez Carry (aby sa nastavilo spravne Carry pre dalsie prenosy)
859                 st      Y+,temp1                ;a nahraj spat upraveny byte
860 PosunInsertBufferLoop:
861                 cpse    USBBufptrY,ByteCount    ;ak nie su vsetky cele byty
862                 rjmp    NoEndPosunInsertBuffer  ;tak pokracuj
863                 ret                             ;inak skonci
864 NoEndPosunInsertBuffer:
865                 ld      temp1,Y                 ;nahraj byte
866                 rol     temp1                   ;a posun vlavo cez Carry (prenos z nizsieho byte a LSB do Carry)
867                 st      Y+,temp1                ;a nahraj spat
868                 rjmp    PosunInsertBufferLoop   ;a pokracuj
869 ;---------------------------------------------------------------------------------------
870 PosunDeleteBuffer:      ;posuv buffera o jeden bit vlavo od konca az po poziciu: byte-USBBufptrY a bit-bitcount
871                 mov     temp0,bitcount          ;vypocet: bitcount= 9-bitcount
872                 ldi     bitcount,9
873                 sub     bitcount,temp0          ;do bitcount poloha bitu, ktory este treba posunut
874                 mov     temp0,USBBufptrY        ;uschovanie pointera do buffera
875                 inc     temp0                   ;pozicia celych bytov do temp0
876                 mov     USBBufptrY,ByteCount    ;maximalna pozicia do pointra
877 PosunDeleteBufferLoop:
878                 ld      temp1,-Y                ;zniz buffer a nahraj byte
879                 ror     temp1                   ;a posun vpravo cez Carry (prenos z vyssieho byte a LSB do Carry)
880                 st      Y,temp1                 ;a nahraj spat
881                 cpse    USBBufptrY,temp0        ;ak nie su vsetky cele byty
882                 rjmp    PosunDeleteBufferLoop   ;tak pokracuj
883
884                 ld      temp1,-Y                ;zniz buffer a nahraj byte ktory este treba posunut od pozicie bitcount
885                 ror     temp1                   ;a posun vpravo cez Carry (prenos z vyssieho byte a LSB do Carry)
886                 ser     temp2                   ;FF do masky - temp2
887 HalfDeletePosuvMask:
888                 dec     bitcount                ;az pokial sa nedosiahne hranica posuvania v byte
889                 breq    DoneMask
890                 lsl     temp2                   ;nula do dalsieho spodneho bitu masky
891                 rjmp    HalfDeletePosuvMask
892 DoneMask:
893                 and     temp1,temp2             ;odmaskuj aby zostali iba vrchne posunute bity v temp1
894                 com     temp2                   ;invertuj masku
895                 ld      temp0,Y                 ;nahraj byte ktory este treba posunut od pozicie bitcount do temp0
896                 and     temp0,temp2             ;odmaskuj aby zostali iba spodne neposunute bity v temp0
897                 or      temp1,temp0             ;a zluc posunutu a neposunutu cast
898                 st      Y,temp1                 ;a nahraj spat
899                 ret                             ;a skonci
```

```
900 ;-----------------------------------------------------------------------------------------
901 MirrorInBufferBytes:
902                 push    USBBufptrY
903                 push    ByteCount
904                 ldi     USBBufptrY,InputBufferBegin
905                 rcall   MirrorBufferBytes
906                 pop     ByteCount
907                 pop     USBBufptrY
908                 ret
909 ;-----------------------------------------------------------------------------------------
910 MirrorBufferBytes:
911                 add     ByteCount,USBBufptrY    ;ByteCount ukazuje na koniec spravy
912 MirrorBufferloop:
913                 ld      temp0,Y                 ;nahraj prijaty byte z buffera
914                 ldi     temp1,8                 ;pocitadlo bitov
915 MirrorBufferByteLoop:
916                 ror     temp0                   ;do carry dalsi najnizsi bit
917                 rol     temp2                   ;z carry dalsi bit na obratene poradie
918                 dec     temp1                   ;bol uz cely byte
919                 brne    MirrorBufferByteLoop    ;ak nie tak opakuj dalsi najnizsi bit
920                 st      Y+,temp2                ;ulozenie spat ako obrateny byte  a zvys pointer do buffera
921                 cp      USBBufptrY,ByteCount    ;ak este neboli vsetky
922                 brne    MirrorBufferloop        ;tak opakuj
923                 ret                             ;inak skonci
924 ;-----------------------------------------------------------------------------------------
925 CheckCRCIn:
926                 push    USBBufptrY
927                 push    ByteCount
928                 ldi     USBBufptrY,InputBufferBegin
929                 rcall   CheckCRC
930                 pop     ByteCount
931                 pop     USBBufptrY
932                 ret
933 ;-----------------------------------------------------------------------------------------
934 AddCRCOut:
935                 push    USBBufptrY
936                 push    ByteCount
937                 ldi     USBBufptrY,OutputBufferBegin
938                 rcall   CheckCRC
939                 com     temp0                   ;negacia CRC
940                 com     temp1
941                 st      Y+,temp1                ;ulozenie CRC na koniec buffera (najskor MSB)
942                 st      Y,temp0                 ;ulozenie CRC na koniec buffera (potom LSB)
943                 dec     USBBufptrY              ;pointer na poziciu CRC
944                 ldi     ByteCount,2             ;otocit 2 byty CRC
945                 rcall   MirrorBufferBytes       ;opacne poradie bitov CRC (pri vysielani CRC sa posiela naskor MSB)
946                 pop     ByteCount
947                 pop     USBBufptrY
948                 ret
949 ;-----------------------------------------------------------------------------------------
950 CheckCRC:       ;vstup: USBBufptrY = zaciatok spravy    ,ByteCount = dlzka spravy
951                 add     ByteCount,USBBufptrY    ;ByteCount ukazuje na koniec spravy
952                 inc     USBBufptrY              ;nastav pointer na zaciatok spravy - vynechat SOP
953                 ld      temp0,Y+                ;nahraj PID do temp0
954                                                 ;a nastav pointer na zaciatok spravy - vynechat aj PID
955                 cpi     temp0,DATA0PID          ;ci je DATA0 pole
956                 breq    ComputeDATACRC          ;pocitaj CRC16
957                 cpi     temp0,DATA1PID          ;ci je DATA1 pole
958                 brne    CRC16End                ;ak nie tak skonci
959 ComputeDATACRC:
960                 ser     temp0                   ;inicializacia zvysku LSB na 0xff
961                 ser     temp1                   ;inicializacia zvysku MSB na 0xff
962 CRC16Loop:
963                 ld      temp2,Y+                ;nahraj spravu do temp2 a zvys pointer do buffera
964                 ldi     temp3,8                 ;pocitadlo bitov v byte - temp3
965 CRC16LoopByte:
966                 bst     temp1,7                 ;do T uloz MSB zvysku (zvysok je iba 16 bitovy - 8 bit vyssieho byte)
967                 bld     bitcount,0              ;do bitcount LSB uloz T - MSB zvysku
968                 eor     bitcount,temp2          ;XOR bitu spravy a bitu zvysku - v LSB bitcount
969                 rol     temp0                   ;posun zvysok dolava - nizsi byte (dva byty - cez carry)
970                 rol     temp1                   ;posun zvysok dolava - vyssi byte (dva byty - cez carry)
971                 cbr     temp0,1                 ;znuluj LSB zvysku
972                 lsr     temp2                   ;posun spravu doprava
973                 ror     bitcount                ;vysledok XOR-u bitov z LSB do carry
974                 brcc    CRC16NoXOR              ;ak je XOR bitu spravy a MSB zvysku = 0 , tak nerob XOR
975                 ldi     bitcount,CRC16poly>>8   ;do bitcount CRC polynom - vrchny byte
976                 eor     temp1,bitcount          ;a urob XOR zo zvyskom a CRC polynomom - vrchny byte
977                 ldi     bitcount,CRC16poly      ;do bitcount CRC polynom - spodny byte
978                 eor     temp0,bitcount          ;a urob XOR zo zvyskom a CRC polynomom - spodny byte
979 CRC16NoXOR:
980                 dec     temp3                   ;boli uz vsetky bity v byte
981                 brne    CRC16LoopByte           ;ak nie, tak chod na dalsi bit
982                 cp      USBBufptrY,ByteCount    ;bol uz koniec spravy
983                 brne    CRC16Loop               ;ak nie tak opakuj
984 CRC16End:
985                 ret                             ;inak skonci (v temp0 a temp1 je vysledok)
986 ;-----------------------------------------------------------------------------------------
987 LoadDescriptorFromROM:
988                 lpm                             ;nahraj z pozicie ROM pointra do R0
989                 st      Y+,R0                   ;R0 uloz do buffera a zvys buffer
990                 adiw    ZH:ZL,1                 ;zvys ukazovatel do ROM
991                 dec     ByteCount               ;pokial nie su vsetky byty
992                 brne    LoadDescriptorFromROM   ;tak nahravaj dalej
993                 rjmp    EndFromRAMROM           ;inak skonci
994 ;-----------------------------------------------------------------------------------------
995 LoadDescriptorFromROMZeroInsert:
996                 lpm                             ;nahraj z pozicie ROM pointra do R0
997                 st      Y+,R0                   ;R0 uloz do buffera a zvys buffer
998
999                 bst     RAMread,3               ;ak je 3 bit jednotkovy - nebude sa vkladat nula
1000                brtc    InsertingZero           ;inak sa bude vkladat nula
1001                adiw    ZH:ZL,1                 ;zvys ukazovatel do ROM
1002                lpm                             ;nahraj z pozicie ROM pointra do R0
1003                st      Y+,R0                   ;R0 uloz do buffera a zvys buffer
1004                clt                             ;a znuluj
1005                bld     RAMread,3               ;treti bit v RAMread - aby sa v dalsom vkladali nuly
1006                rjmp    InsertingZeroEnd        ;a pokracuj
1007 InsertingZero:
1008                clr     R0                      ;na vkladanie nul
1009                st      Y+,R0                   ;nulu uloz do buffera a zvys buffer
1010 InsertingZeroEnd:
1011                adiw    ZH:ZL,1                 ;zvys ukazovatel do ROM
1012                subi    ByteCount,2             ;pokial nie su vsetky byty
1013                brne    LoadDescriptorFromROMZeroInsert ;tak nahravaj dalej
1014                rjmp    EndFromRAMROM           ;inak skonci
1015 ;-----------------------------------------------------------------------------------------
1016 LoadDescriptorFromSRAM:
1017                ld      R0,Z                    ;nahraj z pozicie RAM pointra do R0
1018                st      Y+,R0                   ;R0 uloz do buffera a zvys buffer
1019                inc     ZL                      ;zvys ukazovatel do RAM
1020                dec     ByteCount               ;pokial nie su vsetky byty
1021                brne    LoadDescriptorFromSRAM  ;tak nahravaj dalej
1022                rjmp    EndFromRAMROM           ;inak skonci
1023 ;-----------------------------------------------------------------------------------------
1024 LoadDescriptorFromEEPROM:
1025                out     EEAR,ZL                 ;nastav adresu EEPROM
1026                sbi     EECR,EERE               ;vycitaj EEPROM do registra EEDR
1027                in      R0,EEDR                 ;nahraj z EEDR do R0
1028                st      Y+,R0                   ;R0 uloz do buffera a zvys buffer
```

```
1029                    inc     ZL                      ;zvys ukazovatel do RAM
1030                    dec     ByteCount               ;pokial nie su vsetky byty
1031                    brne    LoadDescriptorFromEEPROM;tak nahravaj dalej
1032                    rjmp    EndFromRAMROM           ;inak skonci
1033    ;---------------------------------------------------------------------------------------
1034    LoadXXXDescriptor:
1035                    ldi     temp0,SOPbyte                   ;SOP byte
1036                    sts     OutputBufferBegin,temp0         ;na zaciatok vysielacieho buffera dat SOP
1037                    ldi     ByteCount,8                     ;8 bytov nahrat
1038                    ldi     USBBufptrY,OutputBufferBegin+2  ;do vysielacieho buffera
1039
1040                    and     RAMread,RAMread                 ;ci sa bude citat z RAM alebo ROM-ky alebo EEPROM-ky
1041                    brne    FromRAMorEEPROM                 ;0=ROM,1=RAM,2=EEPROM,4=ROM s vkladanim nuly
1042    FromROM:
1043                    rjmp    LoadDescriptorFromROM           ;nahrat descriptor z ROM-ky
1044    FromRAMorEEPROM:
1045                    sbrc    RAMread,2                       ;ak RAMread=4
1046                    rjmp    LoadDescriptorFromROMZeroInsert ;citaj z ROM s vkladanim nuly
1047                    sbrc    RAMread,0                       ;ak RAMread=1
1048                    rjmp    LoadDescriptorFromSRAM          ;nahraj data zo SRAM-ky
1049                    rjmp    LoadDescriptorFromEEPROM        ;inak citaj z EEPROM
1050    EndFromRAMROM:
1051                    sbrc    RAMread,7                       ;ak je najvyssi bit v premennej RAMread=1
1052                    clr     RAMread                         ;znuluj RAMread
1053                    rcall   ToggleDATAPID                   ;zmenit DATAPID
1054                    ldi     USBBufptrY,OutputBufferBegin+1  ;do vysielacieho buffera - pozicia DATA PID
1055                    ret
1056    ;---------------------------------------------------------------------------------------
1057    PrepareUSBOutAnswer:    ;pripravenie odpovede do buffera
1058                    rcall   PrepareUSBAnswer                ;pripravenie odpovede do buffera
1059    MakeOutBitStuff:
1060                    inc     BitStuffInOut                   ;vysielaci buffer - vkladanie bitstuff bitov
1061                    ldi     USBBufptrY,OutputBufferBegin    ;do vysielacieho buffera
1062                    rcall   BitStuff
1063                    mov     OutputBufferLength,ByteCount    ;dlzku odpovede zapamatat pre vysielanie
1064                    clr     BitStuffInOut                   ;prijimaci buffer - mazanie bitstuff bitov
1065                    ret
1066    ;---------------------------------------------------------------------------------------
1067    PrepareUSBAnswer:       ;pripravenie odpovede do buffera
1068                    clr     RAMread                         ;nulu do RAMread premennej - cita sa z ROM-ky
1069                    lds     temp0,InputBufferBegin+2        ;bmRequestType do temp0
1070                    lds     temp1,InputBufferBegin+3        ;bRequest do temp1
1071                    cbr     temp0,0b10011111                ;ak je 5 a 6 bit nulovy
1072                    brne    VendorRequest                   ;tak to nie je  Vendor Request
1073                    rjmp    StandardRequest                 ;ale je to standardny request
1074    ;--------------------------
1075    DoSetInfraBufferEmpty:
1076                    clr     InfraBufferFull                 ;nastav prazdnost buffera
1077    SendZeroAnswerInfraEmpty:
1078                    rjmp    OneZeroAnswer                   ;potvrd prijem jednou nulou
1079    ;--------------------------
1080    VendorRequest:
1081                    clr     ZH                              ;pre citanie z RAM alebo EEPROM
1082
1083                    cpi     temp1,1                         ;
1084                    breq    DoSetInfraBufferEmpty           ;restartne infra prijimanie (ak bolo zastavene citanim z RAM-ky)
1085
1086                    cpi     temp1,2                         ;
1087                    breq    DoGetInfraCode                  ;vysle prijaty infra kod (ak je v bufferi)
1088
1089                    cpi     temp1,3                         ;
1090                    breq    DoSetDataPortDirection          ;nastavi smer toku datovych bitov
1091                    cpi     temp1,4                         ;
1092                    breq    DoGetDataPortDirection          ;zisti smer toku datovych bitov
1093
1094                    cpi     temp1,5                         ;
1095                    breq    DoSetOutDataPort                ;nastavi datove bity (ak su vstupne, tak ich pull-up)
1096                    cpi     temp1,6                         ;
1097                    breq    DoGetOutDataPort                ;zisti nastavenie datovych out bitov (ak su vstupne, tak ich pull-up)
1098
1099                    cpi     temp1,7                         ;
1100                    breq    DoGetInDataPort                 ;vrati hodnotu datoveho vstupneho portu
1101
1102                    cpi     temp1,8                         ;
1103                    breq    DoEEPROMRead                    ;vrati obsah EEPROM od urcitej adresy
1104                    cpi     temp1,9                         ;
1105                    breq    DoEEPROMWrite                   ;zapise EEPROM na urcitu adresu urcite data
1106
1107                    cpi     temp1,10                        ;
1108                    breq    DoRS232Send                     ;vysle byte na seriovy linku
1109                    cpi     temp1,11                        ;
1110                    breq    DoRS232Read                     ;vrati prijaty byte zo seriovej linky (ak sa nejaky prijal)
1111
1112                    cpi     temp1,12                        ;
1113                    breq    DoSetRS232Baud                  ;nastavi prenosovu rychlost seriovej linky
1114                    cpi     temp1,13                        ;
1115                    breq    DoGetRS232Baud                  ;vrati prenosovu rychlost seriovej linky
1116
1117                    rjmp    ZeroDATA1Answer                 ;ak to bolo nieco nezname, tak priprav nulovu odpoved
1118    ;---------------------------- USER FUNCTIONS ------------------------------------
1119    DoGetInfraCode:
1120                    cpi     InfraBufferFull,0               ;ak je Infra Buffer prazdny
1121                    breq    SendZeroAnswerInfraEmpty        ;tak nic neposli
1122                    ldi     ZL,InfraBufferBegin
1123                    lds     temp1,InputBufferBegin+4        ;prvy parameter - offset v infra bufferi
1124                    add     ZL,temp1                        ;plus offset v buffri
1125                    lds     temp0,InfraBufferBegin          ;pocet mojich bytovych odpovedi do temp0
1126                    ldi     temp1,3                         ;plus 3 byty hlavicky
1127                    add     temp0,temp1
1128                    inc     RAMread                         ;RAMread=1 - cita sa z RAM-ky
1129                    rjmp    ComposeEndXXXDescriptor         ;a priprav data
1130    DoEEPROMRead:
1131                    lds     ZL,InputBufferBegin+4           ;prvy parameter - offset v EEPROM-ke
1132                    ldi     temp0,2
1133                    mov     RAMread,temp0                   ;RAMread=2 - cita sa z EEPROM-ky
1134                    ldi     temp0,E2END+1                   ;pocet mojich bytovych odpovedi do temp0 - cela dlzka EEPROM
1135                    rjmp    ComposeEndXXXDescriptor         ;inak priprav data
1136    DoEEPROMWrite:
1137                    lds     ZL,InputBufferBegin+4           ;prvy parameter - offset v EEPROM-ke (adresa)
1138                    lds     R0,InputBufferBegin+6           ;druhy parameter - data, ktore sa maju zapisat do EEPROM-ky (data)
1139                    rjmp    EEPROMWrite                     ;zapis do EEPROM a aj ukonci prikaz
1140    DoSetDataPortDirection:
1141                    lds     ACC,InputBufferBegin+4          ;prvy parameter - smer datovych bitov
1142                    rcall   SetDataPortDirection
1143                    rjmp    OneZeroAnswer                   ;potvrd prijem jednou nulou
1144    DoGetDataPortDirection:
1145                    rcall   GetDataPortDirection
1146                    rjmp    DoGetIn
1147
1148    DoSetOutDataPort:
1149                    lds     ACC,InputBufferBegin+4          ;prvy parameter - hodnota datovych bitov
1150                    rcall   SetOutDataPort
1151                    rjmp    OneZeroAnswer                   ;potvrd prijem jednou nulou
1152    DoGetOutDataPort:
1153                    rcall   GetOutDataPort
1154                    rjmp    DoGetIn
1155
1156    DoGetInDataPort:
1157                    rcall   GetInDataPort
```

```
1158 DoGetIn:
1159                 ldi     ZL,0                        ;posiela sa hodnota v R0
1160                 ldi     temp0,0x81                  ;RAMread=1 - cita sa z RAM-ky
1161                 mov     RAMread,temp0               ;(najvyssi bit na 1 - aby sa hned premenna RAMread znulovala)
1162                 ldi     temp0,1                     ;posli iba jeden byte
1163                 rjmp    ComposeEndXXXDescriptor     ;a priprav data
1164
1165 DoSetRS232Baud:
1166                 lds     temp0,InputBufferBegin+4    ;prvy parameter - hodnota baudrate na RS232
1167                 out     UBRR,temp0                  ;nastav rychlost UART-u
1168                 rjmp    OneZeroAnswer               ;potvrd prijem jednou nulou
1169 DoGetRS232Baud:
1170                 in      R0,UBRR                     ;vrat rychlost UART-u v R0
1171                 rjmp    DoGetIn                     ;a ukonci
1172 DoRS232Send:
1173                 lds     temp0,InputBufferBegin+4    ;prvy parameter - hodnota vysielana na RS232
1174                 out     UDR,temp0                   ;vysli data na UART
1175 WaitForRS232Send:
1176                 sbis    UCR,TXEN                    ;ak nie je povoleny UART vysielac
1177                 rjmp    OneZeroAnswer               ;tak skonci - ochrana kvoli zacykleniu v AT90S2323/2343
1178                 sbis    USR,TXC                     ;pockat na dovysielanie bytu
1179                 rjmp    WaitForRS232Send
1180                 rjmp    OneZeroAnswer               ;potvrd prijem jednou nulou
1181 DoRS232Read:
1182                 sbis    USR,RXC                     ;ak je prijimaci register prazdny
1183                 rjmp    TwoZeroAnswer               ;tak iba potvrd prijem dvoma nulami
1184                 in      R0,UDR                      ;inak nacitaj do R0 prijate data z UART-u
1185                 ldi     ZL,0                        ;posiela sa hodnota v R0
1186                 ldi     temp0,0x81                  ;RAMread=1 - cita sa z RAM-ky
1187                 mov     RAMread,temp0               ;(najvyssi bit na 1 - aby sa hned premenna RAMread znulovala)
1188                 ldi     temp0,1                     ;posli iba jeden byte
1189                 sbic    USR,OR                      ;ak bolo pretecenie prijimacieho buffera
1190                 ldi     temp0,3                     ;posli 3 byty :prvy je prijata hodnota (na ostatnych nezalezi)
1191                 rjmp    ComposeEndXXXDescriptor     ;tak priprav data
1192 ;----------------------------- END USER FUNCTIONS ---------------------------------
1193
1194 OneZeroAnswer:          ;posle jednu nulu
1195                 ldi     temp0,1                     ;pocet mojich bytovych odpovedi do temp0
1196                 rjmp    ComposeGET_STATUS2
1197
1198 StandardRequest:
1199                 cpi     temp1,GET_STATUS            ;
1200                 breq    ComposeGET_STATUS           ;
1201
1202                 cpi     temp1,CLEAR_FEATURE         ;
1203                 breq    ComposeCLEAR_FEATURE        ;
1204
1205                 cpi     temp1,SET_FEATURE           ;
1206                 breq    ComposeSET_FEATURE          ;
1207
1208                 cpi     temp1,SET_ADDRESS           ;ak sa ma nastavit adresa
1209                 breq    ComposeSET_ADDRESS          ;nastav adresu
1210
1211                 cpi     temp1,GET_DESCRIPTOR        ;ak sa ziada descriptor
1212                 breq    ComposeGET_DESCRIPTOR       ;vygeneruj ho
1213
1214                 cpi     temp1,SET_DESCRIPTOR        ;
1215                 breq    ComposeSET_DESCRIPTOR       ;
1216
1217                 cpi     temp1,GET_CONFIGURATION     ;
1218                 breq    ComposeGET_CONFIGURATION    ;
1219
1220                 cpi     temp1,SET_CONFIGURATION     ;
1221                 breq    ComposeSET_CONFIGURATION    ;
1222
1223                 cpi     temp1,GET_INTERFACE         ;
1224                 breq    ComposeGET_INTERFACE        ;
1225
1226                 cpi     temp1,SET_INTERFACE         ;
1227                 breq    ComposeSET_INTERFACE        ;
1228
1229                 cpi     temp1,SYNCH_FRAME           ;
1230                 breq    ComposeSYNCH_FRAME          ;
1231                                                     ;ak sa nenasla znama poziadavka
1232                 rjmp    ZeroDATA1Answer             ;ak to bolo nieco nezname, tak priprav nulovu odpoved
1233
1234 ComposeSET_ADDRESS:
1235                 lds     MyUpdatedAddress,InputBufferBegin+4     ;nova adresa do MyUpdatedAddress
1236                 rjmp    ZeroDATA1Answer             ;posli nulovu odpoved
1237
1238 ComposeSET_CONFIGURATION:
1239                 lds     ConfigByte,InputBufferBegin+4   ;cislo konfiguracie do premennej ConfigByte
1240 ComposeCLEAR_FEATURE:
1241 ComposeSET_FEATURE:
1242 ComposeSET_INTERFACE:
1243 ZeroStringAnswer:
1244                 rjmp    ZeroDATA1Answer             ;posli nulovu odpoved
1245 ComposeGET_STATUS:
1246 TwoZeroAnswer:
1247                 ldi     temp0,2                     ;pocet mojich bytovych odpovedi do temp0
1248 ComposeGET_STATUS2:
1249                 ldi     ZH, high(StatusAnswer<<1)   ;ROMpointer na odpoved
1250                 ldi     ZL,  low(StatusAnswer<<1)
1251                 rjmp    ComposeEndXXXDescriptor     ;a dokonci
1252 ComposeGET_CONFIGURATION:
1253                 and     ConfigByte,ConfigByte       ;ak som nenakonfigurovany
1254                 breq    OneZeroAnswer               ;tak posli jednu nulu - inak posli moju konfiguraciu
1255                 ldi     temp0,1                     ;pocet mojich bytovych odpovedi do temp0
1256                 ldi     ZH, high(ConfigAnswerMinus1<<1) ;ROMpointer na odpoved
1257                 ldi     ZL,  low(ConfigAnswerMinus1<<1)+1
1258                 rjmp    ComposeEndXXXDescriptor     ;a dokonci
1259 ComposeGET_INTERFACE:
1260                 ldi     ZH, high(InterfaceAnswer<<1)    ;ROMpointer na odpoved
1261                 ldi     ZL,  low(InterfaceAnswer<<1)
1262                 ldi     temp0,1                     ;pocet mojich bytovych odpovedi do temp0
1263                 rjmp    ComposeEndXXXDescriptor     ;a dokonci
1264 ComposeSYNCH_FRAME:
1265 ComposeSET_DESCRIPTOR:
1266                 rcall   ComposeSTALL
1267                 ret
1268 ComposeGET_DESCRIPTOR:
1269                 lds     temp1,InputBufferBegin+5    ;DescriptorType do temp1
1270                 cpi     temp1,DEVICE                ;DeviceDescriptor
1271                 breq    ComposeDeviceDescriptor     ;
1272                 cpi     temp1,CONFIGURATION         ;ConfigurationDescriptor
1273                 breq    ComposeConfigDescriptor     ;
1274                 cpi     temp1,STRING                ;StringDeviceDescriptor
1275                 breq    ComposeStringDescriptor     ;
1276                 ret
1277 ComposeDeviceDescriptor:
1278                 ldi     ZH, high(DeviceDescriptor<<1)   ;ROMpointer na descriptor
1279                 ldi     ZL,  low(DeviceDescriptor<<1)
1280                 ldi     temp0,0x12                  ;pocet mojich bytovych odpovedi do temp0
1281                 rjmp    ComposeEndXXXDescriptor     ;a dokonci
1282 ComposeConfigDescriptor:
1283                 ldi     ZH, high(ConfigDescriptor<<1)   ;ROMpointer na descriptor
1284                 ldi     ZL,  low(ConfigDescriptor<<1)
1285                 ldi     temp0,9+9+7                 ;pocet mojich bytovych odpovedi do temp0
1286 ComposeEndXXXDescriptor:
```

```
1287                    lds     TotalBytesToSend,InputBufferBegin+8     ;pocet pozadovanych bytov do TotalBytesToSend
1288                    cp      TotalBytesToSend,temp0                  ;ak sa neziada viac ako mozem dodat
1289                    brcs    HostConfigLength                ;vysli tolko kolko sa ziada
1290                    mov     TotalBytesToSend,temp0          ;inak posli pocet mojich odpovedi
1291    HostConfigLength:
1292                    mov     temp0,TotalBytesToSend          ;
1293                    clr     TransmitPart                    ;nuluj pocet 8 bytovych odpovedi
1294                    andi    temp0,0b00000111                ;ak je dlzka delitelna 8-mimi
1295                    breq    Length8Multiply                 ;tak nezapocitaj jednu necelu odpoved (pod 8 bytov)
1296                    inc     TransmitPart                    ;inak ju zapocitaj
1297    Length8Multiply:
1298                    mov     temp0,TotalBytesToSend          ;
1299                    lsr     temp0                           ;dlzka 8 bytovych odpovedi sa dosiahne
1300                    lsr     temp0                           ;delenie celociselne 8-mimi
1301                    lsr     temp0
1302                    add     TransmitPart,temp0              ;a pripocitanim k poslednej necelej 8-mici do premennej TransmitPart
1303                    ldi     temp0,DATA0PID                  ;DATA0 PID - v skutocnosti sa stoggluje na DATA1PID v nahrati deskriptora
1304                    sts     OutputBufferBegin+1,temp0       ;nahraj do vyst buffera
1305                    rjmp    ComposeNextAnswerPart
1306    ComposeStringDescriptor:
1307                    ldi     temp1,4+8                       ;ak RAMread=4(vkladaj nuly z ROM-koveho citania) + 8(za prvy byte nevkldadaj nulu)
1308                    mov     RAMread,temp1
1309                    lds     temp1,InputBufferBegin+4        ;DescriptorIndex do temp1
1310                    cpi     temp1,0                         ;LANGID String
1311                    breq    ComposeLangIDString             ;
1312                    cpi     temp1,2                         ;DevNameString
1313                    breq    ComposeDevNameString            ;
1314                    brcc    ZeroStringAnswer                ;ak je DescriptorIndex vyssi nez 2 - posli nulovu odpoved
1315                                                            ;inak to bude VendorString
1316    ComposeVendorString:
1317                    ldi     ZH, high(VendorStringDescriptor<<1)     ;ROMpointer na descriptor
1318                    ldi     ZL,  low(VendorStringDescriptor<<1)
1319                    ldi     temp0,(VendorStringDescriptorEnd-VendorStringDescriptor)*4-2    ;pocet mojich bytovych odpovedi do temp0
1320                    rjmp    ComposeEndXXXDescriptor         ;a dokonci
1321    ComposeDevNameString:
1322                    ldi     ZH, high(DevNameStringDescriptor<<1)    ;ROMpointer na descriptor
1323                    ldi     ZL,  low(DevNameStringDescriptor<<1)
1324                    ldi     temp0,(DevNameStringDescriptorEnd-DevNameStringDescriptor)*4-2  ;pocet mojich bytovych odpovedi do temp0
1325                    rjmp    ComposeEndXXXDescriptor         ;a dokonci
1326    ComposeLangIDString:
1327                    clr     RAMread
1328                    ldi     ZH, high(LangIDStringDescriptor<<1)     ;ROMpointer na descriptor
1329                    ldi     ZL,  low(LangIDStringDescriptor<<1)
1330                    ldi     temp0,(LangIDStringDescriptorEnd-LangIDStringDescriptor)*2;pocet mojich bytovych odpovedi do temp0
1331                    rjmp    ComposeEndXXXDescriptor         ;a dokonci
1332    ;---------------------------------------------------------------------------------------------
1333    ZeroDATA1Answer:
1334                    rcall   ComposeZeroDATA1PIDAnswer
1335                    ret
1336    ;---------------------------------------------------------------------------------------------
1337    PrepareOutContinuousBuffer:
1338                    rcall   PrepareContinuousBuffer
1339                    rcall   MakeOutBitStuff
1340                    ret
1341    ;---------------------------------------------------------------------------------------------
1342    PrepareContinuousBuffer:
1343                    mov     temp0,TransmitPart
1344                    cpi     temp0,1
1345                    brne    NextAnswerInBuffer              ;ak uz je buffer prazdny
1346                    rcall   ComposeZeroAnswer               ;priprav nulovu odpoved
1347                    ret
1348    NextAnswerInBuffer:
1349                    dec     TransmitPart                    ;znizit celkovu dlzku odpovede
1350    ComposeNextAnswerPart:
1351                    mov     temp1,TotalBytesToSend  ;zniz pocet bytov na vyslanie
1352                    subi    temp1,8                 ;ci je este treba poslat viac ako 8 bytov
1353                    ldi     temp3,8                 ;ak ano - posli iba 8 bytov
1354                    brcc    Nad8Bytov
1355                    mov     temp3,TotalBytesToSend  ;inak posli iba dany pocet bytov
1356                    clr     TransmitPart
1357                    inc     TransmitPart            ;a bude to posledna odpoved
1358    Nad8Bytov:
1359                    mov     TotalBytesToSend,temp1  ;znizeny pocet bytov do TotalBytesToSend
1360                    rcall   LoadXXXDescriptor
1361                    ldi     ByteCount,2             ;dlzka vystupneho buffera (iba SOP a PID)
1362                    add     ByteCount,temp3         ;+ pocet bytov
1363                    rcall   AddCRCOut               ;pridanie CRC do buffera
1364                    inc     ByteCount               ;dlzka vystupneho buffera + CRC16
1365                    inc     ByteCount
1366                    ret                             ;skonci
1367    ;---------------------------------------------------------------------------------------------
1368    .equ    USBversion              =0x0100         ;pre aku verziu USB je to (1.00)
1369    .equ    VendorUSBID             =0x03EB         ;identifikator dodavatela (Atmel=0x03EB)
1370    .equ    DeviceUSBID             =0x0002         ;identifikator vyrobku (USB dialkove ovladanie=0x0002)
1371    .equ    DeviceVersion           =0x0001         ;cislo verzie vyrobku (verzia=0.01)
1372    .equ    MaxUSBCurrent           =40             ;prudovy odber z USB (40mA)
1373    ;---------------------------------------------------------------------------------------------
1374    DeviceDescriptor:
1375                    .db     0x12,0x01               ;0 byte - velkost deskriptora v bytoch
1376                                                    ;1 byte - typ deskriptora: Deskriptor zariadenia
1377                    .dw     USBversion              ;2,3 byte - verzia USB LSB (1.00)
1378                    .db     0x00,0x00               ;4 byte - trieda zariadenia
1379                                                    ;5 byte - podtrieda zariadenia
1380                    .db     0x00,0x08               ;6 byte - kod protokolu
1381                                                    ;7 byte - velkost FIFO v bytoch
1382                    .dw     VendorUSBID             ;8,9 byte - identifikator dodavatela (Cypress=0x04B4)
1383                    .dw     DeviceUSBID             ;10,11 byte - identifikator vyrobku (teplomer=0x0002)
1384                    .dw     DeviceVersion           ;12,13 byte - cislo verzie vyrobku (verzia=0.01)
1385                    .db     0x01,0x02               ;14 byte - index stringu "vyrobca"
1386                                                    ;15 byte - index stringu "vyrobok"
1387                    .db     0x00,0x01               ;16 byte - index stringu "seriove cislo"
1388                                                    ;17 byte - pocet moznych konfiguracii
1389    DeviceDescriptorEnd:
1390    ;---------------------------------------------------------------------------------------------
1391    ConfigDescriptor:
1392                    .db     0x9,0x02                ;dlzka,typ deskriptoru
1393    ConfigDescriptorLength:
1394                    .dw     9+9+7                   ;celkova dlzka vsetkych deskriptorov
1395            ConfigAnswerMinus1:                     ;pre poslanie cisla congiguration number (pozor je  treba este pricitat 1)
1396                    .db     1,1                     ;numInterfaces,congiguration number
1397                    .db     0,0x80                  ;popisny index stringu, atributy;bus powered
1398                    .db     MaxUSBCurrent/2,0x09    ;prudovy odber, interface descriptor length
1399                    .db     0x04,0                  ;interface descriptor; cislo interface
1400            InterfaceAnswer:                        ;pre poslanie cisla alternativneho interface
1401                    .db     0,1                     ;alternativne nastavenie interface; pocet koncovych bodov okrem EP0
1402            StatusAnswer:                           ;2 nulove odpovede (na usetrenie miestom)
1403                    .db     0,0                     ;trieda rozhrania; podtrieda rozhrania
1404                    .db     0,0                     ;kod protokolu; index popisneho stringu
1405                    .db     0x07,0x5                ;dlzka,typ deskriptoru - endpoint
1406                    .db     0x81,0                  ;endpoint address; transfer type
1407                    .dw     0x08                    ;max packet size
1408                    .db     10,0                    ;polling interval [ms]; dummy byte (pre vyplnenie)
1409    ConfigDescriptorEnd:
1410    ;---------------------------------------------------------------------------------------------
1411    LangIDStringDescriptor:
1412                    .db     (LangIDStringDescriptorEnd-LangIDStringDescriptor)*2,3  ;dlzka, typ: string deskriptor
1413                    .dw     0x0409                  ;English
1414    LangIDStringDescriptorEnd:
1415    ;---------------------------------------------------------------------------------------------
```

```
1416 VendorStringDescriptor:
1417                 .db     (VendorStringDescriptorEnd-VendorStringDescriptor)*4-2,3        ;dlzka, typ: string deskriptor
1418 CopyRight:
1419                 .db     "Ing. Igor Cesko, Copyright(c) 2003"
1420 CopyRightEnd:
1421 VendorStringDescriptorEnd:
1422 ;-----------------------------------------------------------------------------------------
1423 DevNameStringDescriptor:
1424                 .db     (DevNameStringDescriptorEnd-DevNameStringDescriptor)*4-2,3;dlzka, typ: string deskriptor
1425                 .db     "IgorPlug-USB (AVR)"
1426 DevNameStringDescriptorEnd:
1427 ;-----------------------------------------------------------------------------------------
1428 MaskPortData:
1429                 bst     ACC,0
1430                 bld     temp0,LEDlsb0
1431                 bst     ACC,1
1432                 bld     temp0,LEDlsb1
1433                 bst     ACC,2
1434                 bld     temp0,LEDlsb2
1435                 bst     ACC,3
1436                 bld     temp1,LEDmsb3
1437                 bst     ACC,4
1438                 bld     temp1,LEDmsb4
1439                 bst     ACC,5
1440                 bld     temp1,LEDmsb5
1441                 bst     ACC,6
1442                 bld     temp1,LEDmsb6
1443                 bst     ACC,7
1444                 bld     temp1,LEDmsb7
1445                 ret
1446 ;-----------------------------------------------------------------------------------------
1447 SetDataPortDirection:
1448                 in      temp0,LEDdirectionLSB           ;nacitaj aktualny stav LSB do temp0 (aby sa nezmenili ostatne smery bitov)
1449                 in      temp1,LEDdirectionMSB           ;nacitaj aktualny stav MSB do temp1 (aby sa nezmenili ostatne smery bitov)
1450                 rcall   MaskPortData
1451                 out     LEDdirectionLSB,temp0           ;a update smeru LSB datoveho portu
1452                 out     LEDdirectionMSB,temp1           ;a update smeru MSB datoveho portu
1453                 ret
1454 ;-----------------------------------------------------------------------------------------
1455 SetOutDataPort:
1456                 in      temp0,LEDPortLSB                ;nacitaj aktualny stav LSB do temp0 (aby sa nezmenili ostatne bity)
1457                 in      temp1,LEDPortMSB                ;nacitaj aktualny stav MSB do temp1 (aby sa nezmenili ostatne bity)
1458                 rcall   MaskPortData
1459                 out     LEDPortLSB,temp0                ;a update LSB datoveho portu
1460                 out     LEDPortMSB,temp1                ;a update MSB datoveho portu
1461                 ret
1462 ;-----------------------------------------------------------------------------------------
1463 GetInDataPort:
1464                 in      temp0,LEDPinMSB                 ;nacitaj aktualny stav MSB do temp0
1465                 in      temp1,LEDPinLSB                 ;nacitaj aktualny stav LSB do temp1
1466 MoveLEDin:
1467                 bst     temp1,LEDlsb0                   ;a daj bity LSB na spravne pozicie (z temp1 do temp0)
1468                 bld     temp0,0                         ;(bity MSB su na spravnom mieste)
1469                 bst     temp1,LEDlsb1
1470                 bld     temp0,1
1471                 bst     temp1,LEDlsb2
1472                 bld     temp0,2
1473                 mov     R0,temp0                        ;a vysledok uloz do R0
1474                 ret
1475 ;-----------------------------------------------------------------------------------------
1476 GetOutDataPort:
1477                 in      temp0,LEDPortMSB                ;nacitaj aktualny stav MSB do temp0
1478                 in      temp1,LEDPortLSB                ;nacitaj aktualny stav LSB do temp1
1479                 rjmp    MoveLEDin
1480 ;-----------------------------------------------------------------------------------------
1481 GetDataPortDirection:
1482                 in      temp0,LEDdirectionMSB           ;nacitaj aktualny stav MSB do temp0
1483                 in      temp1,LEDdirectionLSB           ;nacitaj aktualny stav LSB do temp1
1484                 rjmp    MoveLEDin
1485 ;-----------------------------------------------------------------------------------------
1486 EEPROMWrite:
1487                 out     EEAR,ZL                         ;nastav adresu EEPROM
1488                 out     EEDR,R0                         ;nastav data do EEPROM
1489                 cli                                     ;zakaz prerusenie
1490                 sbi     EECR,EEMWE                      ;nastav master write enable
1491                 sei                                     ;povol prerusenie (este sa vykona nasledujuca instrukcia)
1492                 sbi     EECR,EEWE                       ;samotny zapis
1493 WaitForEEPROMReady:
1494                 sbic    EECR,EEWE                       ;pockaj si na koniec zapisu
1495                 rjmp    WaitForEEPROMReady              ;v slucke (max cca 4ms) (kvoli naslednemu citaniu/zapisu)
1496                 rjmp    OneZeroAnswer                   ;potvrd prijem jednou nulou
1497 ;-----------------------------------------------------------------------------------------
1498 ;*****************************************************************
1499 ;* End of Program
1500 ;*****************************************************************
1501 ;-----------------------------------------------------------------------------------------
1502 ;*****************************************************************
1503 ;* EEPROM contents
1504 ;*****************************************************************
1505 ;-----------------------------------------------------------------------------------------
1506 .eseg           ;data v EEPROM-ke (vo finalnej verzii zapoznamkovat)
1507 ;.org   0x400   ;pre naplnenie EEPROM dat na spravne adresy - hned za kod programu (vo finalnej verzii odpoznamkovat)
1508 EEData:
1509                 .db     "This device was developed by Ing. Igor Cesko: cesko@internet.sk "
1510                 .db     "For more information see: http://www.cesko.host.sk. "
1511                 .db     "S/N:00000001"
1512 ;-----------------------------------------------------------------------------------------
1513 ;*****************************************************************
1514 ;* End of file
1515 ;*****************************************************************
```