

```

1 ;*****
2 ;* U S B   S T A C K   F O R   T H E   A V R   F A M I L Y
3 ;*
4 ;* File Name           : "USB90S2313.asm"
5 ;* Title               : AVR309:USB to UART protocol converter (simple - small FIFO)
6 ;* Date               : 26.01.2004
7 ;* Version            : 2.2
8 ;* Target MCU         : AT90S2313-10
9 ;* AUTHOR             : Ing. Igor Cesko
10 ;*                   : Slovakia
11 ;*                   : cesko@internet.sk
12 ;*                   : http://www.cesko.host.sk
13 ;*
14 ;* DESCRIPTION:
15 ;* USB protocol implementation into MCU with noUSB interface:
16 ;* Device:
17 ;* Universal USB interface (8-bit I/O port + RS232 serial line + EEPROM)
18 ;* + added RS232 FIFO buffer
19 ;*
20 ;* The timing is adapted for 12 MHz crystal (overclocked MCU !!!)
21 ;*
22 ;*
23 ;* to add your own functions - see section: TEMPLATE OF YOUR FUNCTION
24 ;*
25 ;* to customize device to your company you must change VendorUSB ID (VID)
26 ;* to VID assigned to your company (for more information see www.usb.org)
27 ;*
28 ;*****
29 .include "2313def.inc"
30
31 .equ    inputport      =PINB
32 .equ    outputport     =PORTB
33 .equ    USBdirection   =DDRB
34 .equ    DATApplus      =1           ;signal D+ na PB1
35 .equ    DATAminus     =0           ;signal D- na PB0 - treba dat na tento pin pull-up 1.5kOhm
36 .equ    USBpinmask     =0b11111100 ;mask low 2 bits (D+,D-) on PB
37 .equ    USBpinmaskDplus = ~(1<<DATApplus) ;mask D+ bit on PB
38 .equ    USBpinmaskDminus = ~(1<<DATAminus) ;mask D- bit on PB
39
40 .equ    TSOPPort       =PINB
41 .equ    TSOPpullupPort =PORTB
42 .equ    TSOPPin        =2           ;signal OUT z IR senzora TSOP1738 na PB2
43
44 .equ    LEDPortLSB     =PORTD       ;pripojenie LED diod LSB
45 .equ    LEDPinLSB      =PIND        ;pripojenie LED diod LSB (vstup)
46 .equ    LEDdirectionLSB =DDRD      ;vstup/vystup LED LSB
47 .equ    LEDPortMSB     =PORTB      ;pripojenie LED diod MSB
48 .equ    LEDPinMSB      =PINB       ;pripojenie LED diod MSB (vstup)
49 .equ    LEDdirectionMSB =DDRB      ;vstup/vystup LED MSB
50 .equ    LEDlsb0        =3          ;LED0 na pin PD3
51 .equ    LEDlsb1        =5          ;LED1 na pin PD5
52 .equ    LEDlsb2        =6          ;LED2 na pin PD6
53 .equ    LEDmsb3        =3          ;LED3 na pin PB3
54 .equ    LEDmsb4        =4          ;LED4 na pin PB4

```

```

55 .equ LEDmsb5           =5           ;LED5 na pin PB5
56 .equ LEDmsb6           =6           ;LED6 na pin PB6
57 .equ LEDmsb7           =7           ;LED7 na pin PB7
58
59 .equ SOPbyte            =0b10000000 ;Start of Packet byte
60 .equ DATA0PID          =0b11000011 ;PID pre DATA0 pole
61 .equ DATA1PID          =0b01001011 ;PID pre DATA1 pole
62 .equ OUTPID             =0b11100001 ;PID pre OUT pole
63 .equ INPID              =0b01101001 ;PID pre IN pole
64 .equ SOFPID             =0b10100101 ;PID pre SOF pole
65 .equ SETUPPID           =0b00101101 ;PID pre SETUP pole
66 .equ ACKPID             =0b11010010 ;PID pre ACK pole
67 .equ NAKPID             =0b01011010 ;PID pre NAK pole
68 .equ STALLPID           =0b00011110 ;PID pre STALL pole
69 .equ PREPID             =0b00111100 ;PID pre PRE pole
70
71 .equ nSOPbyte            =0b00000001 ;Start of Packet byte - opacne poradie
72 .equ nDATA0PID          =0b11000011 ;PID pre DATA0 pole - opacne poradie
73 .equ nDATA1PID          =0b11010010 ;PID pre DATA1 pole - opacne poradie
74 .equ nOUTPID            =0b10000111 ;PID pre OUT pole - opacne poradie
75 .equ nINPID             =0b10010110 ;PID pre IN pole - opacne poradie
76 .equ nSOFPID            =0b10100101 ;PID pre SOF pole - opacne poradie
77 .equ nSETUPPID          =0b10110100 ;PID pre SETUP pole - opacne poradie
78 .equ nACKPID            =0b01001011 ;PID pre ACK pole - opacne poradie
79 .equ nNAKPID            =0b01011010 ;PID pre NAK pole - opacne poradie
80 .equ nSTALLPID          =0b01111000 ;PID pre STALL pole - opacne poradie
81 .equ nPREPID            =0b00111100 ;PID pre PRE pole - opacne poradie
82
83 .equ nNRZITokenPID       =~0b10000000 ;PID maska pre Token paket (IN,OUT,SOF,SETUP) - opacne poradie NRZI
84 .equ nNRZISOPbyte       =~0b01010101 ;Start of Packet byte - opacne poradie NRZI
85 .equ nNRZIDATA0PID       =~0b11010111 ;PID pre DATA0 pole - opacne poradie NRZI
86 .equ nNRZIDATA1PID       =~0b11001001 ;PID pre DATA1 pole - opacne poradie NRZI
87 .equ nNRZIOUTPID        =~0b10101111 ;PID pre OUT pole - opacne poradie NRZI
88 .equ nNRZIINPID         =~0b10110001 ;PID pre IN pole - opacne poradie NRZI
89 .equ nNRZISOFPID        =~0b10010011 ;PID pre SOF pole - opacne poradie NRZI
90 .equ nNRZISETUPPID       =~0b10001101 ;PID pre SETUP pole - opacne poradie NRZI
91 .equ nNRZIACKPID        =~0b00100111 ;PID pre ACK pole - opacne poradie NRZI
92 .equ nNRZINAKPID        =~0b00111001 ;PID pre NAK pole - opacne poradie NRZI
93 .equ nNRZISTALLPID       =~0b00000111 ;PID pre STALL pole - opacne poradie NRZI
94 .equ nNRZIPREPID        =~0b01111101 ;PID pre PRE pole - opacne poradie NRZI
95 .equ nNRZIADDR0         =~0b01010101 ;Adresa = 0 - opacne poradie NRZI
96
97                               ;stavove byty - State
98 .equ BaseState           =0           ;
99 .equ SetupState          =1           ;
100 .equ InState             =2           ;
101 .equ OutState            =3           ;
102 .equ SOFState            =4           ;
103 .equ DataState           =5           ;
104 .equ AddressChangeState  =6           ;
105
106                               ;Flagy pozadovanej akcie
107 .equ DoNone               =0
108 .equ DoReceiveOutData     =1

```

```

109 .equ DoReceiveSetupData =2
110 .equ DoPrepareOutContinuousBuffer =3
111 .equ DoReadySendAnswer =4
112
113
114 .equ CRC5poly =0b00101 ;CRC5 polynom
115 .equ CRC5zvysok =0b01100 ;CRC5 zvysok po uspesnmp CRC5
116 .equ CRC16poly =0b1000000000000101 ;CRC16 polynom
117 .equ CRC16zvysok =0b1000000000001101 ;CRC16 zvysok po uspesnom CRC16
118
119 .equ MAXUSBBYTES =14 ;maximum bytes in USB input message
120 .equ MAXRS232LENGTH =36 ;maximalna dlzka RS232 kodu (pocet jednotiek a nul spolu) (pozor: MAXRS232LENGTH musi byt parne
cislo !!!)
121 .equ NumberOfFirstBits =10 ;kolko prvych bitov moze byt dlhsich
122 .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)
123 .equ InitBaudRate =12000000/16/57600-1 ;nastavit vysielaciu rychlost UART-u na 57600 (pre 12MHz=12000000Hz)
124
125 .equ InputBufferBegin =RAMEND-127 ;zaciatok prijimacieho shift buffera
126 .equ InputShiftBufferBegin =InputBufferBegin+MAXUSBBYTES ;zaciatok prijimacieho buffera
127 .equ RS232BufferBegin =InputShiftBufferBegin+MAXUSBBYTES ;zaciatok buffera pre RS232 prijem
128
129 .equ MyInAddressSRAM =RS232BufferBegin+MAXRS232LENGTH+1
130 .equ MyOutAddressSRAM =MyInAddressSRAM+1
131
132 .equ OutputBufferBegin =RAMEND-MAXUSBBYTES-2 ;zaciatok vysielacieho buffera
133 .equ AckBufferBegin =OutputBufferBegin-3 ;zaciatok vysielacieho buffera Ack
134 .equ NakBufferBegin =AckBufferBegin-3 ;zaciatok vysielacieho buffera Nak
135
136 .equ StackBegin =NakBufferBegin-1 ;spodok zasobnika
137
138 .def ConfigByte =R1 ;0=unconfigured state
139 .def backupbitcount =R2 ;zaloha bitcount registra v INT0 preruseni
140 .def RAMread =R3 ;ci sa ma citat zo SRAM-ky
141 .def backupSREGTimer =R4 ;zaloha Flag registra v Timer interrupte
142 .def backupSREG =R5 ;zaloha Flag registra v INT0 preruseni
143 .def ACC =R6 ;accumulator
144 .def lastBitstuffNumber =R7 ;pozicia bitstuffingu
145 .def OutBitStuffNumber =R8 ;kolko bitov sa ma este odvysielat z posledneho bytu - bitstuffing
146 .def BitStuffInOut =R9 ;ci sa ma vkladat alebo mazat bitstuffing
147 .def TotalBytesToSend =R10 ;kolko sa ma poslat bytov
148 .def TransmitPart =R11 ;poradove cislo vysielacej casti
149 .def InputBufferLength =R12 ;dlzka pripravena vo vstupnom USB bufferi
150 .def OutputBufferLength =R13 ;dlzka odpovede pripravena v USB bufferi
151 .def MyOutAddress =R14 ;moja USB adresa na update
152 .def MyInAddress =R15 ;moja USB adresa
153
154
155 .def ActionFlag =R16 ;co sa ma urobit v hlavnej slucke programu
156 .def temp3 =R17 ;temporary register
157 .def temp2 =R18 ;temporary register
158 .def temp1 =R19 ;temporary register
159 .def temp0 =R20 ;temporary register
160 .def bitcount =R21 ;counter of bits in byte

```

```

161 .def      ByteCount      =R22      ;pocitadlo maximalneho poctu prijatych bajtov
162 .def      inputbuf      =R23      ;prijimaci register
163 .def      shiftbuf      =R24      ;posuvny prijimaci register
164 .def      State          =R25      ;byte stavu stavoveho stroja
165 .def      RS232BufptrX   =R26      ;XL register - pointer do buffera prijatych IR kodov
166 .def      RS232BufferFull =R27      ;XH register - priznak plneho RS232 Buffera
167 .def      USBBufptrY     =R28      ;YL register - pointer do USB buffera input/output
168 .def      ROMBufptrZ     =R30      ;ZL register - pointer do buffera ROM dat
169
170 ;poziadavky na deskripty
171 .equ      GET_STATUS      =0
172 .equ      CLEAR_FEATURE   =1
173 .equ      SET_FEATURE     =3
174 .equ      SET_ADDRESS     =5
175 .equ      GET_DESCRIPTOR  =6
176 .equ      SET_DESCRIPTOR  =7
177 .equ      GET_CONFIGURATION =8
178 .equ      SET_CONFIGURATION =9
179 .equ      GET_INTERFACE   =10
180 .equ      SET_INTERFACE   =11
181 .equ      SYNCH_FRAME     =12
182
183 ;typy deskriptorov
184 .equ      DEVICE           =1
185 .equ      CONFIGURATION    =2
186 .equ      STRING           =3
187 .equ      INTERFACE        =4
188 .equ      ENDPOINT         =5
189
190 .equ      USER_FNC_NUMBER  =100
191
192
193 ;-----
194 ;*****
195 ;* Interrupt table
196 ;*****
197 .cseg
198 ;-----
199 .org 0                                ;po resete
200                                rjmp     reset
201
202 ;-----
203 .org INT0addr                        ;externe prerusenie INT0
204                                rjmp     INT0handler
205
206 ;-----
207 .org URXCaddr                        ;prijem zo seriovej linky
208                                push     temp0
209                                in        temp0,UDR                ;nacistaj do temp0 prijate data z UART-u
210                                sei                          ;povol interrupty na obsluhu USB
211                                in        backupSREGTimer,SREG      ;zaloha SREG
212                                cbi       UCR,RXCIE                ;zakazat interrupt od prijimania UART
213                                cpi       RS232BufferFull,MAXRS232LENGTH-4
214                                brcc      NoIncRS232BufferFull
215                                push     RS232BufptrX
216                                lds      RS232BufptrX,RS232BufferBegin+2 ;nastavenie sa na zaciatok buffera zapisu RS232 kodu : 3.byte hlavicky (dlzka kodu + citanie + zapis)
217                                stb      RS232BufptrX,RS232BufferFull
218                                pop      RS232BufptrX
219                                rjmp     RS232BufferFull
220
221 ;-----
222 .org 0
223
224 ;-----
225 .org 0
226
227 ;-----
228 .org 0
229
230 ;-----
231 .org 0
232
233 ;-----
234 .org 0
235
236 ;-----
237 .org 0
238
239 ;-----
240 .org 0
241
242 ;-----
243 .org 0
244
245 ;-----
246 .org 0
247
248 ;-----
249 .org 0
250
251 ;-----
252 .org 0
253
254 ;-----
255 .org 0
256
257 ;-----
258 .org 0
259
260 ;-----
261 .org 0
262
263 ;-----
264 .org 0
265
266 ;-----
267 .org 0
268
269 ;-----
270 .org 0
271
272 ;-----
273 .org 0
274
275 ;-----
276 .org 0
277
278 ;-----
279 .org 0
280
281 ;-----
282 .org 0
283
284 ;-----
285 .org 0
286
287 ;-----
288 .org 0
289
290 ;-----
291 .org 0
292
293 ;-----
294 .org 0
295
296 ;-----
297 .org 0
298
299 ;-----
300 .org 0
301
302 ;-----
303 .org 0
304
305 ;-----
306 .org 0
307
308 ;-----
309 .org 0
310
311 ;-----
312 .org 0
313
314 ;-----
315 .org 0
316
317 ;-----
318 .org 0
319
320 ;-----
321 .org 0
322
323 ;-----
324 .org 0
325
326 ;-----
327 .org 0
328
329 ;-----
330 .org 0
331
332 ;-----
333 .org 0
334
335 ;-----
336 .org 0
337
338 ;-----
339 .org 0
340
341 ;-----
342 .org 0
343
344 ;-----
345 .org 0
346
347 ;-----
348 .org 0
349
350 ;-----
351 .org 0
352
353 ;-----
354 .org 0
355
356 ;-----
357 .org 0
358
359 ;-----
360 .org 0
361
362 ;-----
363 .org 0
364
365 ;-----
366 .org 0
367
368 ;-----
369 .org 0
370
371 ;-----
372 .org 0
373
374 ;-----
375 .org 0
376
377 ;-----
378 .org 0
379
380 ;-----
381 .org 0
382
383 ;-----
384 .org 0
385
386 ;-----
387 .org 0
388
389 ;-----
390 .org 0
391
392 ;-----
393 .org 0
394
395 ;-----
396 .org 0
397
398 ;-----
399 .org 0
400
401 ;-----
402 .org 0
403
404 ;-----
405 .org 0
406
407 ;-----
408 .org 0
409
410 ;-----
411 .org 0
412
413 ;-----
414 .org 0
415
416 ;-----
417 .org 0
418
419 ;-----
420 .org 0
421
422 ;-----
423 .org 0
424
425 ;-----
426 .org 0
427
428 ;-----
429 .org 0
430
431 ;-----
432 .org 0
433
434 ;-----
435 .org 0
436
437 ;-----
438 .org 0
439
440 ;-----
441 .org 0
442
443 ;-----
444 .org 0
445
446 ;-----
447 .org 0
448
449 ;-----
450 .org 0
451
452 ;-----
453 .org 0
454
455 ;-----
456 .org 0
457
458 ;-----
459 .org 0
460
461 ;-----
462 .org 0
463
464 ;-----
465 .org 0
466
467 ;-----
468 .org 0
469
470 ;-----
471 .org 0
472
473 ;-----
474 .org 0
475
476 ;-----
477 .org 0
478
479 ;-----
480 .org 0
481
482 ;-----
483 .org 0
484
485 ;-----
486 .org 0
487
488 ;-----
489 .org 0
490
491 ;-----
492 .org 0
493
494 ;-----
495 .org 0
496
497 ;-----
498 .org 0
499
500 ;-----
501 .org 0
502
503 ;-----
504 .org 0
505
506 ;-----
507 .org 0
508
509 ;-----
510 .org 0
511
512 ;-----
513 .org 0
514
515 ;-----
516 .org 0
517
518 ;-----
519 .org 0
520
521 ;-----
522 .org 0
523
524 ;-----
525 .org 0
526
527 ;-----
528 .org 0
529
530 ;-----
531 .org 0
532
533 ;-----
534 .org 0
535
536 ;-----
537 .org 0
538
539 ;-----
540 .org 0
541
542 ;-----
543 .org 0
544
545 ;-----
546 .org 0
547
548 ;-----
549 .org 0
550
551 ;-----
552 .org 0
553
554 ;-----
555 .org 0
556
557 ;-----
558 .org 0
559
560 ;-----
561 .org 0
562
563 ;-----
564 .org 0
565
566 ;-----
567 .org 0
568
569 ;-----
570 .org 0
571
572 ;-----
573 .org 0
574
575 ;-----
576 .org 0
577
578 ;-----
579 .org 0
580
581 ;-----
582 .org 0
583
584 ;-----
585 .org 0
586
587 ;-----
588 .org 0
589
590 ;-----
591 .org 0
592
593 ;-----
594 .org 0
595
596 ;-----
597 .org 0
598
599 ;-----
600 .org 0
601
602 ;-----
603 .org 0
604
605 ;-----
606 .org 0
607
608 ;-----
609 .org 0
610
611 ;-----
612 .org 0
613
614 ;-----
615 .org 0
616
617 ;-----
618 .org 0
619
620 ;-----
621 .org 0
622
623 ;-----
624 .org 0
625
626 ;-----
627 .org 0
628
629 ;-----
630 .org 0
631
632 ;-----
633 .org 0
634
635 ;-----
636 .org 0
637
638 ;-----
639 .org 0
640
641 ;-----
642 .org 0
643
644 ;-----
645 .org 0
646
647 ;-----
648 .org 0
649
650 ;-----
651 .org 0
652
653 ;-----
654 .org 0
655
656 ;-----
657 .org 0
658
659 ;-----
660 .org 0
661
662 ;-----
663 .org 0
664
665 ;-----
666 .org 0
667
668 ;-----
669 .org 0
670
671 ;-----
672 .org 0
673
674 ;-----
675 .org 0
676
677 ;-----
678 .org 0
679
680 ;-----
681 .org 0
682
683 ;-----
684 .org 0
685
686 ;-----
687 .org 0
688
689 ;-----
690 .org 0
691
692 ;-----
693 .org 0
694
695 ;-----
696 .org 0
697
698 ;-----
699 .org 0
700
701 ;-----
702 .org 0
703
704 ;-----
705 .org 0
706
707 ;-----
708 .org 0
709
710 ;-----
711 .org 0
712
713 ;-----
714 .org 0
715
716 ;-----
717 .org 0
718
719 ;-----
720 .org 0
721
722 ;-----
723 .org 0
724
725 ;-----
726 .org 0
727
728 ;-----
729 .org 0
730
731 ;-----
732 .org 0
733
734 ;-----
735 .org 0
736
737 ;-----
738 .org 0
739
740 ;-----
741 .org 0
742
743 ;-----
744 .org 0
745
746 ;-----
747 .org 0
748
749 ;-----
750 .org 0
751
752 ;-----
753 .org 0
754
755 ;-----
756 .org 0
757
758 ;-----
75
```

```

+ rezerva)
215      st      X+,temp0                ;a uloz ho do buffera
216      cpi     RS232BufptrX,RS232BufferBegin+MAXRS232LENGTH+1 ;ak sa nedosiahol maximum RS232 buffera
217      brne    NoUARTBufferOverflow    ;tak pokracuj
218      ldi     RS232BufptrX,RS232BufferBegin+4 ;inak sa nastav na zaciatok buffera
219      NoUARTBufferOverflow:
220      sts     RS232BufferBegin+2,RS232BufptrX ;ulozenie noveho offsetu buffera zapisu RS232 kodu : 3.byte hlavicky (dlzka kodu + citanie + zapis +
rezerva)
221      inc     RS232BufferFull          ;zvys dlzku RS232 Buffera
222      pop     RS232BufptrX
223      NoIncRS232BufferFull:
224      pop     temp0
225      out     SREG,backupSREGTimer     ;obnova SREG
226      cli     ;zakazat interrupt kvoli zacykleniu
227      sbi     UCR,RXCIE               ;povolit interrupt od prijimania UART
228      reti
229      ;-----
230      ;*****
231      ;* Init program
232      ;*****
233      ;-----
234      reset:      ;inicializacia procesora a premennych na spravne hodnoty
235      ldi     temp0,StackBegin          ;inicializacia stacku
236      out     SPL,temp0
237
238      clr     XH                        ;RS232 pointer
239      clr     YH                        ;USB pointer
240      clr     ZH                        ;ROM pointer
241      sts     RS232BufferBegin+0,YH     ;znuluj dlzky RS232 kodu v bufferi
242      ldi     temp0,RS232BufferBegin+4
243      sts     RS232BufferBegin+1,temp0;znuluj ukazovatel citania
244      sts     RS232BufferBegin+2,temp0;znuluj ukazovatel zapisu
245      clr     RS232BufferFull
246
247      rcall   InitACKBuffer             ;inicializacia ACK buffera
248      rcall   InitNAKBuffer             ;inicializacia NAK buffera
249
250      rcall   USBReset                  ;inicializacia USB adresy
251
252      sbi     TSOPpullupPort,TSOPpin    ;nahodit pull-up na TSOP vstupe
253
254      ldi     temp0,(1<<LEDlsb0)+(1<<LEDlsb1)+(1<<LEDlsb2)
255      out     LEDPortLSB,temp0          ;nahodit pull-up na vsetkych LED vstupoch LSB
256      ldi     temp0,(1<<LEDmsb3)+(1<<LEDmsb4)+(1<<LEDmsb5)+(1<<LEDmsb6)+(1<<LEDmsb7)
257      out     LEDPortMSB,temp0          ;nahodit pull-up na vsetkych LED vstupoch MSB
258
259      sbi     PORTD,0                   ;nahodit pull-up na RxD vstupe
260      ldi     temp0,InitBaudRate        ;nastavitvysielaci rychlost UART-u
261      out     UBRR,temp0
262      sbi     UCR,TXEN                  ;povolit vysielanie UART-u
263      sbi     UCR,RXEN                  ;povolit prijimanie UART-u
264      sbi     UCR,RXCIE                 ;povolit interrupt od prijimania UART
265
266      ldi     temp0,0x0F                ;INT0 - reagovanie na nabeznu hranu

```

```

267         out    MCUCR,temp0          ;
268         ldi     temp0,1<<INT0       ;povolit externy interrupt INT0
269         out     GIMSK,temp0
270 ;-----
271 ;*****
272 ;* Main program
273 ;*****
274         sei                      ;povolit interrupty globalne
275 Main:
276         sbis    inputport,DATAMinus  ;cakanie az sa zmeni D- na 0
277         rjmp    CheckUSBReset        ;a skontroluj, ci to nie je USB reset
278
279         cpi     ActionFlag,DoReceiveSetupData
280         breq    ProcReceiveSetupData
281         cpi     ActionFlag,DoPrepareOutContinuousBuffer
282         breq    ProcPrepareOutContinuousBuffer
283         rjmp    Main
284
285 CheckUSBReset:
286         ldi     temp0,255            ;pocitadlo trvania reset-u (podla normy je to cca 10ms - tu je to cca 100us)
287 WaitForUSBReset:
288         sbic    inputport,DATAMinus  ;cakanie az sa zmeni D+ na 0
289         rjmp    Main
290         dec     temp0
291         brne    WaitForUSBReset
292         rcall   USBReset
293         rjmp    Main
294
295 ProcPrepareOutContinuousBuffer:
296         rcall   PrepareOutContinuousBuffer ;priprav pokracovanie odpovede do buffera
297         ldi     ActionFlag,DoReadySendAnswer
298         rjmp    Main
299 ProcReceiveSetupData:
300         ldi     USBBufptrY,InputBufferBegin ;pointer na zaciatok prijimacieho buffera
301         mov     ByteCount,InputBufferLength ;dlzka vstupneho buffera
302         rcall   DecodeNRZI                ;prevod kodovania NRZI na bity
303         rcall   MirrorInBufferBytes        ;prehodit poradie bitov v bajtoch
304         rcall   BitStuff                   ;odstranenie bit stuffing
305         ;rcall   CheckCRCIn                ;kontrola CRC
306         rcall   PrepareUSBOutAnswer        ;pripravenie odpovede do vysielacieho buffera
307         ldi     ActionFlag,DoReadySendAnswer
308         rjmp    Main
309 ;*****
310 ;* Main program END
311 ;*****
312 ;-----
313 ;*****
314 ;* Interrupt0 interrupt handler
315 ;*****
316 INT0Handler:
317         in      backupSREG,SREG          ;prerusenie INT0
318         push    temp0
319         push    temp1
320

```

```

321         ldi     temp0,3           ;pocitadlo trvania log0
322         ldi     temp1,2           ;pocitadlo trvania log1
323         ;cakanie na zaciatok paketu
324 CheckchangeMinus:
325         sbis    inputport,DATAMinus ;cakanie az sa zmeni D- na 1
326         rjmp    CheckchangeMinus
327 CheckchangePlus:
328         sbis    inputport,DATApus  ;cakanie az sa zmeni D+ na 1
329         rjmp    CheckchangePlus
330 DetectSOPEnd:
331         sbis    inputport,DATApus
332         rjmp    Increment0        ;D+ =0
333 Increment1:
334         ldi     temp0,3           ;pocitadlo trvania log0
335         dec     temp1             ;kolko cyklov trvala log1
336         nop
337         breq    USBBeginPacket    ;ak je to koniec SOP - prijimaj paket
338         rjmp    DetectSOPEnd
339 Increment0:
340         ldi     temp1,2           ;pocitadlo trvania log1
341         dec     temp0             ;kolko cyklov trvala log0
342         nop
343         brne    DetectSOPEnd       ;ak nenastal SOF - pokracuj
344         rjmp    EndInt0HandlerPOP2
345 EndInt0Handler:
346         pop     ACC
347         pop     RS232BufptrX
348         pop     temp3
349         pop     temp2
350 EndInt0HandlerPOP:
351         pop     USBBufptrY
352         pop     ByteCount
353         mov     bitcount,backupbitcount ;obnova bitcount registra
354 EndInt0HandlerPOP2:
355         pop     temp1
356         pop     temp0
357         out     SREG,backupsREG
358         ldi     shiftbuf,1<<INTF0 ;znulovat flag interruptu INTF0
359         out     GIFR,shiftbuf
360         reti    ;inak skonci (bol iba SOF - kazdu milisekundu)
361
362 USBBeginPacket:
363         mov     backupbitcount,bitcount ;zaloha bitcount registra
364         in      shiftbuf,inputport    ;ak ano nacistaj ho ako nuly bit priamo do shift registra
365 USBloopBegin:
366         push    ByteCount             ;dalsia zaloha registrov (setrenie casu)
367         push    USBBufptrY
368         ldi     bitcount,6           ;inicializacia pocitadla bitov v bajte
369         ldi     ByteCount,MAXUSBBYTES ;inicializacia max poctu prijatych bajtov v pakete
370         ldi     USBBufptrY,InputShiftBufferBegin ;nastav vstupny buffer
371 USBloop1_6:
372         in      inputbuf,inputport
373         cbr     inputbuf,USBpinmask  ;odmaskovat spodne 2 bity
374         breq    USBloopEnd           ;ak su nulove - koniec USB packetu

```

```

375      ror      inputbuf      ;presun Data+ do shift registra
376      rol      shiftbuf
377      dec      bitcount      ;zmensi pocitadlo bitov
378      brne     USBloop1_6     ;ak nie je nulove - opakuj naplnanie shift registra
379      nop
380 USBloop7:
381      in        inputbuf,inputport
382      cbr       inputbuf,USBpinmask      ;odmaskovat spodne 2 bity
383      breq      USBloopEnd      ;ak su nulove - koniec USB packetu
384      ror      inputbuf      ;presun Data+ do shift registra
385      rol      shiftbuf
386      ldi      bitcount,7      ;inicializacia pocitadla bitov v bajte
387      st        Y+,shiftbuf      ;skopiruj shift register bo buffera a zvyjs pointer do buffera
388 USBloop0:
389      in        shiftbuf,inputport      ;nulty bit priamo do shift registra
390      cbr       shiftbuf,USBpinmask      ;odmaskovat spodne 2 bity
391      breq      USBloopEnd      ;ak su nulove - koniec USB packetu
392      dec      bitcount      ;zmensi pocitadlo bitov
393      nop
394      dec      ByteCount      ;ak sa nedosiahol maximum buffera
395      brne     USBloop1_6     ;tak prijimaj dalej
396
397      rjmp     EndInt0HandlerPOP      ;inak opakuj od zaciatku
398
399 USBloopEnd:
400      cpi      USBBufptrY,InputShiftBufferBegin+3      ;ak sa neprijali aspon 3 byte
401      brcs     EndInt0HandlerPOP      ;tak skonci
402      lds      temp0,InputShiftBufferBegin+0      ;identifikator packetu do temp0
403      lds      temp1,InputShiftBufferBegin+1      ;adresa do temp1
404      brne     TestDataPacket      ;ak je dlzka ina ako 3 - tak to moze byt iba DataPacket
405 TestIOPacket:
406      ;      cp      temp1,MyInAddress      ;ak to nie je urcene (adresa) pre mna
407      ;      brne     TestDataPacket      ;tak to moze byt este Data Packet
408 TestSetupPacket: ;test na SETUP paket
409      cpi      temp0,nNRZISETUPPID
410      brne     TestOutPacket      ;ak nie je Setup PID - dekoduj iny paket
411      cp       temp1,MyInAddress      ;ak to nie je urcene (adresa) pre mna      ;ENG;if this isn't assigned (address) for me
412      brne     TestDataPacket      ;tak to moze byt este Data Packet      ;ENG;then this can be still DataPacket
413      ldi      State,SetupState
414      rjmp     EndInt0HandlerPOP      ;ak je Setup PID - prijimaj nasledny Data paket
415 TestOutPacket: ;test na OUT paket
416      cpi      temp0,nNRZIOUTPID
417      brne     TestInPacket      ;ak nie je Out PID - dekoduj iny paket
418      cp       temp1,MyOutAddress      ;ak to nie je urcene (adresa) pre mna      ;ENG;if this isn't assigned (address) for me
419      brne     TestDataPacket      ;tak to moze byt este Data Packet      ;ENG;then this can be still DataPacket
420      ldi      State,OutState
421      rjmp     EndInt0HandlerPOP      ;ak je Out PID - prijimaj nasledny Data paket
422 TestInPacket: ;test na IN paket
423      cpi      temp0,nNRZIINPID
424      brne     TestDataPacket      ;ak nie je In PID - dekoduj iny paket
425      cp       temp1,MyInAddress      ;ak to nie je urcene (adresa) pre mna      ;ENG;if this isn't assigned (address) for me
426      breq     AnswerToInRequest      ;tak to moze byt este Data Packet      ;ENG;then this can be still DataPacket
427 TestDataPacket: ;test na DATA0 a DATA1 paket
428      cpi      temp0,nNRZIDATA0PID

```



```

429         breq    Data0Packet          ;ak nie je Data0 PID - dekoduj iny paket
430         cpi     temp0,nNRZIDATA1PID
431         brne    NoMyPacked           ;ak nie je Data1 PID - dekoduj iny paket
432 Data0Packet:
433         cpi     State,SetupState      ;ak bol stav Setup
434         breq    ReceiveSetupData      ;prijmi ho
435         cpi     State,OutState        ;ak bol stav Out
436         breq    ReceiveOutData        ;prijmi ho
437 NoMyPacked:
438         ldi     State,BaseState       ;znuluj stav
439         rjmp    EndInt0HandlerPOP     ;a prijimaj nasledny Data paket
440
441 AnswerToInRequest:
442         push    temp2                 ;zazalohuj dalsie registre a pokracuj
443         push    temp3
444         push    RS232BufptrX
445         push    ACC
446         cpi     ActionFlag,DoReadySendAnswer ;ak nie je pripravena odpoved
447         brne    NoReadySend           ;tak posli NAK
448         rcall   SendPreparedUSBAnswer ;poslanie odpovede naspat
449         cpi     State,AddressChangeState;ak je stav AddressChange
450         breq    SetMyNewUSBAddress    ;tak treba zmenit USB adresu
451         ldi     State,InState
452         ldi     ActionFlag,DoPrepareOutContinuousBuffer
453         rjmp    EndInt0Handler        ;a opakuje - caka na dalsiu odozvu z USB
454 ReceiveSetupData:
455         push    temp2                 ;zazalohuj dalsie registre a pokracuj
456         push    temp3
457         push    RS232BufptrX
458         push    ACC
459         rcall   SendACK                ;akceptovanie Setup Data paketu
460         rcall   FinishReceiving        ;ukonci prijem
461         ldi     ActionFlag,DoReceiveSetupData
462         rjmp    EndInt0Handler
463 ReceiveOutData:
464         push    temp2                 ;zazalohuj dalsie registre a pokracuj
465         push    temp3
466         push    RS232BufptrX
467         push    ACC
468         cpi     ActionFlag,DoReceiveSetupData ;ak sa prave spracovava prikaz Setup
469         breq    NoReadySend           ;tak posli NAK
470         rcall   SendACK                ;akceptovanie Out paketu
471         clr     ActionFlag
472         rjmp    EndInt0Handler
473 NoReadySend:
474         rcall   SendNAK                ;este nie som pripraveny s odpovedou
475         rjmp    EndInt0Handler        ;a opakuje - caka na dalsiu odozvu z USB
476 ;-----
477 SetMyNewUSBAddress: ;nastavi novu USB adresu v NRZI kodovani
478         lds     MyInAddress,MyInAddressSRAM
479         lds     MyOutAddress,MyOutAddressSRAM
480         rjmp    EndInt0Handler
481 ;-----
482 FinishReceiving: ;korekcne akcie na ukoncenie prijmu

```

```

483         cpi      bitcount,7           ;prenes do buffera aj posledny necely byte
484         breq     NoRemainingBits      ;ak boli vsetky byty prenesene, tak neprenasaj nic
485         inc      bitcount
486 ShiftRemainingBits:
487         rol      shiftbuf             ;posun ostavajuce necele bity na spravnú pozíciu
488         dec      bitcount
489         brne     ShiftRemainingBits
490         st       Y+,shiftbuf          ;a skopiruj shift register bo buffera - necely byte
491 NoRemainingBits:
492         mov      ByteCount,USBBufptrY
493         subi     ByteCount,InputShiftBufferBegin-1      ;v ByteCount je pocet prijatych byte (vratane necelych byte)
494
495         mov      InputBufferLength,ByteCount             ;a uchovat pre pouzitie v hlavnom programe
496         ldi      USBBufptrY,InputShiftBufferBegin       ;pointer na zaciatok prijimacieho shift buffera
497         ldi      RS232BufptrX,InputBufferBegin+1        ;data buffer (vynechat SOP)
498 MoveDataBuffer:
499         ld       temp0,Y+
500         st       X+,temp0
501         dec      ByteCount
502         brne     MoveDataBuffer
503
504         ldi      ByteCount,nNRZISOPbyte
505         sts      InputBufferBegin,ByteCount              ;ako keby sa prijal SOP - nekopiruje sa zo shift buffera
506         ret
507 ;-----
508 ;*****
509 ;* Other procedures
510 ;*****
511 ;-----
512 USBReset:                ;inicializacia USB stavoveho stroja
513         ldi      temp0,nNRZIADDR0          ;inicializacia USB adresy
514         mov      MyOutAddress,temp0
515         mov      MyInAddress,temp0
516         clr      State                     ;inicializacia stavoveho stroja
517         clr      BitStuffInOut
518         clr      OutBitStuffNumber
519         clr      ActionFlag
520         clr      RAMread                   ;bude sa vycitavat z ROM-ky
521         clr      ConfigByte                ;nenakonfiguravany stav
522         ret
523 ;-----
524 SendPreparedUSBAnswer:   ;poslanie kodovanim NRZI OUT buffer s dlzkou OutputBufferLength do USB
525         mov      ByteCount,OutputBufferLength          ;dlzka odpovede
526 SendUSBAnswer:           ;poslanie kodovanim NRZI OUT buffer do USB
527         ldi      USBBufptrY,OutputBufferBegin          ;pointer na zaciatok vysielacieho buffera
528 SendUSBBuffer:           ;poslanie kodovanim NRZI dany buffer do USB
529         ldi      temp1,0                               ;zvyšovanie pointra (pomocna premenna)
530         mov      temp3,ByteCount                       ;pocitadlo bytov: temp3 = ByteCount
531         ldi      temp2,0b00000011                     ;maska na xorovanie
532         ld       inputbuf,Y+                           ;nacitanie prveho bytu do inputbuf a zvyš pointer do buffera
533                                     ;USB ako vystup:
534         cbi      outputport,DATApplus                  ;zhodenie DATApplus : kludovy stav portu USB
535         sbi      outputport,DATAMinus                   ;nahodenie DATAMinus : kludovy stav portu USB
536         sbi      USBdirection,DATApplus                 ;DATApplus ako vystupny

```

```

537         sbi      USBdirection,DATAMinus    ;DATAMinus ako vystupny
538
539         in        temp0,outputport          ;kludovy stav portu USB do temp0
540 SendUSBAnswerLoop:
541         ldi      bitcount,7                ;pocitadlo bitov
542 SendUSBAnswerByteLoop:
543         nop
544         ror      inputbuf                  ;oneskorenie kvoli casovaniu
545         brcs     NoXORSend                  ;do carry vysielany bit (v smere naskor LSB a potom MSB)
546         eor      temp0,temp2              ;ak je jedna - nemen stav na USB
547         NoXORSend:
548         out      outputport,temp0          ;vysli von na USB
549         dec      bitcount                  ;zmeni pocitadlo bitov - podla carry flagu
550         brne     SendUSBAnswerByteLoop     ;ak pocitadlo bitov nie je nulove - opakuj vysielanie s dalsim bitom
551         sbrs     inputbuf,0               ;ak je vysielany bit jedna - nemen stav na USB
552         eor      temp0,temp2              ;inak sa bude stav menit
553 NoXORSendLSB:
554         dec      temp3                    ;zniz pocitadlo bytov
555         ld       inputbuf,Y+              ;nacitanie dalsieho bytu a zvyš pointer do buffera
556         out      outputport,temp0          ;vysli von na USB
557         brne     SendUSBAnswerLoop         ;opakuj pre cely buffer (pokial temp3=0)
558
559         mov      bitcount,OutBitStuffNumber ;pocitadlo bitov pre bitstuff
560         cpi      bitcount,0               ;ak nie je potrebný bitstuff
561         breq     ZeroBitStuf
562 SendUSBAnswerBitstuffLoop:
563         ror      inputbuf                  ;do carry vysielany bit (v smere naskor LSB a potom MSB)
564         brcs     NoXORBitstuffSend         ;ak je jedna - nemen stav na USB
565         eor      temp0,temp2              ;inak sa bude stav menit
566 NoXORBitstuffSend:
567         out      outputport,temp0          ;vysli von na USB
568         nop
569         dec      bitcount                  ;oneskorenie kvoli casovaniu
570         brne     SendUSBAnswerBitstuffLoop ;zmeni pocitadlo bitov - podla carry flagu
571         ld       inputbuf,Y               ;ak pocitadlo bitov nie je nulove - opakuj vysielanie s dalsim bitom
572         ZeroBitStuf:
573         nop
574         cbr      temp0,3                  ;oneskorenie 2 cykly
575         out      outputport,temp0          ;oneskorenie 1 cyklus
576
577         ldi      bitcount,5                ;vysli EOP na USB
578 SendUSBWaitEOP:
579         dec      bitcount
580         brne     SendUSBWaitEOP
581
582         sbi      outputport,DATAMinus     ;nahodenie DATAMinus : kludovy stav na port USB
583         sbi      outputport,DATAMinus     ;oneskorenie 2 cykly: Idle ma trvat 1 bit (8 cyklov pri 12MHz)
584         cbi      USBdirection,DATApplus   ;DATApplus ako vstupny
585         cbi      USBdirection,DATAMinus    ;DATAMinus ako vstupny
586         cbi      outputport,DATAMinus     ;zhodenie DATAMinus : tretí stav na port USB
587         ret
588 ;-----
589 ToggleDATAPID:
590         lds      temp0,OutputBufferBegin+1 ;nahraj posledné PID

```

```

591          cpi      temp0,DATA1PID          ;ak bolo posledne DATA1PID byte
592          ldi      temp0,DATA0PID
593          breq     SendData0PID            ;tak posli nulovu odpoved s DATA0PID
594          ldi      temp0,DATA1PID          ;inak posli nulovu odpoved s DATA1PID
595 SendData0PID:
596          sts      OutputBufferBegin+1,temp0 ;DATA0PID byte
597          ret
598 ;-----
599 ComposeZeroDATA1PIDAnswer:
600          ldi      temp0,DATA0PID          ;DATA0 PID - v skutocnosti sa stoggluje na DATA1PID v nahrati deskriptora
601          sts      OutputBufferBegin+1,temp0 ;nahraj do vyst buffera
602 ComposeZeroAnswer:
603          ldi      temp0,SOPbyte
604          sts      OutputBufferBegin+0,temp0 ;SOP byte
605          rcall    ToggleDATAPID          ;zmen DATAPID
606          ldi      temp0,0x00
607          sts      OutputBufferBegin+2,temp0 ;CRC byte
608          sts      OutputBufferBegin+3,temp0 ;CRC byte
609          ldi      ByteCount,2+2          ;dlzka vystupneho buffera (SOP a PID + CRC16)
610          ret
611 ;-----
612 InitACKBuffer:
613          ldi      temp0,SOPbyte
614          sts      ACKBufferBegin+0,temp0  ;SOP byte
615          ldi      temp0,ACKPID
616          sts      ACKBufferBegin+1,temp0  ;ACKPID byte
617          ret
618 ;-----
619 SendACK:
620          push     USBBufptrY
621          push     bitcount
622          push     OutBitStuffNumber
623          ldi      USBBufptrY,ACKBufferBegin ;pointer na zaciatok ACK buffera
624          ldi      ByteCount,2              ;pocet vyslanych bytov (iba SOP a ACKPID)
625          clr      OutBitStuffNumber
626          rcall    SendUSBBuffer
627          pop      OutBitStuffNumber
628          pop      bitcount
629          pop      USBBufptrY
630          ret
631 ;-----
632 InitNAKBuffer:
633          ldi      temp0,SOPbyte
634          sts      NAKBufferBegin+0,temp0  ;SOP byte
635          ldi      temp0,NAKPID
636          sts      NAKBufferBegin+1,temp0  ;NAKPID byte
637          ret
638 ;-----
639 SendNAK:
640          push     OutBitStuffNumber
641          ldi      USBBufptrY,NAKBufferBegin ;pointer na zaciatok ACK buffera
642          ldi      ByteCount,2              ;pocet vyslanych bytov (iba SOP a NAKPID)
643          clr      OutBitStuffNumber
644          rcall    SendUSBBuffer

```

```

645         pop        OutBitStuffNumber
646         ret
647 ;-----
648 ComposeSTALL:
649         ldi         temp0,SOPbyte
650         sts         OutputBufferBegin+0,temp0        ;SOP byte
651         ldi         temp0,STALLPID
652         sts         OutputBufferBegin+1,temp0        ;STALLPID byte
653         ldi         ByteCount,2                      ;dlzka vystupneho buffera (SOP a PID)
654         ret
655 ;-----
656 DecodeNRZI:    ;enkodovanie buffera z NRZI kodu do binarneho
657         push        USBBufptrY                      ;zalohuj pointer do buffera
658         push        ByteCount                      ;zalohuj dlzku buffera
659         add         ByteCount,USBBufptrY            ;koniec buffera do ByteCount
660         ser         temp0                          ;na zabezpecenie jednotkového carry (v nasledujucej rotácii)
661 NRZIloop:
662         ror         temp0                          ;naplnenie carry z predchadzajuceho byte
663         ld          temp0,Y                        ;nahraj prijaty byte z buffera
664         mov         temp2,temp0                    ;posunuty register o jeden bit vpravo a XOR na funkciu NRZI dekodovania
665         ror         temp2                          ;carry do najvyssieho bitu a sucasne posuv
666         eor         temp2,temp0                    ;samotne dekodovanie NRZI
667         com         temp2                          ;negovanie
668         st          Y+,temp2                       ;ulozenie spat ako dekodovany byte a zvyš pointer do buffera
669         cp          USBBufptrY,ByteCount            ;ak este neboli vsetky
670         brne        NRZIloop                       ;tak opakuj
671         pop         ByteCount                      ;obnov dlzku buffera
672         pop         USBBufptrY                    ;obnov pointer do buffera
673         ret                                          ;inak skonci
674 ;-----
675 BitStuff:     ;odstranenie bit-stuffingu v buffri
676         clr         temp3                          ;pocitadlo vynechaných bitov
677         clr         lastBitstufNumber              ;0xFF do lastBitstufNumber
678         dec         lastBitstufNumber
679 BitStuffRepeat:
680         push        USBBufptrY                      ;zalohuj pointer do buffera
681         push        ByteCount                      ;zalohuj dlzku buffera
682         mov         temp1,temp3                    ;pocitadlo všetkých bitov
683         ldi         temp0,8                        ;spocitat všetky bity v bufferi
684 SumAllBits:
685         add         temp1,temp0
686         dec         ByteCount
687         brne        SumAllBits
688         ldi         temp2,6                        ;inicializuj pocitadlo jednotiek
689         pop         ByteCount                      ;obnov dlzku buffera
690         push        ByteCount                      ;zalohuj dlzku buffera
691         add         ByteCount,USBBufptrY            ;koniec buffera do ByteCount
692         inc         ByteCount                      ;a pre istotu ho zvyš o 2 (kvôli posuvaniu)
693         inc         ByteCount
694 BitStuffLoop:
695         ld          temp0,Y                        ;nahraj prijaty byte z buffera
696         ldi         bitcount,8                     ;pocitadlo bitov v byte
697 BitStuffByteLoop:
698         ror         temp0                          ;naplnenie carry z LSB

```

```

699          brcs    IncrementBitstuff      ;ak LSB=0
700          ldi     temp2,7                ;inicializuj pocitadlo jednotiek +1 (ak bola nula)
701 IncrementBitstuff:
702          dec     temp2                  ;zniz pocitadlo jednotiek (predpoklad jednotkoveho bitu)
703          brne    DontShiftBuffer        ;ak este nebolo 6 jednotiek za sebou - neposun buffer
704          cp      temp1,lastBitstufNumber ;
705          ldi     temp2,6                ;inicializuj pocitadlo jednotiek (ak by sa nerobil bitstuffing tak sa musi zacat odznova)
706          brcc    DontShiftBuffer        ;ak sa tu uz robil bitstuffing - neposun buffer
707
708          dec     temp1
709          mov     lastBitstufNumber,temp1 ;zapamataj si poslednu poziciu bitstuffingu
710          cpi     bitcount,1              ;aby sa ukazovalo na 7 bit (ktory sa ma vymazat alebo kde sa ma vlozit nula)
711          brne    NoBitcountCorrect
712          ldi     bitcount,9
713          inc     USBBufptrY              ;zvys pointer do buffera
714 NoBitcountCorrect:
715          dec     bitcount
716          bst     BitStuffInOut,0
717          brts    CorrectOutBuffer        ;ak je Out buffer - zvys dlzku buffera
718          rcall   ShiftDeleteBuffer      ;posun In buffer
719          dec     temp3                  ;zniz pocitadlo vynechani
720          rjmp    CorrectBufferEnd
721 CorrectOutBuffer:
722          rcall   ShiftInsertBuffer      ;posun Out buffer
723          inc     temp3                  ;zvys pocitadlo vynechani
724 CorrectBufferEnd:
725          pop     ByteCount              ;obnov dlzku buffera
726          pop     USBBufptrY            ;obnov pointer do buffera
727          rjmp    BitStuffRepeat        ;a restartni od zaciatku
728 DontShiftBuffer:
729          dec     temp1                  ;ak uz boli vsetky bity
730          breq    EndBitStuff            ;ukonci cyklus
731          dec     bitcount                ;zniz pocitadlo bitov v byte
732          brne    BitStuffByteLoop      ;ak este neboli vsetky bity v byte - chod na dalsi bit
733          ;inak nahraj dalsi byte
734          inc     USBBufptrY            ;zvys pointer do buffera
735          rjmp    BitStuffLoop          ;a opakuj
736 EndBitStuff:
737          pop     ByteCount              ;obnov dlzku buffera
738          pop     USBBufptrY            ;obnov pointer do buffera
739          bst     BitStuffInOut,0
740          brts    IncrementLength        ;ak je Out buffer - zvys dlzku Out buffera
741 DecrementLength:
742          ;ak je In buffer - zniz dlzku In buffera
743          ;bolo aspon jedno znizenie
744          cpi     temp3,0
745          breq    NoChangeByteCount      ;ak nie - nemen dlzku buffera
746          dec     ByteCount              ;ak je In buffer - zniz dlzku buffera
747          subi    temp3,256-8            ;ak nebolo viac ako 8 bitov naviac
748          brcc    NoChangeByteCount      ;tak skonci
749          dec     ByteCount              ;inak este zniz dlzku buffera
750          ret
751 IncrementLength:
752          mov     OutBitStuffNumber,temp3 ;zapamataj si pocet bitov naviac
753          subi    temp3,8                ;ak nebolo viac ako 8 bitov naviac
754          brcs    NoChangeByteCount      ;tak skonci

```

```

753         inc      ByteCount          ;inak zvys dlzku buffera
754         mov      OutBitStuffNumber,temp3 ;a zapamataj si pocet bitov naviac (znizene o 8)
755 NoChangeByteCount:
756         ret                                ;skonci
757 ;-----
758 ShiftInsertBuffer: ;posuv buffera o jeden bit vpravo od konca az po poziciu: byte-USBBufptrY a bit-bitcount
759         mov      temp0,bitcount        ;vypocet: bitcount= 9-bitcount
760         ldi      bitcount,9
761         sub      bitcount,temp0        ;do bitcount poloha bitu, ktory treba nulovat
762
763         ld       temp1,Y               ;nahraj byte ktory este treba posunut od pozicie bitcount
764         rol      temp1                 ;a posun vpravo cez Carry (prenos z vyssieho byte a LSB do Carry)
765         ser      temp2                 ;FF do masky - temp2
766 HalfInsertPosuvMask:
767         lsl      temp2                 ;nula do dalsieho spodneho bitu masky
768         dec      bitcount              ;az pokial sa nedosiahne hranica posuvania v byte
769         brne     HalfInsertPosuvMask
770
771         and      temp1,temp2           ;odmaskuj aby zostali iba vrchne posunute bity v temp1
772         com      temp2                 ;invertuj masku
773         lsr      temp2                 ;posun masku vpravo - na vloženie nuloveho bitu
774         ld       temp0,Y               ;nahraj byte ktory este treba posunut od pozicie bitcount do temp0
775         and      temp0,temp2           ;odmaskuj aby zostali iba spodne neposunute bity v temp0
776         or       temp1,temp0          ;a zluc posunutú a neposunutú časť
777
778         ld       temp0,Y               ;nahraj byte ktory este treba posunut od pozicie bitcount
779         rol      temp0                 ;a posun ho vpravo cez Carry (aby sa nastavilo správne Carry pre ďalšie prenosi)
780         st       Y+,temp1              ;a nahraj späť upravený byte
781 ShiftInsertBufferLoop:
782         cpse     USBBufptrY,ByteCount ;ak nie sú všetky celé byty
783         rjmp     NoEndShiftInsertBuffer ;tak pokračuj
784         ret                                ;inak skonci
785 NoEndShiftInsertBuffer:
786         ld       temp1,Y               ;nahraj byte
787         rol      temp1                 ;a posun vpravo cez Carry (prenos z nižšieho byte a LSB do Carry)
788         st       Y+,temp1              ;a nahraj späť
789         rjmp     ShiftInsertBufferLoop ;a pokračuj
790 ;-----
791 ShiftDeleteBuffer: ;posuv buffera o jeden bit vľavo od konca az po poziciu: byte-USBBufptrY a bit-bitcount
792         mov      temp0,bitcount        ;vypocet: bitcount= 9-bitcount
793         ldi      bitcount,9
794         sub      bitcount,temp0        ;do bitcount poloha bitu, ktory este treba posunut
795         mov      temp0,USBBufptrY      ;uschovanie pointera do buffera
796         inc      temp0                 ;pozicia celých bytov do temp0
797         mov      USBBufptrY,ByteCount ;maximalna pozicia do pointera
798 ShiftDeleteBufferLoop:
799         ld       temp1,-Y               ;zníž buffer a nahraj byte
800         ror      temp1                 ;a posun vpravo cez Carry (prenos z vyssieho byte a LSB do Carry)
801         st       Y,temp1               ;a nahraj späť
802         cpse     USBBufptrY,temp0      ;ak nie sú všetky celé byty
803         rjmp     ShiftDeleteBufferLoop ;tak pokračuj
804
805         ld       temp1,-Y               ;zníž buffer a nahraj byte ktory este treba posunut od pozicie bitcount
806         ror      temp1                 ;a posun vpravo cez Carry (prenos z vyssieho byte a LSB do Carry)

```

```

807         ser      temp2                ;FF do masky - temp2
808 HalfDeletePosuvMask:
809         dec      bitcount              ;az pokial sa nedosiahne hranica posuvania v byte
810         breq     DoneMask
811         lsl      temp2                ;nula do dalsieho spodneho bitu masky
812         rjmp     HalfDeletePosuvMask
813 DoneMask:
814         and      temp1,temp2           ;odmaskuj aby zostali iba vrchne posunute bity v temp1
815         com      temp2                ;invertuj masku
816         ld       temp0,Y              ;nahraj byte ktory este treba posunut od pozicie bitcount do temp0
817         and      temp0,temp2          ;odmaskuj aby zostali iba spodne neposunute bity v temp0
818         or       temp1,temp0          ;a zluc posunutu a neposunutu cast
819         st       Y,temp1              ;a nahraj spat
820         ret
821         ;-----
822 MirrorInBufferBytes:
823         push     USBBufptrY
824         push     ByteCount
825         ldi      USBBufptrY,InputBufferBegin
826         rcall    MirrorBufferBytes
827         pop      ByteCount
828         pop      USBBufptrY
829         ret
830         ;-----
831 MirrorBufferBytes:
832         add      ByteCount,USBBufptrY  ;ByteCount ukazuje na koniec spravy
833 MirrorBufferloop:
834         ld       temp0,Y              ;nahraj prijaty byte z buffera
835         ldi      temp1,8              ;pocitadlo bitov
836 MirrorBufferByteLoop:
837         ror      temp0                ;do carry dalsi najnizsi bit
838         rol      temp2                ;z carry dalsi bit na obratene poradie
839         dec      temp1                ;bol uz cely byte
840         brne     MirrorBufferByteLoop ;ak nie tak opakuj dalsi najnizsi bit
841         st       Y+,temp2             ;ulozenie spat ako obrateny byte a zvyš pointer do buffera
842         cp       USBBufptrY,ByteCount ;ak este neboli vsetky
843         brne     MirrorBufferloop     ;tak opakuj
844         ret
845         ;-----
846 ;CheckCRCIn:
847         ;
848         push     USBBufptrY
849         push     ByteCount
850         ldi      USBBufptrY,InputBufferBegin
851         rcall    CheckCRC
852         pop      ByteCount
853         pop      USBBufptrY
854         ;-----
855 AddCRCOut:
856         push     USBBufptrY
857         push     ByteCount
858         ldi      USBBufptrY,OutputBufferBegin
859         rcall    CheckCRC
860         com      temp0                ;negacia CRC

```



```

861      com      temp1
862      st        Y+,temp1          ;ulozenie CRC na koniec buffera (najskor MSB)
863      st        Y,temp0          ;ulozenie CRC na koniec buffera (potom LSB)
864      dec      USBBufptrY        ;pointer na poziciu CRC
865      ldi      ByteCount,2       ;potocit 2 byty CRC
866      rcall    MirrorBufferBytes ;opacne poradie bitov CRC (pri vysielani CRC sa posielala naskor MSB)
867      pop      ByteCount
868      pop      USBBufptrY
869      ret
870 ;-----
871 CheckCRC:      ;vstup: USBBufptrY = zaciatok spravy ,ByteCount = dlzka spravy
872      add      ByteCount,USBBufptrY ;ByteCount ukazuje na koniec spravy
873      inc      USBBufptrY          ;nastav pointer na zaciatok spravy - vynechat SOP
874      ld       temp0,Y+           ;nahraj PID do temp0
875      ;a nastav pointer na zaciatok spravy - vynechat aj PID
876      cpi      temp0,DATA0PID      ;ci je DATA0 pole
877      breq     ComputedATACRC      ;pocitaj CRC16
878      cpi      temp0,DATA1PID      ;ci je DATA1 pole
879      brne     CRC16End            ;ak nie tak skonci
880 ComputedATACRC:
881      ser      temp0              ;inicializacia zvytku LSB na 0xff
882      ser      temp1              ;inicializacia zvytku MSB na 0xff
883 CRC16Loop:
884      ld       temp2,Y+           ;nahraj spravu do temp2 a zvyt pointer do buffera
885      ldi      temp3,8            ;pocitadlo bitov v byte - temp3
886 CRC16LoopByte:
887      bst      temp1,7            ;do T uloz MSB zvytku (zvyt je iba 16 bitovy - 8 bit vyssieho byte)
888      bld      bitcount,0         ;do bitcount LSB uloz T - MSB zvytku
889      eor      bitcount,temp2     ;XOR bitu spravy a bitu zvytku - v LSB bitcount
890      rol      temp0              ;posun zvyt dolava - nizsi byte (dva byty - cez carry)
891      rol      temp1              ;posun zvyt dolava - vyssi byte (dva byty - cez carry)
892      cbr      temp0,1            ;znuluj LSB zvytku
893      lsr      temp2              ;posun spravu doprava
894      ror      bitcount           ;vysledok XOR-u bitov z LSB do carry
895      brcc     CRC16NoXOR         ;ak je XOR bitu spravy a MSB zvytku = 0 , tak nerob XOR
896      ldi      bitcount,CRC16poly>>8 ;do bitcount CRC polynom - vrchny byte
897      eor      temp1,bitcount     ;a urob XOR zo zvytkom a CRC polynomom - vrchny byte
898      ldi      bitcount,CRC16poly ;do bitcount CRC polynom - spodny byte
899      eor      temp0,bitcount     ;a urob XOR zo zvytkom a CRC polynomom - spodny byte
900 CRC16NoXOR:
901      dec      temp3              ;boli uz vsetky bity v byte
902      brne     CRC16LoopByte      ;ak nie, tak chod na dalsi bit
903      cp       USBBufptrY,ByteCount ;bol uz koniec spravy
904      brne     CRC16Loop         ;ak nie tak opakuj
905 CRC16End:
906      ret                        ;inak skonci (v temp0 a temp1 je vysledok)
907 ;-----
908 LoadDescriptorFromROM:
909      lpm      ;nahraj z pozicie ROM pointra do R0
910      st        Y+,R0             ;R0 uloz do buffera a zvyt buffer
911      adiw     ZH:ZL,1            ;zvyt ukazovatel do ROM
912      dec      ByteCount          ;pokial nie su vsetky byty
913      brne     LoadDescriptorFromROM ;tak nahravaj dalej
914      rjmp     EndFromRAMROM      ;inak skonci

```

```

915 ;-----
916 LoadDescriptorFromROMZeroInsert:
917     lpm                ;nahraj z pozicie ROM pointra do R0
918     st      Y+,R0      ;R0 uloz do buffera a zvyas buffer
919
920     bst      RAMread,3  ;ak je 3 bit jednotkovy - nebude sa vkladat nula
921     brtc     InsertingZero ;inak sa bude vkladat nula
922     adiw     ZH:ZL,1    ;zvyas ukazovatel do ROM
923     lpm                ;nahraj z pozicie ROM pointra do R0
924     st      Y+,R0      ;R0 uloz do buffera a zvyas buffer
925     clt                      ;a znuluj
926     bld      RAMread,3  ;treti bit v RAMread - aby sa v dalsom vkladali nuly
927     rjmp     InsertingZeroEnd ;a pokracuj
928 InsertingZero:
929     clr      R0          ;na vkladanie nul
930     st      Y+,R0        ;nulu uloz do buffera a zvyas buffer
931 InsertingZeroEnd:
932     adiw     ZH:ZL,1    ;zvyas ukazovatel do ROM
933     subi     ByteCount,2 ;pokial nie su vsetky byty
934     brne     LoadDescriptorFromROMZeroInsert ;tak nahravaj dalej
935     rjmp     EndFromRAMROM ;inak skonci
936 ;-----
937 LoadDescriptorFromSRAM:
938     ld      R0,Z        ;nahraj z pozicie RAM pointra do R0
939     st      Y+,R0        ;R0 uloz do buffera a zvyas buffer
940     inc      ZL          ;zvyas ukazovatel do RAM
941     dec      ByteCount   ;pokial nie su vsetky byty
942     brne     LoadDescriptorFromSRAM ;tak nahravaj dalej
943     rjmp     EndFromRAMROM ;inak skonci
944 ;-----
945 LoadDescriptorFromEEPROM:
946     out      EEAR,ZL     ;nastav adresu EEPROM
947     sbi      EECR,EERE   ;vycitaj EEPROM do registra EEDR
948     in       R0,EEDR     ;nahraj z EEDR do R0
949     st      Y+,R0        ;R0 uloz do buffera a zvyas buffer
950     inc      ZL          ;zvyas ukazovatel do RAM
951     dec      ByteCount   ;pokial nie su vsetky byty
952     brne     LoadDescriptorFromEEPROM ;tak nahravaj dalej
953     rjmp     EndFromRAMROM ;inak skonci
954 ;-----
955 LoadXXXXDescriptor:
956     ldi      temp0,SOPbyte ;SOP byte
957     sts      OutputBufferBegin,temp0 ;na zaciatok vysielacieho buffera dat SOP
958     ldi      ByteCount,8   ;8 bytov nahrat
959     ldi      USBBufptrY,OutputBufferBegin+2 ;do vysielacieho buffera
960
961     and      RAMread,RAMread ;ci sa bude citat z RAM alebo ROM-ky alebo EEPROM-ky
962     brne     FromRAMorEEPROM ;0=ROM,1=RAM,2=EEPROM,4=ROM s vkladanim nuly
963 FromROM:
964     rjmp     LoadDescriptorFromROM ;nahrat descriptor z ROM-ky
965 FromRAMorEEPROM:
966     sbrc     RAMread,2     ;ak RAMread=4
967     rjmp     LoadDescriptorFromROMZeroInsert ;citaj z ROM s vkladanim nuly
968     sbrc     RAMread,0     ;ak RAMread=1

```

```

969      rjmp    LoadDescriptorFromSRAM      ;nahraj data zo SRAM-ky
970      rjmp    LoadDescriptorFromEEPROM    ;inak citaj z EEPROM
971  EndFromRAMROM:
972      sbrc     RAMread,7                   ;ak je najvyssi bit v premennej RAMread=1
973      clr      RAMread                    ;znuluj RAMread
974      rcall    ToggleDATAPID              ;zmenit DATAPID
975      ldi      USBBufptrY,OutputBufferBegin+1 ;do vysielacieho buffera - pozicia DATA PID
976      ret
977  ;-----
978  PrepareUSBOutAnswer: ;pripravenie odpovede do buffera
979      rcall    PrepareUSBAnswer            ;pripravenie odpovede do buffera
980  MakeOutBitStuff:
981      inc      BitStuffInOut               ;vysielaci buffer - vkladanie bitstuff bitov
982      ldi      USBBufptrY,OutputBufferBegin ;do vysielacieho buffera
983      rcall    BitStuff
984      mov      OutputBufferLength,ByteCount ;dlzku odpovede zapamatat pre vysielanie
985      clr      BitStuffInOut               ;prijimaci buffer - mazanie bitstuff bitov
986      ret
987  ;-----
988  PrepareUSBAnswer: ;pripravenie odpovede do buffera
989      clr      RAMread                     ;nulu do RAMread premennej - cita sa z ROM-ky
990      lds      temp0,InputBufferBegin+2    ;bmRequestType do temp0
991      lds      temp1,InputBufferBegin+3    ;bRequest do temp1
992      cbr      temp0,0b10011111           ;ak je 5 a 6 bit nulovy
993      brne     VendorRequest               ;tak to nie je Vendor Request
994      rjmp     StandardRequest             ;ale je to standardny request
995  ;-----
996  DoSetInfraBufferEmpty:
997      rjmp     OneZeroAnswer               ;potvrđ prijem jednou nulou
998  ;-----
999  DoSetRS232Baud:
1000     lds      temp0,InputBufferBegin+4    ;prvy parameter - hodnota baudrate na RS232
1001     out      UBRR,temp0                  ;nastav rychlost UART-u
1002     rjmp     OneZeroAnswer               ;potvrđ prijem jednou nulou
1003  ;-----
1004  DoGetRS232Baud:
1005     in      R0,UBRR                      ;vrat rychlost UART-u v R0
1006     rjmp     DoGetIn                     ;a ukonci
1007  ;-----
1008  DoRS232Send:
1009     lds      temp0,InputBufferBegin+4    ;prvy parameter - hodnota vysielana na RS232
1010     out      UDR,temp0                   ;vysli data na UART
1011  WaitForRS232Send:
1012     sbis     UCR,TXEN                     ;ak nie je povoleny UART vysielac
1013     rjmp     OneZeroAnswer               ;tak skonci - ochrana kvoli zacykleniu v AT90S2323/2343
1014     sbis     USR,TXC                      ;pockat na dovysielanie bytu
1015     rjmp     WaitForRS232Send
1016     rjmp     OneZeroAnswer               ;potvrđ prijem jednou nulou
1017  ;-----
1018  DoRS232Read:
1019     rjmp     TwoZeroAnswer               ;iba potvrđ prijem dvoma nulami
1020  ;-----
1021  VendorRequest:
1022     clr      ZH                           ;pre citanie z RAM alebo EEPROM

```

```

1023
1024         cpi      temp1,1                ;
1025         breq      DoSetInfraBufferEmpty  ;restartne infra prijimanie (ak bolo zastavene citanim z RAM-ky)
1026
1027         cpi      temp1,2                ;
1028         breq      DoGetInfraCode         ;vysle prijaty infra kod (ak je v bufferi)
1029
1030         cpi      temp1,3                ;
1031         breq      DoSetDataPortDirection ;nastavi smer toku datovych bitov
1032         cpi      temp1,4                ;
1033         breq      DoGetDataPortDirection ;zisti smer toku datovych bitov
1034
1035         cpi      temp1,5                ;
1036         breq      DoSetOutDataPort       ;nastavi datove bity (ak su vstupne, tak ich pull-up)
1037         cpi      temp1,6                ;
1038         breq      DoGetOutDataPort       ;zisti nastavenie datovych out bitov (ak su vstupne, tak ich pull-up)
1039
1040         cpi      temp1,7                ;
1041         breq      DoGetInDataPort        ;vrati hodnotu datoveho vstupneho portu
1042
1043         cpi      temp1,8                ;
1044         breq      DoEEPROMRead           ;vrati obsah EEPROM od urcitej adresy
1045         cpi      temp1,9                ;
1046         breq      DoEEPROMWrite          ;zapise EEPROM na urcitu adresu urcite data
1047
1048         cpi      temp1,10               ;
1049         breq      DoRS232Send            ;vysle byte na seriovy linku
1050         cpi      temp1,11               ;
1051         breq      DoRS232Read            ;vrati prijaty byte zo seriovej linky (ak sa nejaky prijal)
1052
1053         cpi      temp1,12               ;
1054         breq      DoSetRS232Baud         ;nastavi prenosovu rychlost seriovej linky
1055         cpi      temp1,13               ;
1056         breq      DoGetRS232Baud         ;vrati prenosovu rychlost seriovej linky
1057         cpi      temp1,14               ;
1058         breq      DoGetRS232Buffer       ;vrati RS232 buffer
1059
1060         cpi      temp1,USER_FNC_NUMBER+0 ;
1061         breq      DoUserFunction0        ;vykona uzivatelsku rutinu0
1062         cpi      temp1,USER_FNC_NUMBER+1 ;
1063         breq      DoUserFunction1        ;vykona uzivatelsku rutinu1
1064         cpi      temp1,USER_FNC_NUMBER+2 ;
1065         breq      DoUserFunction2        ;vykona uzivatelsku rutinu2
1066
1067         rjmp      ZeroDATAAnswer         ;ak to bolo nieco nezname, tak priprav nulovu odpoved
1068
1069 ;----- USER FUNCTIONS -----
1070
1071 ;-----TEMPLATE OF YOUR FUNCTION-----
1072 ;----- BEGIN: This is template how to write own function -----
1073
1074 ;free of use are registers:
1075         ;temp0,temp1,temp2,temp3,ACC,ZH,ZL
1076         ;registers are destroyed after execution (use push/pop to save content)

```

```

1077
1078 ;at the end of routine you must correctly set registers:
1079 ;RAMread - 0=reading from ROM, 1=reading from RAM, 2=reading from EEPROM
1080 ;temp0 - number of transmitted data bytes
1081 ;ZH,ZL - pointer to buffer of transmitted data (pointer to ROM/RAM/EEPROM)
1082
1083 ;to transmit data (preparing data to buffer) :
1084 ;to transmit data you must jump to "ComposeEndXXXXDescriptor"
1085 ;to transmit one zero byte you can jump to "OneZeroAnswer" (commonly used as confirmation of correct processing)
1086 ;to transmit two zero byte you can jump to "TwoZeroAnswer" (commonly used as confirmation of error in processing)
1087
1088 DoUserFunctionX:
1089 DoUserFunction0: ;send byte(s) of RAM starting at position given by first parameter in function
1090             lds     temp0,InputBufferBegin+4           ;prvy parameter Lo do temp0
1091             ;lds     temp1,InputBufferBegin+5           ;prvy parameter Hi do temp1
1092             ;lds     temp2,InputBufferBegin+6           ;druhy parameter Lo do temp2
1093             ;lds     temp3,InputBufferBegin+7           ;druhy parameter Hi do temp3
1094             ;lds     ACC,InputBufferBegin+8             ;pocet pozadovanych bytov do ACC
1095
1096             ;Tu si pridajte vlastny kod:
1097             ;-----
1098             nop                                     ;priklad na kod - nic nerobi
1099             nop
1100             nop
1101             nop
1102             nop
1103             ;-----
1104
1105             mov      ZL,temp0                         ;bude sa posielat hodnota RAM adresy ulozena v temp0 (prvy parameter funkcie)
1106             inc      RAMread                           ;RAMread=1 - cita sa z RAM-ky
1107             ldi      temp0,RAMEND+1                    ;posli max pocet byte - celu RAM
1108             rjmp     ComposeEndXXXXDescriptor          ;a priprav data
1109 DoUserFunction1:
1110             rjmp     OneZeroAnswer                     ;potvrđ prijem jednou nulou
1111 DoUserFunction2:
1112             rjmp     TwoZeroAnswer                     ;potvrđ prijem dvoma nulami
1113 ;----- END: This is template how to write own function -----
1114
1115 ;----- USER FUNCTIONS -----
1116
1117 DoGetInfraCode:
1118             rjmp     OneZeroAnswer                     ;potvrđ prijem jednou nulou
1119
1120 DoEEPROMRead:
1121             lds      ZL,InputBufferBegin+4             ;prvy parameter - offset v EEPROM-ke
1122             ldi      temp0,2
1123             mov      RAMread,temp0                     ;RAMread=2 - cita sa z EEPROM-ky
1124             ldi      temp0,E2END+1                     ;pocet mojich bytovych odpovedi do temp0 - cela dlzka EEPROM
1125             rjmp     ComposeEndXXXXDescriptor          ;inak priprav data
1126 DoEEPROMWrite:
1127             lds      ZL,InputBufferBegin+4             ;prvy parameter - offset v EEPROM-ke (adresa)
1128             lds      R0,InputBufferBegin+6             ;druhy parameter - data, ktore sa maju zapisat do EEPROM-ky (data)
1129             rjmp     EEPROMWrite                       ;zapis do EEPROM a aj ukonci prikaz
1130 DoSetDataPortDirection:

```

```

1131         lds      ACC,InputBufferBegin+4      ;prvy parameter - smer datovych bitov
1132         rcall    SetDataPortDirection
1133         rjmp     OneZeroAnswer                ;potvrđ prijem jednou nulou
1134 DoGetDataPortDirection:
1135         rcall    GetDataPortDirection
1136         rjmp     DoGetIn
1137
1138 DoSetOutDataPort:
1139         lds      ACC,InputBufferBegin+4      ;prvy parameter - hodnota datovych bitov
1140         rcall    SetOutDataPort
1141         rjmp     OneZeroAnswer                ;potvrđ prijem jednou nulou
1142 DoGetOutDataPort:
1143         rcall    GetOutDataPort
1144         rjmp     DoGetIn
1145
1146 DoGetInDataPort:
1147         rcall    GetInDataPort
1148 DoGetIn:
1149         ldi      ZL,0                          ;posiela sa hodnota v R0
1150         ldi      temp0,0x81                    ;RAMread=1 - cita sa z RAM-ky
1151         mov      RAMread,temp0                 ;(najvyssi bit na 1 - aby sa hned premenna RAMread znulovala)
1152         ldi      temp0,1                      ;posli iba jeden byte
1153         rjmp     ComposeEndXXXXDescriptor      ;a priprav data
1154
1155 DoGetRS232Buffer:
1156         mov      temp0,RS232BufferFull        ;zisti dlzku buffera RS232 kodu
1157         cpi      temp0,0                      ;ak je RS232 Buffer prazdny
1158         breq     OneZeroAnswer                ;tak nic neposli a potvrđ prijem jednou nulou
1159
1160         lds      ACC,InputBufferBegin+8      ;pocet pozadovanych bytov do ACC
1161         inc      temp0                        ;pocet moznych dodanych bajtov (plus byte dlzky buffera)
1162         cp       ACC,temp0                    ;ak sa neziada viac ako mozem dodat
1163         brcc     NoShortGetRS232Buffer        ;vysli tolko kolko sa ziada
1164         mov      temp0,ACC
1165 NoShortGetRS232Buffer:
1166         dec      temp0                        ;uber byte dlzky
1167         lds      temp1,RS232BufferBegin+1     ;zisti ukazovatel citania buffera RS232 kodu : 2.byte hlavicky (dlzka kodu + citanie + zapis +
rezerva)
1168         add      temp1,temp0                  ;zisti kde je koniec
1169         cpi      temp1,RS232BufferBegin+MAXRS232LENGTH+1 ;ak by mal pretiect
1170         brcs     ReadNoOverflow
1171         subi     temp1,RS232BufferBegin+MAXRS232LENGTH+1 ;vypocitaj kolko sa neprenesie
1172         sub      temp0,temp1                  ;a o to skrat dlzku citania
1173         ldi      temp1,RS232BufferBegin+4     ;a zacni od nuly
1174 ReadNoOverflow:
1175         lds      ZL,RS232BufferBegin+1        ;zisti ukazovatel citania buffera RS232 kodu : 2.byte hlavicky (dlzka kodu + citanie + zapis +
rezerva)
1176         sts      RS232BufferBegin+1,temp1     ;zapis novy ukazovatel citania buffera RS232 kodu : 2.byte hlavicky (dlzka kodu + citanie + zapis +
rezerva)
1177         dec      ZL                          ;priestor pre udaj dlky - prenasá sa ako prvý bajt
1178
1179         sub      RS232BufferFull,temp0        ;zniz dlzku buffera
1180         st       Z,RS232BufferFull            ;a uloz skutocnu dlzku do paketu
1181         inc      temp0                        ;a o tento jeden bajt zvyš pocet prenasanych bajtov (dlzka buffera)

```

```

1182         inc      RAMread          ;RAMread=1 - cita sa z RAM-ky
1183         rjmp     ComposeEndXXXXDescriptor ;a priprav data
1184 ;----- END USER FUNCTIONS -----
1185
1186 OneZeroAnswer: ;posle jednu nulu
1187         ldi      temp0,1           ;pocet mojich bytovych odpovedi do temp0
1188         rjmp     ComposeGET_STATUS2
1189
1190 StandardRequest:
1191         cpi      temp1,GET_STATUS   ;
1192         breq     ComposeGET_STATUS   ;
1193
1194         cpi      temp1,CLEAR_FEATURE ;
1195         breq     ComposeCLEAR_FEATURE ;
1196
1197         cpi      temp1,SET_FEATURE   ;
1198         breq     ComposeSET_FEATURE   ;
1199
1200         cpi      temp1,SET_ADDRESS    ;ak sa ma nastavit adresa
1201         breq     ComposeSET_ADDRESS    ;nastav adresu
1202
1203         cpi      temp1,GET_DESCRIPTOR ;ak sa ziada descriptor
1204         breq     ComposeGET_DESCRIPTOR ;vygeneruj ho
1205
1206         cpi      temp1,SET_DESCRIPTOR ;
1207         breq     ComposeSET_DESCRIPTOR ;
1208
1209         cpi      temp1,GET_CONFIGURATION ;
1210         breq     ComposeGET_CONFIGURATION ;
1211
1212         cpi      temp1,SET_CONFIGURATION ;
1213         breq     ComposeSET_CONFIGURATION ;
1214
1215         cpi      temp1,GET_INTERFACE   ;
1216         breq     ComposeGET_INTERFACE   ;
1217
1218         cpi      temp1,SET_INTERFACE   ;
1219         breq     ComposeSET_INTERFACE   ;
1220
1221         cpi      temp1,SYNCH_FRAME     ;
1222         breq     ComposeSYNCH_FRAME     ;
1223         ;ak sa nenasla znama poziadavka
1224         rjmp     ZeroDATAAnswer        ;ak to bolo nieco nezname, tak priprav nulovu odpoved
1225
1226 ComposeSET_ADDRESS:
1227         lds      temp1,InputBufferBegin+4 ;nova adresa do temp1
1228         rcall    SetMyNewUSBAddresses    ;ENG;and compute NRZI and bitstuffing coded addresses
1229         ldi      State,AddressChangeState ;nastav stav pre zmenu adresy
1230         rjmp     ZeroDATAAnswer        ;posli nulovu odpoved
1231
1232 ComposeSET_CONFIGURATION:
1233         lds      ConfigByte,InputBufferBegin+4 ;cislo konfiguracie do premennej ConfigByte
1234 ComposeCLEAR_FEATURE:
1235 ComposeSET_FEATURE:

```

```

1236 ComposeSET_INTERFACE:
1237 ZeroStringAnswer:
1238     rjmp     ZeroDATAAnswer           ;posli nulovu odpoved
1239 ComposeGET_STATUS:
1240 TwoZeroAnswer:
1241     ldi      temp0,2                 ;pocet mojich bytovych odpovedi do temp0
1242 ComposeGET_STATUS2:
1243     ldi      ZH, high(StatusAnswer<<1) ;ROMpointer na odpoved
1244     ldi      ZL, low(StatusAnswer<<1)
1245     rjmp     ComposeEndXXXDescriptor ;a dokonci
1246 ComposeGET_CONFIGURATION:
1247     and      ConfigByte,ConfigByte   ;ak som nenakonfigurovany
1248     breq     OneZeroAnswer           ;tak posli jednu nulu - inak posli moju konfiguraciu
1249     ldi      temp0,1                 ;pocet mojich bytovych odpovedi do temp0
1250     ldi      ZH, high(ConfigAnswerMinus1<<1) ;ROMpointer na odpoved
1251     ldi      ZL, low(ConfigAnswerMinus1<<1)+1
1252     rjmp     ComposeEndXXXDescriptor ;a dokonci
1253 ComposeGET_INTERFACE:
1254     ldi      ZH, high(InterfaceAnswer<<1) ;ROMpointer na odpoved
1255     ldi      ZL, low(InterfaceAnswer<<1)
1256     ldi      temp0,1                 ;pocet mojich bytovych odpovedi do temp0
1257     rjmp     ComposeEndXXXDescriptor ;a dokonci
1258 ComposeSYNCH_FRAME:
1259 ComposeSET_DESCRIPTOR:
1260     rcall    ComposeSTALL
1261     ret
1262 ComposeGET_DESCRIPTOR:
1263     lds      temp1,InputBufferBegin+5 ;DescriptorType do temp1
1264     cpi      temp1,DEVICE              ;DeviceDescriptor
1265     breq     ComposeDeviceDescriptor ;
1266     cpi      temp1,CONFIGURATION       ;ConfigurationDescriptor
1267     breq     ComposeConfigDescriptor  ;
1268     cpi      temp1,STRING              ;StringDeviceDescriptor
1269     breq     ComposeStringDescriptor ;
1270     ret
1271 ComposeDeviceDescriptor:
1272     ldi      ZH, high(DeviceDescriptor<<1) ;ROMpointer na descriptor
1273     ldi      ZL, low(DeviceDescriptor<<1)
1274     ldi      temp0,0x12                 ;pocet mojich bytovych odpovedi do temp0
1275     rjmp     ComposeEndXXXDescriptor ;a dokonci
1276 ComposeConfigDescriptor:
1277     ldi      ZH, high(ConfigDescriptor<<1) ;ROMpointer na descriptor
1278     ldi      ZL, low(ConfigDescriptor<<1)
1279     ldi      temp0,9+9+7                 ;pocet mojich bytovych odpovedi do temp0
1280 ComposeEndXXXDescriptor:
1281     lds      TotalBytesToSend,InputBufferBegin+8 ;pocet pozadovanych bytov do TotalBytesToSend
1282     cp       TotalBytesToSend,temp0          ;ak sa neziada viac ako mozem dodat
1283     brcs     HostConfigLength               ;vysli tolko kolko sa ziada
1284     mov      TotalBytesToSend,temp0         ;inak posli pocet mojich odpovedi
1285 HostConfigLength:
1286     mov      temp0,TotalBytesToSend         ;
1287     clr      TransmitPart                  ;nuluj pocet 8 bytovych odpovedi
1288     andi     temp0,0b00000111             ;ak je dlzka delitelna 8-mimi
1289     breq     Length8Multiply              ;tak nezapocitaj jednu necelu odpoved (pod 8 bytov)

```



```

1290      inc      TransmitPart      ;inak ju zapocitaj
1291 Length8Multiply:
1292      mov      temp0,TotalBytesToSend      ;
1293      lsr      temp0      ;dlzka 8 bytovych odpovedi sa dosiahne
1294      lsr      temp0      ;delenie celociselne 8-mimi
1295      lsr      temp0
1296      add      TransmitPart,temp0      ;a pripocitanim k poslednej necelej 8-mici do premennej TransmitPart
1297      ldi      temp0,DATA0PID      ;DATA0 PID - v skutocnosti sa stoggluje na DATA1PID v nahrati deskriptora
1298      sts      OutputBufferBegin+1,temp0      ;nahraj do vyst buffera
1299      rjmp     ComposeNextAnswerPart
1300 ComposeStringDescriptor:
1301      ldi      temp1,4+8      ;ak RAMread=4(vkladaj nuly z ROM-koveho citania) + 8(za prvý byte nevkladať nulu)
1302      mov      RAMread,temp1
1303      lds      temp1,InputBufferBegin+4      ;DescriptorIndex do temp1
1304      cpi      temp1,0      ;LANGID String
1305      breq     ComposeLangIDString      ;
1306      cpi      temp1,2      ;DevNameString
1307      breq     ComposeDevNameString      ;
1308      brcc     ZeroStringAnswer      ;ak je DescriptorIndex vyšší než 2 - posli nulovú odpoveď
1309      ;inak to bude VendorString
1310 ComposeVendorString:
1311      ldi      ZH, high(VendorStringDescriptor<<1)      ;ROMpointer na descriptor
1312      ldi      ZL, low(VendorStringDescriptor<<1)
1313      ldi      temp0,(VendorStringDescriptorEnd-VendorStringDescriptor)*4-2      ;počet mojich bytových odpovedí do temp0
1314      rjmp     ComposeEndXXXXDescriptor      ;a dokonci
1315 ComposeDevNameString:
1316      ldi      ZH, high(DevNameStringDescriptor<<1)      ;ROMpointer na descriptor
1317      ldi      ZL, low(DevNameStringDescriptor<<1)
1318      ldi      temp0,(DevNameStringDescriptorEnd-DevNameStringDescriptor)*4-2      ;počet mojich bytových odpovedí do temp0
1319      rjmp     ComposeEndXXXXDescriptor      ;a dokonci
1320 ComposeLangIDString:
1321      clr      RAMread
1322      ldi      ZH, high(LangIDStringDescriptor<<1)      ;ROMpointer na descriptor
1323      ldi      ZL, low(LangIDStringDescriptor<<1)
1324      ldi      temp0,(LangIDStringDescriptorEnd-LangIDStringDescriptor)*2;pocet mojich bytovych odpovedi do temp0
1325      rjmp     ComposeEndXXXXDescriptor      ;a dokonci
1326 ;-----
1327 ZeroDATA1Answer:
1328      rcall    ComposeZeroDATA1PIDAnswer
1329      ret
1330 ;-----
1331 SetMyNewUSBAddresses:      ;nastavi nove USB adresy v NRZI kodovani      ;ENG;set new USB addresses in NRZI coded
1332      mov      temp2,temp1      ;address to temp2 and temp1 and temp3
1333      mov      temp3,temp1      ;
1334      cpi      temp1,0b01111111      ;ENG;if address contains less than 6 ones
1335      brne     NewAddressNo6ones      ;ENG;then don't add bitstuffing
1336      ldi      temp1,0b10111111      ;ENG;else insert one zero - bitstuffing
1337 NewAddressNo6ones:
1338      andi     temp3,0b00000111      ;ENG;mask 3 low bits of Address
1339      cpi      temp3,0b00000111      ;ENG;and if 3 low bits of Address is not all ones
1340      brne     NewAddressNo3ones      ;ENG;then no change address
1341      ;ENG;else insert zero after 3-rd bit (bitstuffing)
1342      sec      ;set carry
1343      rol      temp2      ;ENG;rotate left

```

```

1344      andi    temp2,0b11110111      ;ENG;and inserted zero after 3-rd bit
1345  NewAddressNo3ones:
1346      sts     MyOutAddressSRAM,temp2 ;ENG;store new non-coded address Out (temp2)
1347      ;ENG;and now perform NRZI coding
1348      rcall   NRZIforAddress          ;ENG;NRZI for AddressIn (in temp1)
1349      sts     MyInAddressSRAM,ACC     ;ENG;store NRZI coded AddressIn
1350
1351      lds     temp1,MyOutAddressSRAM ;ENG;load non-coded address Out (in temp1)
1352      rcall   NRZIforAddress          ;ENG;NRZI for AddressOut
1353      sts     MyOutAddressSRAM,ACC     ;ENG;store NRZI coded AddressOut
1354
1355      ret                                ;ENG;and return
1356  ;-----
1357  NRZIforAddress:
1358      clr     ACC                     ;vychodzi stav odpovede - mojej nNRZI USB adresy ;ENG;original answer state - of my nNRZI USB address
1359      ldi     temp2,0b00000001        ;maska na xorovanie ;ENG;mask for xoring
1360      ldi     temp3,8                  ;pocitadlo bitov ;ENG;bits counter
1361  SetMyNewUSBAddressesLoop:
1362      mov     temp0,ACC                ;zapamatat si koncovu odpoved ;ENG;remember final answer
1363      ror     temp1                    ;do carry vysielany bit LSB (v smere naskor LSB a potom MSB) ;ENG;to carry transmitting bit LSB (in
direction firstly LSB then MSB)
1364      brcs    NoXORBits                ;ak je jedna - nemen stav ;ENG;if one - don't change state
1365      eor     temp0,temp2              ;inak sa bude stav menit podla posledneho bitu odpovede ;ENG;otherwise state will be changed according to
last bit of answer
1366  NoXORBits:
1367      ror     temp0                    ;posledny bit zmenenej odpovede do carry ;ENG;last bit of changed answer to carry
1368      rol     ACC                     ;a z carry do koncovej odpovede na miesto LSB (a sucasne prehodenie LSB a MSB poradia) ;ENG;and from carry
to final answer to the LSB place (and reverse LSB and MSB order)
1369      dec     temp3                    ;zmensi pocitadlo bitov ;ENG;decrement bits counter
1370      brne    SetMyNewUSBAddressesLoop ;ak pocitadlo bitov nie je nulove opakuj vysielanie s dalsim bitom ;ENG;if bits counter isn't
zero repeat transmitting with next bit
1371      ret
1372  ;-----
1373  ;-----
1374  PrepareOutContinuousBuffer:
1375      rcall   PrepareContinuousBuffer
1376      rcall   MakeOutBitStuff
1377      ret
1378  ;-----
1379  PrepareContinuousBuffer:
1380      mov     temp0,TransmitPart
1381      cpi     temp0,1
1382      brne    NextAnswerInBuffer      ;ak uz je buffer prazdny
1383      rcall   ComposeZeroAnswer        ;priprav nulovu odpoved
1384      ret
1385  NextAnswerInBuffer:
1386      dec     TransmitPart              ;znizit celkovu dlzku odpovede
1387  ComposeNextAnswerPart:
1388      mov     temp1,TotalBytesToSend    ;zniz pocet bytov na vyslanie
1389      subi    temp1,8                  ;ci je este treba poslat viac ako 8 bytov
1390      ldi     temp3,8                  ;ak ano - posli iba 8 bytov
1391      brcc    Nad8Bytov
1392      mov     temp3,TotalBytesToSend    ;inak posli iba dany pocet bytov
1393      clr     TransmitPart

```

```

1394             inc      TransmitPart          ;a bude to posledna odpoved
1395 Nad8Bytov:
1396             mov       TotalBytesToSend,temp1 ;znizeny pocet bytov do TotalBytesToSend
1397             rcall    LoadXXXDescriptor
1398             ldi       ByteCount,2           ;dlzka vystupneho buffera (iba SOP a PID)
1399             add       ByteCount,temp3       ;+ pocet bytov
1400             rcall    AddCRCOut              ;pridanie CRC do buffera
1401             inc       ByteCount             ;dlzka vystupneho buffera + CRC16
1402             inc       ByteCount
1403             ret
1404 ;-----
1405 .equ    USBVersion      =0x0101           ;pre aku verziu USB je to (1.01)
1406 .equ    VendorUSBID     =0x03EB          ;identifikator dodavateľa (Atmel=0x03EB)
1407 .equ    DeviceUSBID     =0x21FE          ;identifikator výrobku (USB to RS232 converter AT90S2313=0x21FE)
1408 .equ    DeviceVersion   =0x0002          ;cislo verzie výrobku (verzia=0.02)
1409 .equ    MaxUSBCurrent   =46              ;prudovy odber z USB (46mA)
1410 ;-----
1411 DeviceDescriptor:
1412             .db        0x12,0x01          ;0 byte - velkost deskriptora v bytoch
1413                                     ;1 byte - typ deskriptora: Deskriptor zariadenia
1414             .dw        USBVersion         ;2,3 byte - verzia USB LSB (1.00)
1415             .db        0x00,0x00          ;4 byte - trieda zariadenia
1416                                     ;5 byte - podtrieda zariadenia
1417             .db        0x00,0x08          ;6 byte - kod protokolu
1418                                     ;7 byte - velkost FIFO v bytoch
1419             .dw        VendorUSBID        ;8,9 byte - identifikator dodavateľa (Cypress=0x04B4)
1420             .dw        DeviceUSBID        ;10,11 byte - identifikator výrobku (teplomer=0x0002)
1421             .dw        DeviceVersion      ;12,13 byte - cislo verzie výrobku (verzia=0.01)
1422             .db        0x01,0x02          ;14 byte - index stringu "vyrobca"
1423                                     ;15 byte - index stringu "vyrobok"
1424             .db        0x00,0x01          ;16 byte - index stringu "seriove cislo"
1425                                     ;17 byte - pocet moznych konfiguracii
1426 DeviceDescriptorEnd:
1427 ;-----
1428 ConfigDescriptor:
1429             .db        0x09,0x02          ;dlzka,typ deskriptoru
1430 ConfigDescriptorLength:
1431             .dw        9+9+7              ;celkova dlzka vsetkych deskriptorov
1432             ConfigAnswerMinus1:
1433             .db        1,1                ;pre poslanie cisla congiguration number (pozor je treba este pricitat 1)
1434             .db        0,0x80             ;numInterfaces,congiguration number
1435             .db        MaxUSBCurrent/2,0x09 ;popisny index stringu, atributy;bus powered
1436             .db        0x04,0             ;prudovy odber, interface descriptor length
1437             InterfaceAnswer:
1438             .db        0,1                ;interface descriptor; cislo interface
1439             StatusAnswer:
1440             .db        0,0                ;pre poslanie cisla alternativneho interface
1441             .db        0,0                ;alternativne nastavenie interface; pocet koncovych bodov okrem EP0
1442             .db        0x07,0x5           ;2 nulove odpovede (na usetrenie miestom)
1443             .db        0x81,0             ;trieda rozhrania; podtrieda rozhrania
1444             .dw        0x08               ;kod protokolu; index popisneho stringu
1445             .db        10,0              ;dlzka,typ deskriptoru - endpoint
1446             .db        0x81,0             ;endpoint address; transfer type
1447             .db        0x08               ;max packet size
1448             .db        10,0              ;polling interval [ms]; dummy byte (pre vyplnenie)
1449 ConfigDescriptorEnd:
1450 ;-----

```

```

1448 LangIDStringDescriptor:
1449     .db      (LangIDStringDescriptorEnd-LangIDStringDescriptor)*2,3 ;dlzka, typ: string deskriptor
1450     .dw      0x0409 ;English
1451 LangIDStringDescriptorEnd:
1452 ;-----
1453 VendorStringDescriptor:
1454     .db      (VendorStringDescriptorEnd-VendorStringDescriptor)*4-2,3 ;dlzka, typ: string deskriptor
1455 Copyright:
1456     .db      "Ing. Igor Cesko"
1457 CopyrightEnd:
1458 VendorStringDescriptorEnd:
1459 ;-----
1460 DevNameStringDescriptor:
1461     .db      (DevNameStringDescriptorEnd-DevNameStringDescriptor)*4-2,3;dlzka, typ: string deskriptor
1462     .db      "AVR309:USB to UART protocol converter (simple)"
1463 DevNameStringDescriptorEnd:
1464 ;-----
1465 MaskPortData:
1466     bst      ACC,0
1467     bld      temp0,LEDlsb0
1468     bst      ACC,1
1469     bld      temp0,LEDlsb1
1470     bst      ACC,2
1471     bld      temp0,LEDlsb2
1472     bst      ACC,3
1473     bld      temp1,LEDmsb3
1474     bst      ACC,4
1475     bld      temp1,LEDmsb4
1476     bst      ACC,5
1477     bld      temp1,LEDmsb5
1478     bst      ACC,6
1479     bld      temp1,LEDmsb6
1480     bst      ACC,7
1481     bld      temp1,LEDmsb7
1482     ret
1483 ;-----
1484 SetDataPortDirection:
1485     in      temp0,LEDdirectionLSB ;nacitaj aktualny stav LSB do temp0 (aby sa nezmenili ostatne smery bitov)
1486     in      temp1,LEDdirectionMSB ;nacitaj aktualny stav MSB do temp1 (aby sa nezmenili ostatne smery bitov)
1487     rcall   MaskPortData
1488     out     LEDdirectionLSB,temp0 ;a update smeru LSB datoveho portu
1489     out     LEDdirectionMSB,temp1 ;a update smeru MSB datoveho portu
1490     ret
1491 ;-----
1492 SetOutDataPort:
1493     in      temp0,LEDPortLSB ;nacitaj aktualny stav LSB do temp0 (aby sa nezmenili ostatne bity)
1494     in      temp1,LEDPortMSB ;nacitaj aktualny stav MSB do temp1 (aby sa nezmenili ostatne bity)
1495     rcall   MaskPortData
1496     out     LEDPortLSB,temp0 ;a update LSB datoveho portu
1497     out     LEDPortMSB,temp1 ;a update MSB datoveho portu
1498     ret
1499 ;-----
1500 GetInDataPort:
1501     in      temp0,LEDPinMSB ;nacitaj aktualny stav MSB do temp0

```

```

1502      in      temp1,LEDPinLSB          ;nacitaj aktualny stav LSB do temp1
1503 MoveLEDin:
1504      bst     temp1,LEDlsb0            ;a daj bity LSB na spravne pozicie (z temp1 do temp0)
1505      bld     temp0,0                  ;(bity MSB su na spravnom mieste)
1506      bst     temp1,LEDlsb1
1507      bld     temp0,1
1508      bst     temp1,LEDlsb2
1509      bld     temp0,2
1510      mov     R0,temp0                ;a vysledok uloz do R0
1511      ret
1512 ;-----
1513 GetOutDataPort:
1514      in      temp0,LEDPortMSB         ;nacitaj aktualny stav MSB do temp0
1515      in      temp1,LEDPortLSB         ;nacitaj aktualny stav LSB do temp1
1516      rjmp    MoveLEDin
1517 ;-----
1518 GetDataPortDirection:
1519      in      temp0,LEDdirectionMSB    ;nacitaj aktualny stav MSB do temp0
1520      in      temp1,LEDdirectionLSB    ;nacitaj aktualny stav LSB do temp1
1521      rjmp    MoveLEDin
1522 ;-----
1523 EEPROMWrite:
1524      out     EEAR,ZL                  ;nastav adresu EEPROM
1525      out     EEDR,R0                  ;nastav data do EEPROM
1526      cli     ;zakaz prerusenie
1527      sbi     EECR,EEMWE               ;nastav master write enable
1528      sei     ;povol prerusenie (este sa vykona nasledujuca instrukcia)
1529      sbi     EECR,EWE                 ;samotny zapis
1530 WaitForEEPROMReady:
1531      sbic    EECR,EWE                 ;pockaj si na koniec zapisu
1532      rjmp    WaitForEEPROMReady       ;v slucke (max cca 4ms) (kvoli naslednemu citaniu/zapisu)
1533      rjmp    OneZeroAnswer            ;potvrđ prijem jednou nulou
1534 ;-----
1535 ;*****
1536 ;* End of Program
1537 ;*****
1538 ;-----
1539 ;-----
1540 ;*****
1541 ;* End of file
1542 ;*****

```