```
/*-----------------------------------------------------------------------------
 Eisenbahnsteuerung made by: Andreas Lutters / Achim Walder

 portions by Ulrich Radig

 Kommandointerpreter

 Dieses Programm ist freie Software. Sie können es unter den Bedingungen der
 GNU General Public License, wie von der Free Software Foundation veröffentlicht,
 weitergeben und/oder modifizieren, entweder gemäß Version 2 der Lizenz oder
 (nach Ihrer Option) jeder späteren Version.

 Die Veröffentlichung dieses Programms erfolgt in der Hoffnung,
 daß es Ihnen von Nutzen sein wird, aber OHNE IRGENDEINE GARANTIE,
 sogar ohne die implizite Garantie der MARKTREIFE oder der VERWENDBARKEIT
 FÜR EINEN BESTIMMTEN ZWECK. Details finden Sie in der GNU General Public License.

 Sie sollten eine Kopie der GNU General Public License zusammen mit diesem
 Programm erhalten haben.
 Falls nicht, schreiben Sie an die Free Software Foundation, gct

 Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307, USA.

 ##############################################################################
 Program uses parts of the above mentioned Software of Ulrich Radig.
 All other code is solely public domain software contributed to the public under the
 above mentioned GNU licence

##############################################################################

Beschreibung: https://www.mikrocontroller.net/articles/AVR-GCC-Tutorial

-----------------------------------------------------------------------------*/

#include "config.h"
#include "cmd.h"
#include <avr/io.h>
#include <util/delay.h>
#include <string.h>
#include <stdlib.h>
#include <avr/eeprom.h>
#include "usart.h"
#include "timer.h"

volatile unsigned int variable[MAX_VAR];

COMMAND_STRUCTUR COMMAND_TABELLE[] = ------------------------------------------------------------//
Befehls-Tabelle
------{
------//@ MC-Befehle
        ------------------------------------------------------------------------------------------
        ------
------{"MC_RESET",------------set_MC_Reset},------------------------------------------------// MC neu
        starten
------{"MC_INIT",-------------set_MC_INIT},-------------------------------------------------// MC-Port
        A,C,D in Grundstellung
------{"MC_READY",------------set_MC_ready},------------------------------------------------// MC-Aktiv ?
------{"MC_LED",--------------set_MC_LED},--------------------------------------------------// MC-LED
        Testlauf

------//@ MC Reister setzten
        ------------------------------------------------------------------------------------------
------{"MC_StReg",------------set_MC_SteuerRegister},---------------------------------------//
        MC-SteuerRegister
------{"MC_BM",---------------set_MC_Betrieb_Modus},-----------------------------------------//
        MC-Betriebs-Modus 0 - 4
------{"MC_RM",---------------set_MC_Response_Modus},----------------------------------------//
        MC-Rückmeldung an
------{"MC_TM",---------------set_MC_Test_Modus},-------------------------------------------//
        MC-TestModus ?
------{"Bahn_INIT",----------set_MC_Bahn_INIT},-----------------------------------------------// Bahn alle
```

```
        Werte auf 0
61  ->{"Bahn_AM",->->->->set_MC_Bahn_Automatik_Modus},->->->->// Bahn
        Automatic-Modus 0 - 4
62  ->{"Bahn_NS",->->->->set_MC_Bahn_NotStopp},->->->->->->// Bahn
        NotStopp
63
64  ->//@ Test-Modus
        ------------------------------------------------------------------------
        ------
65  ->{"MC_Daten_NEU",->->set_MC_Daten_NEU},->->->->->->->->// MC-Port-C
        Grundzustand
66  ->{"MC_Daten_OUT",->->set_MC_Daten_OUT},->->->->->->->->// MC-Port-C
        Output
67  ->{"MC_Daten_IN",->->->set_MC_Daten_IN}, ->->->->->->->->// MC-Port-D
        Input
68  //->{"MC_Daten_Test",->->get_MC_Daten_Test},->->->->->->->//
    MC-Daten-Test
69  ->{"MC_Port_A_00",->->->set_MC_Port_A_00},->->->->->->->->// MC-Port-A
        = 00
70  ->{"MC_Port_C_00",->->->set_MC_Port_C_00},->->->->->->->->// MC-Port-C
        = 00
71  ->{"MC_Port_D_00",->->->set_MC_Port_D_00},->->->->->->->->// MC-Port-D
        = 00
72  ->{"MC_Port_A_FF",->->->set_MC_Port_A_FF},->->->->->->->->// MC-Port-A
        = FF
73  ->{"MC_Port_C_FF",->->->set_MC_Port_C_FF},->->->->->->->->// MC-Port-C
        = FF
74  ->{"MC_Port_A",->->->->set_MC_Port_A},->->->->->->->->->->// MC-Port-A
        PinNr BitWert = 0/1, analog 1-7
75  ->{"MC_Port_C",->->->->set_MC_Port_C},->->->->->->->->->->// MC-Port-C
        PinNr BitWert = 0/1, analog 1-7
76  ->{"MC_Port_D",->->->->set_MC_Port_D},->->->->->->->->->->// MC-Port-D
        PinNr BitWert = 0/1, analog 1-7
77  //->{"MC_Port_C_Wert",->->get_MC_Daten_Test},->->->->->->->//
    MC-Daten-Test
78
79  ->//@ Bahn-Befehle
        ------------------------------------------------------------------------
        ------
80  ->{"Bl_Daten",->->->->set_MC_Bahn_Block_Daten}, ->->->->->->// Bahn
        Block-Daten = Daten für einen Blockabschnitt
81  ->{"St_Status",->->->->set_MC_Bahn_Strecken_Status},->->->->->// Bahn
        Strecken-Status-Register setzen
82  ->{"Bl_Status",->->->->set_MC_Bahn_Block_Status},->->->->->->// Bahn
        Block-Status-Register 0 - 1 setzen
83  ->{"Bl_Aktiv",->->->->set_MC_Bahn_Block_Aktiv},->->->->->->// Bahn
        Block-Aktiv-Register 0 - 2 setzen
84  ->{"Bl_Delay",->->->->set_MC_Bahn_Block_Delay},->->->->->->// Bahn alle
        Block-Strecken-Delay schalten  1..32
85  ->{"Bl_FS_max",->->->->set_MC_Bahn_Block_FS_max},->->->->->// Bahn alle
        Block-Fahrstufe max  1..32
86  ->{"Bl_FS_min",->->->->set_MC_Bahn_Block_FS_min},->->->->->// Bahn alle
        Block-Fahrstufe min  1..32
87  ->{"Bl_FS_soll",->->->->set_MC_Bahn_Block_FS_soll},->->->->// Bahn alle
        Block-Fahrstufe soll  1..32
88  ->{"Bl_FS_Delay1",->->->set_MC_Bahn_Block_FS_Delay1},->->->->// Bahn alle
        Block-Fahrstufe Delay1 schalten  1..32
89  ->{"Bl_FS_Delay2",->->->set_MC_Bahn_Block_FS_Delay2},->->->->// Bahn alle
        Block-Fahrstufe Delay2 schalten  1..32
90
91  ->//@ Weichen-Befehle
        ------------------------------------------------------------------------
        ------
92  ->{"Weichen",->->->->set_MC_Bahn_Weichen},->->->->->->->->// Bahn
        Weiche schalten [x] [y]
93  ->{"Weichen_Reset",->->->set_MC_Bahn_Weichen_Reset},->->->->// Bahn
        Weichen Reset
94  ->{"Weichen_Delay",->->->set_MC_Bahn_Weichen_Delay}, ->->->->// Bahn
        Weiche Delay*
95
96
```

```
97    ⟶//@ MC Rückmelungen an PC
          ------------------------------------------------------------------------
98    ⟶{"HELP",⟶⟶⟶⟶⟶show_help},⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶//
      MC-Befehlsliste anzeigen
99    ⟶{"?",⟶⟶⟶⟶⟶⟶⟶show_help},⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶//
      MC-Befehlsliste anzeigen
100   ⟶{"show_MC_StReg",⟶⟶⟶show_MC_SteuerRegister},⟶⟶⟶⟶⟶⟶⟶⟶⟶//
      MC-SteuerRegister
101   ⟶{"show_Bahn_REG",⟶⟶⟶show_MC_Bahn_Register},⟶⟶⟶⟶⟶⟶⟶⟶⟶//
      MC-Register anzeigen
102   ⟶{"show_Bahn_Bl_FSist",⟶show_MC_Bahn_Block_FSist},⟶⟶⟶⟶⟶⟶⟶// Bahn
      Block FS ist
103   ⟶{"Bahn_St_BM",⟶⟶⟶⟶get_MC_Bahn_Strecken_Beleg},⟶⟶⟶⟶⟶⟶⟶// Bahn
      Beleg-Messung
104   ⟶{"Bahn_St_KM",⟶⟶⟶⟶get_MC_Bahn_Strecken_Kurzschluss},⟶⟶⟶⟶⟶// Bahn
      Kurzschluss-Messung
105   //⟶{"Bahn_Bl_Test",⟶⟶⟶get_MC_Bahn_Block_Test},⟶⟶⟶⟶⟶⟶⟶// Bahn
      Beleg-TestMessung
106
107   ⟶{NULL,NULL}
108   ⟶};
109
110   //##########################################################################################
      ################mc_HELP
111
112   //@@ MC-Befehlsliste
113
114   PROGMEM const char helptext[] =
115   ⟶{
116   ⟶"\r\n"
117   ⟶"==== Microcontroler Steuerung
          ==============================================\r\n"
118   ⟶"MC_RESET ··········· - setzen MC-Reset\r\n"
119   ⟶"MC_INIT ··········· - setzen MC Grundeinstellung\r\n"
120   ⟶"MC_Ready ··········· - setzen MC betriebsbereit ?\r\n"
121   ⟶"MC_LED ··········· - setzen MC LED-Test\r\n"
122   ⟶"MC_BM 0..5 ··········· - setzen MC-Betrieb-Modus 0 - 5\r\n"
123   ⟶"MC_TM x ··········· - setzen MC-Test-Modus [0..1]\r\n"
124   ⟶"MC_RM x ··········· - setzen MC-Rückmeldung [0..1]\r\n"
125
126   ⟶"===== Test-Modus
          ============================================================\r\n"
127   ⟶"MC_StReg ··········· - setzen MC-SteuerRegister\r\n"
128   ⟶"MC_Daten_OUT ····· - setzen MC-Port-C Output\r\n"
129   ⟶"MC_Daten_IN ····· - setzen MC-Port-D Input\r\n"
130   ⟶"MC_Daten_Test ····· - setzen MC-Port-C Wert an PC senden\r\n"
131   ⟶"MC_Port_A_00 ····· - setzen MC-Port-A = 00\r\n"
132   ⟶"MC_Port_C_00 ····· - setzen MC-Port-C = 00\r\n"
133   ⟶"MC_Port_D_00 ····· - setzen MC-Port-D = 00\r\n"
134   ⟶"MC_Port_A_FF ····· - setzen MC-Port-A = FF\r\n"
135   ⟶"MC_Port_C_FF ····· - setzen MC-Port-C = FF\r\n"
136   ⟶"MC_Port_A x ····· - setzen MC-Port-A PinNr BitWert = 0/1, analog für A0 --
      A7\r\n"
137   ⟶"MC_Port_C x ····· - setzen MC-Port-C PinNr BitWert = 0/1, analog für C0 --
      C7\r\n"
138   ⟶"MC_Port_D x ····· - setzen MC-Port-D PinNr BitWert = 0/1, analog für D2 --
      D7\r\n"
139
140   ⟶"==== Weichensteuerung (BM3)
          ================================================\r\n"
141   ⟶"Weichen ··········· - setzte Weichen [x] [y]\r\n"
142   ⟶"Weichen_Delay ····· - setze Weichen-Delay\r\n"
143
144   ⟶"==== Bahnsteuerung (BM4)
          ================================================\r\n"
145   ⟶"Bahn_INIT ··········· - setzen Bahn alle Werte auf 0\r\n"
146   ⟶"Bahn_NS ··········· - setzen Bahn NotStopp\r\n"
147   ⟶"Bahn_AM ··········· - setzen Bahn AutmatikModus 0 - 4\r\n"
148   ⟶"===== Block- und Streckendaten
          ============================================\r\n"
149   ⟶"Bl_Daten ··········· - setzen Bahn Block-Daten\r\n"
```

```
150   ⟶"Bl_Status x          - setzen Bahn Block-Status-Register 0 - 1 setzen\r\n"
151   ⟶"Bl_Aktiv x           - setzen Bahn Block-Aktiv-Register 0 - 2 setzen\r\n"
152   ⟶"Bl_Delay xx          - setzen Bahn Block-Delay (32 Werte 0 - 99)\r\n"
153   ⟶"Bl_FS_max xx         - setzen Bahn Block FS-max (32 Werte 0 - 80)\r\n"
154   ⟶"Bl_FS_min xx         - setzen Bahn Block FS-min (32 Werte 0 - 80)\r\n"
155   ⟶"Bl_FS_soll xx        - setzen Bahn Block FS-soll (32 Werte 0 -80)\r\n"
156   ⟶"Bl_FS_Delay1 xx      - setzen Bahn Block Delay1 = beschleinigen (32 Werte 0 -
          99)\r\n"
157   ⟶"Bl_FS_Delay2 xx      - setzen Bahn Block Delay2 = bremsen (32 Werte 0 - 99)\r\n"
158   ⟶"Bl_FSist             - setzen Block FS ist\r\n"
159   ⟶"St_Status            - setzen Bahn Strecken-Status-Register setzen\r\n"
160   ⟶"St_BM                - setzen Bahn Block Beleg-Messung\r\n"
161   ⟶"St_KM                - setzen Bahn Block Kurzschluss-Messung\r\n"
162   ⟶"BR                   - setzen Bahn Register\r\n"
163   ⟶
164   ⟶"===== MC Rückmeldungen
          ========================================================\r\n"
165   ⟶"HELP                 - Rückmeldung MC-Hilfe-Liste\r\n"
166   ⟶"?                    - Rückmeldung MC-Hilfe-Liste\r\n"
167   ⟶"show_MC_StReg        - Rückmeldung MC-SteuerRegister\r\n"
168   ⟶"show_Bahn_REG        - Rückmeldung MC-Bahn_Register\r\n"
169   ⟶"show_Bahn_Bl_FSist   - Rückmeldung MC-Bahn Block FS ist\r\n"
170   ⟶"Bahn_St_BM           - Rückmeldung MC-Bahn Beleg-Messung\r\n"
171   ⟶"Bahn_St_KM           - Rückmeldung MC-Bahn Kurzschluss-Messung\r\n"
172   //⟶{"Bahn_Bl_Test",      get_MC_Bahn_Block_Test},⟶⟶⟶⟶// Bahn
      Beleg-TestMessung
173
174   ⟶
175   ⟶};
176
177   //#############################################################################################
      ################################
178   //#############################################################################################
      ################################
179   //
180   // MC Hilfs-Routinen
181   //
182   //#############################################################################################
      ################################
183   //#############################################################################################
      ################################
184
185   //#############################################################################################
      ################################
186   //⟶Kommando auswerten - Werte die Kommandotabelle COMMAND_STRUCTUR COMMAND_TABELLE
      aus
187   //-------------------------------------------------------------------------------------------
      --------------------------------
188   unsigned char extract_cmd (char *string_pointer)
189   ⟶{
190   ⟶//Stringzeiger;
191   ⟶char ⟶⟶⟶*string_pointer_tmp;
192   ⟶unsigned char ⟶cmd_index = 0;
193   ⟶unsigned char ⟶VarNr = 1;
194   //⟶unsigned ⟶⟶CheckSummeExtrakt = 0;
195
196   ⟶// alle Variablen[..] = 0 -- A.Walder 6-6-2017
197   ⟶for (VarNr = 1; VarNr < MAX_VAR; VarNr++)
198   ⟶⟶variable[VarNr] = 0;
199
200   ⟶string_pointer_tmp = strsep(&string_pointer," ");
201   ⟶//Kommando in Tabelle suchen
202   ⟶while(strcasecmp(COMMAND_TABELLE[cmd_index].cmd,string_pointer_tmp))
203   ⟶⟶{
204   ⟶⟶if (COMMAND_TABELLE[++cmd_index].cmd == 0) ⟶⟶⟶//Abruch Whileschleife
          und Unterprogramm verlassen
205   ⟶⟶⟶return(0);
206   ⟶⟶}
207   ⟶//Variablen finden und auswerten
208   ⟶for (VarNr = 1; VarNr < MAX_VAR; VarNr++)
209   ⟶⟶{
```

```
210         string_pointer_tmp = strsep(&string_pointer,", ");
211     ->->variable[VarNr] = strtol(string_pointer_tmp,NULL,0);
212     ->->}
213
214     /*
215     ->// MC-Rückmeldung -- Anzeige der Varibalen (Walder) ----------------------------
216     ->if (MC_Responce_Modus == 1)
217     ->->{
218     ->->CheckSummeExtrakt = 0;
219     ->->usart_write("Extact_Command ");
220     ->->for (VarNr = 1; VarNr < MAX_VAR; VarNr++)
221     ->->->{
222     ->->->usart_write("%3i ",variable[VarNr]);
223     ->->->if ((VarNr % 8 == 0) | (VarNr % 16 == 0) | (VarNr % 24 == 0) | (VarNr %
              32 == 0))
224     ->->->->{usart_write(".");};
225     ->->->CheckSummeExtrakt = CheckSummeExtrakt + variable[VarNr];
226     ->->->};
227     ->->usart_write("  MC-CS = %i \r\n", (CheckSummeExtrakt / 2));
228     ->->}
229     ->//-----------------------------------------------------------------
230     */
231       //Kommando ausführen
232     ->COMMAND_TABELLE[cmd_index].fp();
233
234     ->return(1);
235     ->}
236     ->
237     //#############################################################################################
    ################################
238     //  Pause
239     //-----------------------------------------------------------------------------------------
    --------------------------------
240     void long_delay(int ms)
241     ->{
242     ->if (MC_Betrieb_Modus != 4)
243     ->->for(; ms > 0; ms--)
244     ->->->_delay_ms(1);
245     ->}
246
247
248
249     //#############################################################################################
    ################################
250     //#############################################################################################
    ################################
251     //
252     // MC Start
253     //
254     //#############################################################################################
    ################################
255     //#############################################################################################
    ################################
256
257     //@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
    @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
258     //@ RESET = MC Reset ausführen
259     //@ PC ==> Reset
260     //@-----------------------------------------------------------------------------------------
    --------------------------------
261     void set_MC_Reset(void)
262     ->{
263     ->usart_write("\r\n");
264     ->usart_write("-----------------\r\n");
265     ->usart_write("MC-Reset !!\r\n");
266     ->usart_write("-----------------\r\n\r\n");
267     ->RESET();
268     ->asm("nop");->->->->->->->->->->->->->->->->->->->->->->->->->->->->->->->->->->->->->->->->->->->->
        // !!!! mindestens 4 * NOP
269     ->asm("nop");
270     ->asm("nop");
```

```
271     asm("nop");
272     asm("nop");
273     asm("nop");
274     
275     set_MC_INIT();                                                  //
            MC-INIT ==>  Werte in Grundstellung
276     }
277     
278     //@#########################################################################################
        ################################
279     //@ MC-INIT ==>  Werte in Grundstellung
280     //@ MC-Port A,C,D = output in Grundstellung
281     //@-----------------------------------------------------------------------------------------
        --------------------------------
282     void set_MC_INIT()
283     {
284     // MC Einstelungen
            ------------------------------------------------------------------------
285     MC_Betrieb_Modus = 0;                                           //
            MC-BetriebsModus = 0
286     MC_Responce_Modus = 1;                                          //
            MC-RückmeldungModus = 0 keine Rückmeldung
287     MC_Test_Modus = 0;                                              // MC-TestModus
            = 0
288     
289     // MC1 Bahn-Einstellungen
            ------------------------------------------------------------------------
290     Bahn_Automatik_Modus = 0;                                       //
            MC-AutomatikModus = 0
291     
292     // MC2 Weichen-Einstellungen
            ------------------------------------------------------------------------
293     Weichen_Delay = 1;                                              // Delay für
            Weichen-Rückstellung
294     Weichen_TimerDelay = 1;                                         // Delay für
            Weichen-Rückstellung
295     Weichen_MC_TimerDelay = 1;                                      // Delay für
            Weichen-Rückstellung
296     Weichen_NEU = 0;                                                // Flag, wenn
            neue Weichenstellung dann 1
297     
298     //@ MC-Register in Grundstellung
            ================================================================================
            =====
299     
300     //@ MC-Port-D Steuerung ----------------------------------------------
301     DDRD = 0xFF;                                                    // MC-Port-D
            ==>Steuerung Output
302     asm("nop");
303     PORTD = 0b11011000;                                             // MC-Port-D
            => [11_1.1___]
304                                                                     // MC-Port-D7
                                                                        => [1___.____] =>
                                                                        Interface
                                                                        74LS154 Encoder
                                                                        kein CS für
                                                                        Blockplatine
305                                                                     // MC-Port-D6
                                                                        => [_1__.____] =>
                                                                        Interface
                                                                        74LS245 ??
306                                                                     // MC-Port-D4
                                                                        => [___1.____] =>
                                                                        Interface
                                                                        74LS245 Daten
                                                                        nicht aktiv
307                                                                     // MC-Port-D3
                                                                        =>
```

```
                                                                [____.1___] => Interface 74LS245 Daten output
308     asm("nop");
309
310     //@ MC-Port-A Adressen ----------------------------------------------------
311     DDRA = 0xFF;                                    // MC-Port-A Adressen Output
312     asm("nop");
313     PORTA = 0x00;                                   // MC-Port-A Adressen = [00]
314     asm("nop");
315
316     //@ MC-Port-C Daten -------------------------------------------------------
317     DDRC = 0xFF;                                    // MC-Port-C Control Output
318     asm("nop");
319     PORTC = 0x00;                                   // MC-Port-C Daten = [00]
320     asm("nop");
321
322     // MC-Rückmeldung --- MC Steuer-Register anzeigen ---------------------------------
323     usart_write("\r\n");
324     usart_write("-------------------------------------------------------------------------------\r\n");
325     usart_write("MC-INIT !!\r\n");
326     usart_write("MC-Eisenbahnprogramm (c) Achim Walder & Andreas Lutters ==> 2020-11-25\r\n");
327     usart_write("-------------------------------------------------------------------------------\r\n");
328     usart_write("MC-Port-D = [11_1.1___]\r\n");
329     usart_write("MC-Port-A = [0000.0000]\r\n");
330     usart_write("MC-Port-C = [0000.0000]\r\n");
331     usart_write("MC-Register --> BM = %1i, NS = %1i, RM = %1i, X, X, AM-HS = %1i, AM-SB = %1i, AM-NS = %1i, AM-GB = %1i, AM-St = %1i, X, X \r\n",
332     MC_Betrieb_Modus, MC_NotStopp, MC_Responce_Modus, Bahn_Automatik_HS_Modus, Bahn_Automatik_SB_Modus, Bahn_Automatik_NS_Modus, Bahn_Automatik_GB_Modus, Bahn_Automatik_St_Modus);
333     }
334
335     //#############################################################################################################
336     // MC-OK - ist MC betriebsbereit
        ? ------------------------------------------------------------------------------------------
337     // PC ==> MC
338     //-------------------------------------------------------------------------------------------------------------
339     void set_MC_ready(void)
340     {
341     usart_write("\r\n");
342     usart_write("---------------\r\n");
343     usart_write("MC-Ready !!\r\n");
344     usart_write("---------------\r\n\r\n");
345     return;
346     }
347
348
349     //#############################################################################################################
350     //#############################################################################################################
351     //
352     //      MC Steuerung
353     //
354     //#############################################################################################################
355     //#############################################################################################################
```

```
################################


//########################################################################################################################
// MC SteuerRegister setzen
//--------------------------------------------------------------------------------------------------------------------------
void set_MC_SteuerRegister(void)
	{
	MC_Betrieb_Modus					= variable[1];
	MC_NotStopp						= variable[2];
	MC_Responce_Modus					= variable[3];
	Bahn_Automatik_HS_Modus 				= variable[6];
	Bahn_Automatik_SB_Modus				= variable[7];
	Bahn_Automatik_NS_Modus				= variable[8];
	Bahn_Automatik_GB_Modus				= variable[9];
	Bahn_Automatik_St_Modus				= variable[10];
	
	// // Startwert setzten ! ----------------------
	if (MC_Betrieb_Modus == 4)
		{
		zeile = 1;
		spalte = 1;
		Peek = 1;
		};

	// MC-Rückmeldung --- MC Steuer-Register anzeigen ---------------------------------
	usart_write("MC-Register --> BM = %1i, NS = %1i, RM = %1i, X, X, AM-HS = %1i, AM-SB = %1i, AM-NS = %1i, AM-GB = %1i, AM-St = %1i, X, X \r\n", 
	MC_Betrieb_Modus, MC_NotStopp, MC_Responce_Modus, Bahn_Automatik_HS_Modus, Bahn_Automatik_SB_Modus, Bahn_Automatik_NS_Modus, Bahn_Automatik_GB_Modus, Bahn_Automatik_St_Modus);
	return;
	}


//##########################################################################################################################################
// MC-Betrieb-Modus einstellen						21-12-2013
// PC ==> BM 0, BM 1, BM 2, BM 3, BM 4
//--------------------------------------------------------------------------------------------------------------------------
void set_MC_Betrieb_Modus(void)
	{
	set_MC_Bahn_INIT();						// Bahn-INIT MC-Betriebsmodus = 0, Bahn-Autoamtik-Modus = 0
	MC_Betrieb_Modus = variable[1];

	switch (MC_Betrieb_Modus)								
		{
		case 0 : usart_write("MC-Betrieb = 0\r\n");break;
		case 1 : usart_write("MC-Betrieb = 1\r\n");break;
		case 2 : usart_write("MC-Betrieb = 2\r\n");break;
		case 3 : usart_write("MC-Betrieb = 3\r\n");break;
		case 4 : usart_write("MC-Betrieb = 4\r\n");break;
		};

	// // Startwert setzten ! ----------------------
	if (MC_Betrieb_Modus == 4)								
		{
		zeile = 1;
		spalte = 1;
		Peek = 1;
		};					

	// MC-Rückmeldung --- MC Steuer-Register anzeigen ---------------------------------
	usart_write("MC-Register --> BM = %1i, NS = %1i, RM = %1i, X, X, AM-HS = %1i,
```

```
	AM-SB = %1i, AM-NS = %1i, AM-GB = %1i, AM-St = %1i, X, X \r\n",
	MC_Betrieb_Modus, MC_NotStopp, MC_Responce_Modus, Bahn_Automatik_HS_Modus, Bahn_Automatik_SB_Modus, Bahn_Automatik_NS_Modus, Bahn_Automatik_GB_Modus, Bahn_Automatik_St_Modus);
	return;
	}

//#########################################################################################################################
//
MC-Rückmeldungen										18. März 2012
// PC ==> RC 0 oder RC 1
//---------------------------------------------------------------------------------------------------------------------------
void set_MC_Response_Modus(void)
	{
	MC_Responce_Modus = variable[1];
	if (MC_Responce_Modus == 0)
		{usart_write("MC-Responce-Modus = aus\r\n");}
	else
		{usart_write("MC-Responce-Modus = ein\r\n");}

	// MC-Rückmeldung --- MC Steuer-Register anzeigen ----------------------------------
	usart_write("MC-Register --> BM = %1i, NS = %1i, RM = %1i, X, X, AM-HS = %1i, AM-SB = %1i, AM-NS = %1i, AM-GB = %1i, AM-St = %1i, X, X \r\n",
	MC_Betrieb_Modus, MC_NotStopp, MC_Responce_Modus, Bahn_Automatik_HS_Modus, Bahn_Automatik_SB_Modus, Bahn_Automatik_NS_Modus, Bahn_Automatik_GB_Modus, Bahn_Automatik_St_Modus);
	return;
	}

//#########################################################################################################################
// MC-Test
// Test Rückmeldungen
//---------------------------------------------------------------------------------------------------------------------------
void set_MC_Test_Modus(void)
	{
	MC_Test_Modus = variable[1];
	if (MC_Test_Modus == 0)
		{usart_write("MC-Test-Modus = aus\r\n");}
	else
		{usart_write("MC-Test-Modus = ein\r\n");}

	// MC-Rückmeldung --- MC Steuer-Register anzeigen ----------------------------------
	usart_write("MC-Register --> BM = %1i, NS = %1i, RM = %1i, X, X, AM-HS = %1i, AM-SB = %1i, AM-NS = %1i, AM-GB = %1i, AM-St = %1i, X, X \r\n",
	MC_Betrieb_Modus, MC_NotStopp, MC_Responce_Modus, Bahn_Automatik_HS_Modus, Bahn_Automatik_SB_Modus, Bahn_Automatik_NS_Modus, Bahn_Automatik_GB_Modus, Bahn_Automatik_St_Modus);
	return;
	}


//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//
// Textroutinen für Platinentest
//
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
```

```
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ HELP = Hilfetext Ausgabe
//@ PC ==> Help oder ?
void show_help (void)
	{
	#if HELPTEXT
		unsigned char data;
		PGM_P helptest_pointer = helptext;
		do
			{
			data = pgm_read_byte(helptest_pointer++);
			usart_write("%c",data);
			}
		while(data != 0);
	#endif
	}


//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ MC LED-Test - alle LEDs einmal ein und aus 								- 10. November 2012
void set_MC_LED(void)
	{
	unsigned int Pin;
//	set_MC_Bahn_NotStopp();								// Bahn-Notstopp
	MC_Responce_Modus = 1; 								// MC-Rückmeldung ein
	MC_Betrieb_Modus = 1; 								// Muss gesetzt sein, damit es überhaupt eine Reaktion gibt!
	// MC-Rückmeldung -------------------------------------
	usart_write("\r\n");
	usart_write("---------------\r\n");
	usart_write("MC-LED-Test\r\n");
	usart_write("---------------\r\n\r\n");

	// MC-Port-D Steuerleitungen -------------------------------------------------
	for( Pin = 2; Pin < 8; Pin++)
		{
		PORTD |= _BV(Pin);
		long_delay(10);
		}
	for( Pin = 2; Pin < 8; Pin++)
		{
		PORTD &= 255-_BV(Pin);
		long_delay(10);
		}
	// MC-Port-D in Grundstellung
	DDRD = 0xFF;																	// MC-Port-D ==>Steuerung Output
	asm("nop");
	PORTD = 0b11011000;															// MC-Port-D => [11_1.1___]

	// MC-Port-C Datenleitungen ---------------------------------------------------
	for( Pin = 0; Pin < 8; Pin++)
		{
		PORTC |= _BV(Pin);
		long_delay(10);
		}
	for( Pin = 0; Pin < 8; Pin++)
		{
		PORTC &= 255-_BV(Pin);
		long_delay(10);
		}

	// MC-Port-A Datenleitungen ---------------------------------------------------
	for( Pin = 0; Pin < 8; Pin++)
		{
		PORTA |= _BV(Pin);
		long_delay(10);
```

```c
		}
	for( Pin = 0; Pin < 8; Pin++)
		{
		PORTA &= 255-_BV(Pin);
		long_delay(10);
		}

	set_MC_INIT();											// MC in Grundstellung Port A,C,D = output
	}

//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@
//@ MC Testprogramme
//@
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ setzte MC-Daten NEU
void set_MC_Daten_NEU(void)
	{
	if (MC_Betrieb_Modus < 3)
		{
		PORTD |= _BV(7);									// MC-Port-D7 auf 1 => Encoder-74LS154 nicht aktiv, kein CS
		PORTD |= _BV(4);									// MC-Port-D3 auf 1 => Interface-74LS245 nicht aktiv
		PORTD |= _BV(3);									// MC-Port-D3 auf 1 => Interface-74LS245 Daten output
		DDRC = 0xFF;										// MC-Port-C output
		usart_write("MC-Daten-Grundzustand\r\n");
		}
	else
		usart_write("==> ERROR !! BM > 3 \r\n");
	}
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ setzte MC-Daten Output
void set_MC_Daten_OUT(void)
	{
	if (MC_Betrieb_Modus < 3)
		{
		PORTD |= _BV(7);									// MC-Port-D7 auf 1 => Encoder-74LS154 nicht aktiv, kein CS
		PORTD |= _BV(3);									// MC-Port-D3 auf 1 => Interface-74LS245 Daten output
		PORTD &= 255-_BV(4);								// MC-Port-D4 auf 0 => Interface-74LS245 Daten aktiv
		DDRC = 0xFF;										// MC-Port-C output
		usart_write("MC-Daten-Output\r\n");	
		}
	else
		usart_write("==> ERROR !! BM > 3 \r\n");
	}
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ setzte MC-Daten Input
void set_MC_Daten_IN(void)
	{
	if (MC_Betrieb_Modus < 3)
		{
		PORTD |= _BV(7);									// MC-D7 auf 1 => 74LS154 Encoder nicht aktiv, kein CS
		DDRC = 0x00; 										// MC-Port-C (Daten) als Eingang
		PORTC = 0xFF; 										//
```

```
		MC-Port-C (Daten) als Eingang - with pull-ups enabled
		PORTD &= 255-_BV(3);											// MC-D3 auf 0 => 74LS245 Interface-Daten input
		PORTD &= 255-_BV(4);											// MC-D4 auf 0 => 74LS245 Interface-Daten aktiv
		usart_write("MC-Daten-Input\r\n");	
		}
	else
		usart_write("==> ERROR !! BM > 3 \r\n");
	}
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ get MC-Daten Test
void get_MC_Daten_Test(void)
	{
	unsigned int Port_C_Wert;
	if (MC_Betrieb_Modus < 3)
		{
		Port_C_Wert = PINC;												// Einlesen von MC-Port-C
		usart_write("MC-Daten = %3i \r\n",Port_C_Wert);						// sendet MC-Wert an PC
		}
	else
		usart_write("==> ERROR !! BM > 3 \r\n");
	}
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ set MC-Port-A
void set_MC_Port_A(void)
	{
	unsigned char APinNr;
	unsigned char BitWert;
	APinNr = variable[1];
	BitWert = variable[2];
	if (MC_Betrieb_Modus < 3)
		{
		if (APinNr < 8)
			{
			if (BitWert == 1)
				{
				DDRA = 0xFF;
				PORTA |= _BV(APinNr);										// ein MC-Port-A x = 1
				usart_write("MC-Port-A %1i = [1]\r\n", APinNr);
				}
			else
				{
				DDRA = 0xFF;
				PORTA &= 255 - _BV(APinNr);									// ein MC-Port-A x = 0
				usart_write("MC-Port-A %1i = [0]\r\n", APinNr);
				}
			}	
		}
	else
		{
		usart_write("==> ERROR !! BM > 3 \r\n");
		}
	}
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ set MC-Port-C
void set_MC_Port_C(void)
	{
	unsigned char CPinNr;
	unsigned char BitWert;
	CPinNr = variable[1];
	BitWert = variable[2];
	if (MC_Betrieb_Modus < 3)
		{
```

```
		if (CPinNr < 8)
			{
			if (BitWert == 1)
				{
				DDRC = 0xFF;
				PORTC |= _BV(CPinNr);											// MC-Port-C x = 1
				usart_write("MC-Port-C %1i = [1]\r\n", CPinNr);
				}
			else
				{
				DDRC = 0xFF;
				PORTC  &= 255 - _BV(CPinNr);										// MC-Port-C x = 0
				usart_write("MC-Port-C %1i = [0]\r\n", CPinNr);
				}
			}	
		}	
	else
		usart_write("==> ERROR !! BM > 3 \r\n");
	}
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ set MC-Port-D (PIN-D0 und PIN-D1 werden nicht geschaltet)
void set_MC_Port_D(void)
	{
	unsigned char DPinNr;
	unsigned char BitWert;
	DPinNr = variable[1];
	BitWert = variable[2];
	if (MC_Betrieb_Modus < 3)
		{
		if ((DPinNr > 1) &  (DPinNr < 8))
			{
			if (BitWert == 1)
				{
				DDRD = 0xFF;
				PORTD |= _BV(DPinNr);											// MC-Port-D x = 1
				usart_write("MC-Port-D %1i = [1]\r\n", DPinNr);
				}
			else
				{
				DDRD = 0xFF;
				PORTD  &= 255 - _BV(DPinNr);										// MC-Port-D x = 0
				usart_write("MC-Port-D %1i = [0]\r\n", DPinNr);
				}
			}	
		}
	else
		usart_write("==> ERROR !! BM > 3 \r\n");
	}
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ setzte MC-Port-A auf 00
void set_MC_Port_A_00(void)
	{
	if (MC_Betrieb_Modus < 3)
		{
		DDRA = 0xFF;
		PORTA = 0x00; 												// MC-Port-A = 00
		usart_write("MC-Port-A = [00] \r\n");
		}
	else
		usart_write("==> ERROR !! BM > 3 \r\n");
	}
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ setzte MC-Port-C auf 00
```

```
void set_MC_Port_C_00(void)
    {
    if (MC_Betrieb_Modus < 3)
        {
        DDRC = 0xFF;
        PORTC = 0x00;                                                       // MC-Port-C = 00
        usart_write("MC-Port-C = [00] \r\n");
        }
    else
        usart_write("==> ERROR !! BM > 3 \r\n");
    }
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ setzte MC-Port-D auf 00
void set_MC_Port_D_00(void)
    {
    if (MC_Betrieb_Modus < 3)
        {
        DDRD = 0xFF;
        PORTD = 0x00;                                                       // MC-Port-D = 00
        usart_write("MC-Port-D = [00] \r\n");
        }
    else
        usart_write("==> ERROR !! BM > 3 \r\n");
    }
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ setzte MC-Port-A auf FF
void set_MC_Port_A_FF(void)
    {
    if (MC_Betrieb_Modus < 3)
        {
        DDRA = 0xFF;
        PORTA = 0xFF;                                                       // MC-Port-A = FF
        usart_write("MC-Port-A = [FF] \r\n");
        }
    else
        usart_write("==> ERROR !! BM > 3 \r\n");
    }
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ setzte MC-Port-C auf FF
void set_MC_Port_C_FF(void)
    {
    if (MC_Betrieb_Modus < 3)
        {
        DDRC = 0xFF;
        PORTC = 0xFF;                                                       // MC-Port-C = FF
        usart_write("MC-Port-C = [FF] \r\n");
        }
    else
        usart_write("==> ERROR !! BM > 3 \r\n");
    }
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ setzte MC-Port-A auf xx
void set_register_Port_A(void)
    {
    unsigned int WertA;
    WertA = variable[1];
    if (MC_Betrieb_Modus < 3)
        {
        usart_write("MC-Port-A = [%2i]\r\n",WertA);
        DDRA &= 0xFF;
        PORTA = WertA;
        }
    else
```

```
		usart_write("==> ERROR !! BM > 3 \r\n");
	}
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ setzte MC-Port-C auf xx
void set_register_Port_C(void)
	{
	unsigned int WertC;
	WertC = variable[1];
	if (MC_Betrieb_Modus < 3)
		{
		usart_write("MC-Port-C = [%2i]\r\n",WertC);
		DDRC &= 0xFF;
		PORTC = WertC;
		}
	else
		usart_write("==> ERROR !! BM > 3 \r\n");
	}
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@ setzte MC-Port-D auf xx
void set_register_Port_D(void)
	{
	unsigned int WertD;
	WertD = variable[1];
	if (MC_Betrieb_Modus < 3)
		{
		usart_write("MC-Port-D = [%2i]\r\n",WertD);
		DDRD &= 0xFF;
	  PORTD = WertD;
		}
	else
		usart_write("==> ERROR !! BM > 3 \r\n");
	}

//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//
// Bahnbetrieb (BM = 4)
//
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@


//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
// MC-SteuerRegister anzeigen
void show_MC_SteuerRegister(void)
	{
	// MC-Rückmeldung --- MC Steuer-Register anzeigen ----------------------------------
	usart_write("MC-Register --> BM = %1i, NS = %1i, RM = %1i, X, X, AM-HS = %1i, AM-SB = %1i, AM-NS = %1i, AM-GB = %1i, AM-St = %1i, X, X \r\n",
			MC_Betrieb_Modus, MC_NotStopp, MC_Responce_Modus, Bahn_Automatik_HS_Modus, Bahn_Automatik_SB_Modus, Bahn_Automatik_NS_Modus,
			Bahn_Automatik_GB_Modus, Bahn_Automatik_St_Modus);
	}
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
// MC-Register anzeigen
void show_MC_Bahn_Register(void)
	{
	unsigned char TT;  // interner Zähler

	// Steuer-Register ----------------------------------
	usart_write("\r\n");
```

```
831     ---->// MC-Rückmeldung --- MC Steuer-Register anzeigen
            ---------------------------------
832     ---->usart_write("MC-Register --> BM = %1i, NS = %1i, RM = %1i, X, X, AM-HS = %1i,
            AM-SB = %1i, AM-NS = %1i, AM-GB = %1i, AM-St = %1i, X, X \r\n",
833     ---->MC_Betrieb_Modus, MC_NotStopp, MC_Responce_Modus, Bahn_Automatik_HS_Modus,
            Bahn_Automatik_SB_Modus, Bahn_Automatik_NS_Modus, Bahn_Automatik_GB_Modus,
            Bahn_Automatik_St_Modus);
834     ---->// Strecken-Status-Register ---------------------------------
835     ---->usart_write("=> St-Status I..");
836     ---->for (TT = 1; TT <= 16; TT++)
837     ---->---->{
838     ---->---->usart_write("%3i ",Strecken_Status_Register[TT]);
839     ---->---->if ((TT % 8) == 0)
840     ---->---->  {usart_write(" ");};
841     ---->---->};
842     ---->usart_write("\r\n");
843     ---->usart_write("St-Status II.");
844     ---->for (TT = 17; TT <= 32; TT++)
845     ---->---->{
846     ---->---->usart_write("%3i ",Strecken_Status_Register[TT]);
847     ---->---->if ((TT % 24) == 0)
848     ---->---->  {usart_write(" ");};
849     ---->---->};
850     ---->usart_write("\r\n");
851     ---->// Block Nummer ---------------------------------
852     ---->usart_write("Bl-Nr........ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16  17
            18 19 20 21 22 23 24  25 26 27 28 29 30 31 32 \r\n");
853     ---->// Block-Status-Register ---------------------------------
854     ---->usart_write("Bl-Status....");
855     ---->for (TT = 1; TT < 33; TT++)
856     ---->---->{
857     ---->---->usart_write(" %1i ",Block_Status_Register[TT]);
858     ---->---->if (((TT % 8) == 0) && (TT < 32))
859     ---->---->  {usart_write(" ");};
860     ---->---->};
861     ---->usart_write("\r\n");
862     ---->// Block-Aktiv-Register ---------------------------------
863     ---->usart_write("Bl-Aktiv.....");
864     ---->for (TT = 1; TT < 33; TT++)
865     ---->---->{
866     ---->---->usart_write(" %1i ",Block_Aktiv_Register[TT]);
867     ---->---->if (((TT % 8) == 0) && (TT < 32))
868     ---->---->  {usart_write(" ");};
869     ---->---->};
870     ---->usart_write("\r\n");
871     ---->// Block-Delay-Register ---------------------------------
872     ---->usart_write("Bl-Delay.....");
873     ---->for (TT = 1; TT < 33; TT++)
874     ---->---->{
875     ---->---->usart_write("%2i ",Block_Delay_Register[TT]);
876     ---->---->if (((TT % 8) == 0) && (TT < 32))
877     ---->---->  {usart_write(" ");};
878     ---->---->};
879     ---->usart_write("\r\n");
880     ---->// Block-FS-max-Register ---------------------------------
881     ---->usart_write("Bl-FS-max....");
882     ---->for (TT = 1; TT < 33; TT++)
883     ---->---->{
884     ---->---->usart_write("%2i ",Block_FS_max_Register[TT]);
885     ---->---->if (((TT % 8) == 0) && (TT < 32))
886     ---->---->  {usart_write(" ");};
887     ---->---->};
888     ---->usart_write("\r\n");
889     ---->// Block-FS-min-Register ---------------------------------
890     ---->usart_write("Bl-FS-min....");
891     ---->for (TT = 1; TT < 33; TT++)
892     ---->---->{
893     ---->---->usart_write("%2i ",Block_FS_min_Register[TT]);
894     ---->---->if (((TT % 8) == 0) && (TT < 32))
895     ---->---->  {usart_write(" ");};
896     ---->---->};
```

```c
	usart_write("\r\n");
	// Block-FS-Delay1-Register ----------------------------------
	usart_write("Bl-FS-Delay1.");
	for (TT = 1; TT < 33; TT++)
		{
		usart_write("%2i ",Block_FS_Delay1_Register[TT]);
		if (((TT % 8) == 0) && (TT < 32))
		  {usart_write(" ");};
		};
	usart_write("\r\n");
	// Block-FS-Delay2-Register ----------------------------------
	usart_write("Bl-FS-Delay2.");
	for (TT = 1; TT < 33; TT++)
		{
		usart_write("%2i ",Block_FS_Delay2_Register[TT]);
		if (((TT % 8) == 0) && (TT < 32))
		  {usart_write(" ");};
		};
	usart_write("\r\n");
	// Block-FS-soll-Register ----------------------------------
	usart_write("Bl-FS-soll...");
	for (TT = 1; TT < 33; TT++)
		{
		usart_write("%2i ",Block_FS_soll_Register[TT]);
		if (((TT % 8) == 0) && (TT < 32))
		  {usart_write(" ");};
		};
	usart_write("\r\n");
	// Block-FS-ist-Register ----------------------------------
	usart_write("Bl-FS-T-ist..");
	for (TT = 1; TT < 33; TT++)
		{
		usart_write("%2i ",Timer_Block_FS_ist_Register[TT]);
		if (((TT % 8) == 0) && (TT < 32))
		  {usart_write(" ");};
		};
	usart_write("\r\n");
	}

//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
// Bahn Block FS ist Register senden
void show_MC_Bahn_Block_FSist(void)
	{
	unsigned char TT;  // interner Zähler

	usart_write("Bl-Nr........ 1  2  3  4  5  6  7  8   9 10 11 12 13 14 15 16  17 18 19 20 21 22 23 24  25 26 27 28 29 30 31 32 \r\n");
	usart_write("Bl-FS-T-ist..");
	for (TT = 1; TT < 33; TT++)
		{
		usart_write("%2i ",Timer_Block_FS_ist_Register[TT]);
		if (((TT % 8) == 0) && (TT < 32))
		  {usart_write(" ");};
		};
	usart_write("\r\n");
	}






//#####################################################################################################################
//#####################################################################################################################
//
// Bahn-Routinen
//
// diese Routinen laufen nur im BM 0
```

```
964   //
965   //#########################################################################################
      ##################################
966   //#########################################################################################
      ##################################
967
968
969   //#########################################################################################
      ###############################################
970   // AM = Automatic-Modus einstellen----------------------------------------------------
      21-12-2013
971   // PC ==> AM 0, AM 1, AM 2, AM 4
972   //-----------------------------------------------------------------------------------
      --------------------------------
973   void set_MC_Bahn_Automatik_Modus(void)
974   ----{
975   ----Bahn_Automatik_Modus = variable[1];
976   ----switch (Bahn_Automatik_Modus)
977   ----------{
978   ----------case 0 : usart_write("Bahn-Automatik = 0\r\n");break;
979   ----------case 1 : usart_write("Bahn-Automatik = 1\r\n");break;
980   ----------case 2 : usart_write("Bahn-Automatik = 2\r\n");break;
981   ----------case 3 : usart_write("Bahn-Automatik = 3\r\n");break;
982   ----------case 4 : usart_write("Bahn-Automatik = 4\r\n");break;
983   ----------case 5 : usart_write("Bahn-Automatik = 5\r\n");break;
984   ----------}----
985   ----// Steuer-Register Anzeigen ---------------------------------
986   ----// MC-Rückmeldung --- MC Steuer-Register anzeigen
          ---------------------------------
987   ----usart_write("MC-Register --> BM = %1i, NS = %1i, RM = %1i, X, X, AM-HS = %1i,
          AM-SB = %1i, AM-NS = %1i, AM-GB = %1i, AM-St = %1i, X, X \r\n",
988   ----MC_Betrieb_Modus, MC_NotStopp, MC_Responce_Modus, Bahn_Automatik_HS_Modus,
          Bahn_Automatik_SB_Modus, Bahn_Automatik_NS_Modus, Bahn_Automatik_GB_Modus,
          Bahn_Automatik_St_Modus);
989   ----return;----------------------------------------
990   ----}----
991   ----
992
993   //#########################################################################################
      ##################################
994   // Bahn_NS = Bahn-Notstop ausführen ausführen
995   //-----------------------------------------------------------------------------------
      --------------------------------
996   void set_MC_Bahn_NotStopp(void)
997   ----{
998   ----unsigned char----Block_Nr, FahrStufe_Nr, St_Karten_Nr;
999
1000  ----// alle MC-Timer-Register = 0 ---------------------------------------
1001  ----for (Block_Nr = 1; Block_Nr <= 32; Block_Nr++)----------// Block-Nr
          [1..32] - Regelung
1002  ----------{
1003  ----------Timer_Block_FS_soll_Register[Block_Nr] = 0;
1004  ----------Timer_Block_FS_ist_Register[Block_Nr] =0;
1005  ----------Timer_Block_FS_Delay_Register[Block_Nr] = 0;
1006  ----------}----
1007
1008  ----// alle MC-Fahrstufen-Register = 0
          ---------------------------------------------------------------
1009  ----for (FahrStufe_Nr = 0; FahrStufe_Nr < Fahrstufen_MAX; FahrStufe_Nr++)----// alle
          Fahrstufen-Register = 0
1010  ----------{
1011  ----------FahrStufe[FahrStufe_Nr][0] = 0;
1012  ----------FahrStufe[FahrStufe_Nr][1] = 0;
1013  ----------FahrStufe[FahrStufe_Nr][2] = 0;
1014  ----------FahrStufe[FahrStufe_Nr][3] = 0;
1015  ----------}
1016  ----
1017  ----MC_Responce_Modus = 0;----------// MC-Rückmeldung = 0
1018  ----MC_Test_Modus = 0;----------// MC-Test = 0
1019  ----Bahn_Automatik_Modus = 0;----------// Bahn-Automatic = 0
1020
```

```
1021    ⟶// alle MC-Register = 0 ----------------------------------------
1022    ⟶for (Block_Nr = 1; Block_Nr < 33; Block_Nr++)  ⟶⟶⟶⟶
1023    ⟶⟶{
1024    ⟶⟶Block_Status_Register[Block_Nr] = 0;⟶⟶⟶⟶⟶⟶⟶⟶// alle
            Block-FS-soll = 0
1025    ⟶⟶Block_Aktiv_Register[Block_Nr] = 0;⟶⟶⟶⟶⟶⟶⟶⟶// alle
            Block-FS-soll = 0
1026    //⟶⟶Block_FS_max_Register[Block_Nr] = 0;⟶⟶⟶⟶⟶⟶⟶⟶// alle
    Block-FS-soll = 0
1027    //⟶⟶Block_FS_min_Register[Block_Nr] = 0;⟶⟶⟶⟶⟶⟶⟶⟶// alle
    Block-FS-soll = 0
1028    ⟶⟶Block_FS_soll_Register[Block_Nr] = 0;⟶⟶⟶⟶⟶⟶⟶⟶// alle
            Block-FS-soll = 0
1029    //⟶⟶Block_FS_Delay1_Register[Block_Nr] = 0;⟶⟶⟶⟶⟶⟶⟶// alle
    Block-FS-soll = 0
1030    //⟶⟶Block_FS_Delay2_Register[Block_Nr] = 0;⟶⟶⟶⟶⟶⟶⟶// alle
    Block-FS-soll = 0
1031    ⟶⟶Timer_Block_FS_soll_Register[Block_Nr] = 0;⟶⟶⟶⟶⟶⟶// alle
            Timer-Block-FS-soll = 0
1032    ⟶⟶Timer_Block_FS_Delay_Register[Block_Nr] = 0; ⟶⟶⟶⟶⟶⟶// alle
            Timer-Block-FS-Delay = 0
1033    ⟶⟶Timer_Block_FS_ist_Register[Block_Nr] = 0; ⟶⟶⟶⟶⟶⟶// alle
            Timer-Block-FS-ist = 0
1034    ⟶⟶};
1035
1036    ⟶for ( St_Karten_Nr = 0;  St_Karten_Nr < 35;  St_Karten_Nr++)
1037    ⟶⟶Strecken_Status_Register[St_Karten_Nr] = 0;⟶⟶⟶⟶⟶⟶// alle
            Strecken_Register = 0
1038    ⟶set_MC_Bahn_RACK();⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶// Ausgabe
        an Bahn-Hardware
1039    ⟶
1040    ⟶// MC-Rückmeldung --- MC SteuerRegister anzeigen
        ---------------------------------
1041    ⟶usart_write("Bahn-Notstopp !!\r\n");⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶//
        MC-Betriebsmodus = 0
1042    ⟶usart_write("MC-Register --> BM = %1i, NS = %1i, RM = %1i, X, X, AM-HS = %1i,
        AM-SB = %1i, AM-NS = %1i, AM-GB = %1i, AM-St = %1i, X, X \r\n",
1043    ⟶MC_Betrieb_Modus, MC_NotStopp, MC_Responce_Modus, Bahn_Automatik_HS_Modus,
        Bahn_Automatik_SB_Modus, Bahn_Automatik_NS_Modus, Bahn_Automatik_GB_Modus,
        Bahn_Automatik_St_Modus);
1044    ⟶return;  ⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶// Nicht
        sinnloser Eintrag
1045    ⟶}
1046
1047    //#############################################################################
    ###############################
1048    // Bahn-INIT
1049    // Alle MC-Bahn-Werte = 0
1050    //-----------------------------------------------------------------------------
    --------------------------------
1051    void set_MC_Bahn_INIT(void)
1052    ⟶{
1053    ⟶unsigned char St_Karten_Nr, Block_Nr;
1054    ⟶unsigned char CheckSumme;
1055
1056    ⟶MC_Responce_Modus = 0;⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶//
        MC-Rückmeldung aus
1057    ⟶MC_Test_Modus = 0;
1058    ⟶Bahn_Automatik_Modus = 0;
1059
1060    ⟶for (St_Karten_Nr = 1;  St_Karten_Nr < 35;  St_Karten_Nr++)⟶⟶⟶⟶⟶
1061    ⟶⟶{Strecken_Status_Register[St_Karten_Nr] = 0;}⟶⟶⟶⟶⟶⟶// alle
            Strecken_Register = [00]
1062    ⟶set_MC_Bahn_RACK();⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶// Ausgabe
        an Bahn-Hardware
1063
1064    ⟶for (Block_Nr = 1;  Block_Nr <= 32;  Block_Nr++)⟶⟶⟶⟶⟶⟶
1065    ⟶⟶{
1066    ⟶⟶Block_Status_Register[Block_Nr] = 0;⟶⟶⟶⟶⟶⟶⟶⟶// alle
            Block-Status = 0
1067    ⟶⟶Block_Aktiv_Register[Block_Nr] = 0;⟶⟶⟶⟶⟶⟶⟶⟶// alle
```

```
Block-Aktiv = 0
		Block_Delay_Register[Block_Nr] = 0;							// alle Block-Delay = 0
		Block_FS_max_Register[Block_Nr] = 0;							// alle Block-FS-MAX = 0
		Block_FS_min_Register[Block_Nr] = 0;							// alle Block-FS-MIN = 0
		Block_FS_Delay1_Register[Block_Nr] = 0;						// alle Block-FS-Delay1 = 0
		Block_FS_Delay2_Register[Block_Nr] = 0;						// alle Block-FS-Delay2 = 0
		Block_FS_soll_Register[Block_Nr] = 0;							// alle Block-FS-soll = 0
		Timer_Block_FS_soll_Register[Block_Nr] = 0;					// alle Timer-Block-FS-soll = 0
		Timer_Block_FS_Delay_Register[Block_Nr] = 0; 					// alle Timer-Block-FS-Delay = 0
		Timer_Block_FS_ist_Register[Block_Nr] = 0; 					// alle Timer-Block-FS-ist = 0
		}

	// PC-Anzeige ------------------------------------------------------------
	CheckSumme = 0;
	usart_write("MC-INIT !!\r\n");
	usart_write("MC-Port-D = [1__1.1___]\r\n");
	usart_write("MC-Port-A = [0000.0000]\r\n");
	usart_write("MC-Port-C = [0000.0000]\r\n");
	// MC-Rückmeldung --- MC Steuer-Register anzeigen ---------------------------------
	usart_write("MC-Register --> BM = %1i, NS = %1i, RM = %1i, X, X, AM-HS = %1i, AM-SB = %1i, AM-NS = %1i, AM-GB = %1i, AM-St = %1i, X, X \r\n",
	MC_Betrieb_Modus, MC_NotStopp, MC_Responce_Modus, Bahn_Automatik_HS_Modus, Bahn_Automatik_SB_Modus, Bahn_Automatik_NS_Modus, Bahn_Automatik_GB_Modus, Bahn_Automatik_St_Modus);
	}

//##############################################################################################################
// Bahn-RACK
// direkte Ausgabe des MC-Strecken_Status_Register an die Bahn-Hardware
//--------------------------------------------------------------------------------------------------------------
void set_MC_Bahn_RACK(void)
	{
	unsigned char St_Karten_Nr;			// St-Karten-Nr [1..35] --- 35 Eisenbahnkarten im Rack

	// MC und Inteface Output -----
	PORTD |= _BV(7);								// MC-D7 auf 1 => 74LS154 Encoder nicht aktiv
	PORTD |= _BV(3);								// MC-D3 auf 1 => 74LS245 Interface-Daten output
	PORTD &= 255-_BV(4);							// MC-D4 auf 0 => 74LS245 Interface-Daten aktiv
	DDRC = 0xFF;									// MC-Port-C Daten Output
	// Streckenteile aktualisieren -------
	for (St_Karten_Nr = 1; St_Karten_Nr < 35; St_Karten_Nr++)	  	// Karten-Nr [0..34]
		{
		PORTA = St_Karten_Nr;		 				// Port-A => Adresse der Karte ausgeben
		asm("nop");
		PORTC = Strecken_Status_Register[St_Karten_Nr];		// Port-C => Strecken-Register-Nr beginnt mit 1 => KartenAdresse beginnt mit 0
		asm("nop");
		PORTD &= 255-_BV(7);							// MC-Steuer-Port-D7 = 0 => 74LS154 Decoder
		asm("nop");								// !!!! mindestens 4 * NOP
		asm("nop");
		asm("nop");
		asm("nop");
```

```
1115    ⟶⟶asm("nop");
1116    ⟶⟶asm("nop");
1117    ⟶⟶PORTD |= _BV(7);⟶⟶⟶⟶⟶⟶⟶⟶⟶// MC-Steuer-Port-D =>
            Karten-CS 74LS244 = deaktiv
1118    ⟶⟶}
1119    ⟶// Interface Grundstellung -----
1120    ⟶PORTD |= _BV(7);⟶⟶⟶⟶⟶⟶⟶⟶⟶// MC-D7 auf 1 => 74LS154
        Encoder nicht aktiv
1121    ⟶PORTD |= _BV(3);⟶⟶⟶⟶⟶⟶⟶⟶⟶// MC-D3 auf 1 => 74LS245
        Interface-Daten output
1122    ⟶PORTD |= _BV(4);⟶⟶⟶⟶⟶⟶⟶⟶⟶// MC-D4 auf 1 => 74LS245
        Interface-Daten inaktiv
1123    ⟶PORTA = 0x00;⟶⟶⟶⟶⟶⟶⟶⟶⟶// MC-Port-D Adresse = [00]
1124    ⟶PORTC = 0x00;⟶⟶⟶⟶⟶⟶⟶⟶⟶// MC-Port-C Daten = [00]
1125
1126    ⟶// MC-Rückmeldung --- MC Steuer-Register anzeigen
        ---------------------------------
1127    ⟶usart_write("MC-Register --> BM = %1i, NS = %1i, RM = %1i, X, X, AM-HS = %1i,
        AM-SB = %1i, AM-NS = %1i, AM-GB = %1i, AM-St = %1i, X, X \r\n",
1128    ⟶MC_Betrieb_Modus, MC_NotStopp, MC_Responce_Modus, Bahn_Automatik_HS_Modus,
        Bahn_Automatik_SB_Modus, Bahn_Automatik_NS_Modus, Bahn_Automatik_GB_Modus,
        Bahn_Automatik_St_Modus);
1129    ⟶}
1130
1131    //##############################################################################
    ##############################
1132    // Strecken-Status-Register schreiben
1133    // Streckenteile ein- oder ausschalten
1134    //##############################################################################
    ##############################
1135    void set_MC_Bahn_Strecken_Status(void)
1136    ⟶{
1137    ⟶unsigned char ⟶VarStRegNr;⟶⟶// Variable-Nr für St-Register-Nr [0..34] --- 35
        * 8 = 280 Streckenteile in einem Register
1138    ⟶unsigned char ⟶StRegNr;⟶⟶⟶// St-Register-Nr [1..35] --- 35 * 8 = 280
        Streckenteile in einem Register
1139    ⟶unsigned ⟶⟶⟶CheckSumme;
1140
1141    ⟶// Strecken-Stringauf Strecken_Status_Register[1..35(Byte)] verteilen
1142    ⟶for (VarStRegNr = 1; VarStRegNr <= 32; VarStRegNr++)
1143    ⟶⟶{
1144    ⟶⟶Strecken_Status_Register[VarStRegNr] = variable[VarStRegNr];⟶⟶⟶⟶//
            Variablen [0..x]
1145    ⟶⟶};
1146
1147    ⟶// MC-Rückmeldung -- Anzeige Strecken_Status_Register (Walder)
        ------------------------------
1148    ⟶CheckSumme = 0;
1149    ⟶usart_write("St1 ");
1150    ⟶for (StRegNr = 1; StRegNr <= 16; StRegNr++)
1151    ⟶⟶{
1152    ⟶⟶usart_write("%3i ",Strecken_Status_Register[StRegNr]);
1153    ⟶⟶if ((StRegNr % 8) == 0)
1154    ⟶⟶⟶{usart_write(" ");};
1155    ⟶⟶};
1156    ⟶usart_write("\r\n");
1157    ⟶usart_write("St2 ");
1158    ⟶for (StRegNr = 17; StRegNr <= 32; StRegNr++)
1159    ⟶⟶{
1160    ⟶⟶usart_write("%3i ",Strecken_Status_Register[StRegNr]);
1161    ⟶⟶if ((StRegNr % 24) == 0)
1162    ⟶⟶⟶{usart_write(" ");};
1163    ⟶⟶};
1164    ⟶for (StRegNr = 1; StRegNr <= 32; StRegNr++)
1165    ⟶⟶CheckSumme = CheckSumme + Strecken_Status_Register[StRegNr];
1166    ⟶usart_write("  CS = %i \r\n", CheckSumme);
1167
1168    ⟶// direkte Ausgabe des Strecken_Status_Register im BM = 1
        ------------------------------
1169    ⟶if (MC_Betrieb_Modus == 1)
1170    ⟶⟶{
```

```
		set_MC_Bahn_RACK();
		}

	MC_Bahn_Timer_Register_aktualisieren();									// Timer-Register für MAIN aktualisieren
	STR_NEU = 1;														// neue STR-Werte für Timerausgabe
	return;
	}

//@###############################################################################################################
//@ Block-Daten setzen ===> [Block_Nr] [Status] [Aktiv] [Delay] [FS-max] [FS-min] [FS-Delay1] [FS-Delay2]
//@ Block-Nummer, Block-Status, Block-Aktiv, Block-Delay, Block-FS-max, Block-FS-min, Block-FS-Delay1, Block-FS-Delay2
//@###############################################################################################################
void set_MC_Bahn_Block_Daten(void)
	{
	unsigned char BlDaNr;					// Block-Nummer [1..32]
	unsigned      CheckSumme;
	
	BlDaNr 								= variable[1];
	Block_Status_Register[BlDaNr] 			= variable[2];
	Block_Aktiv_Register[BlDaNr] 			= variable[3];
	Block_Delay_Register[BlDaNr] 			= variable[4];
	Block_FS_max_Register[BlDaNr] 			= variable[5];
	Block_FS_min_Register[BlDaNr] 			= variable[6];
	Block_FS_Delay1_Register[BlDaNr] 		= variable[7];
	Block_FS_Delay2_Register[BlDaNr] 		= variable[8];
	
	MC_Bahn_Timer_Register_aktualisieren();										// Timer-Register für MAIN aktualisieren

	// MC-Rückmeldung -- Anzeige Block-Daten -----------------------------
	CheckSumme = 0;
	usart_write ("Block %2i => %2i %2i %2i - F %2i %2i - D %2i %2i", BlDaNr, Block_Status_Register[BlDaNr], Block_Aktiv_Register[BlDaNr], Block_Delay_Register[BlDaNr],
			Block_FS_max_Register[BlDaNr], Block_FS_min_Register[BlDaNr], Block_FS_Delay1_Register[BlDaNr], Block_FS_Delay2_Register[BlDaNr]);
	CheckSumme  = BlDaNr + Block_Status_Register[BlDaNr] + Block_Aktiv_Register[BlDaNr] + Block_Delay_Register[BlDaNr] +
					Block_FS_max_Register[BlDaNr] + Block_FS_min_Register[BlDaNr] + Block_FS_Delay1_Register[BlDaNr] + Block_FS_Delay2_Register[BlDaNr];
	usart_write("   CS=%i \r\n", CheckSumme);
	}

//@##############################################################################################################
// Block-Status-Register ==> Block ist nicht rot !"
// [0..32] Werte in Block-Status-Register schreiben
// 		Bl_Status = 0  oder 1
//@##############################################################################################################
void set_MC_Bahn_Block_Status(void)
	{
	unsigned char BlNrSt;																// Block-Nummer [1..32]
	unsigned      CheckSumme;

	for (BlNrSt = 1; BlNrSt <= 32; BlNrSt++)     				// Block-Nr [1..32] - Regelung ==> Variablen [0..x]
		{
		Block_Status_Register[BlNrSt] = variable[BlNrSt];
		}      					

	MC_Bahn_Timer_Register_aktualisieren();								//
```

```
    Timer-Register für MAIN aktualisieren

	// MC-Rückmeldung -- Anzeige Block_Status_Register ----------------------------
	CheckSumme = 0;
	usart_write("Bl-Status....");
	for (BlNrSt = 1; BlNrSt <= 32; BlNrSt++)
		{
		usart_write("%2i ", Block_Status_Register[BlNrSt]);
		if (((BlNrSt % 8) == 0) && (BlNrSt < 32))
		  {usart_write(" ");};
		CheckSumme  = CheckSumme + Block_Status_Register[BlNrSt];
		};
	usart_write("  CS=%i \r\n", CheckSumme);
	}

//##################################################################################################################
// Block-Aktiv-Register ==> Block stellt neue Geschwindigkeit ein
// [0..32] Werte in Block-Aktiv-Register schreiben
// 		Bl_Aktiv = 0  =>  Timer-Block-FS-soll[i] = Block-FS-soll[i]
// 		Bl_Aktiv = 1  =>  Timer-Block-FS-soll[i] = Block-FS-Vmax[i]
// 		Bl_Aktiv = 2  =>  Timer-Block-FS-soll[i] = Block-FS-min[i]
//##################################################################################################################
void set_MC_Bahn_Block_Aktiv(void)
	{
	unsigned char BlNrA;																// Block-Nummer [1..32]
	unsigned      CheckSumme;

	for (BlNrA = 1; BlNrA <= 32; BlNrA++)  										// Block-Nr [1..32] - Regelung ==> Variablen [0..x]
		{
		Block_Aktiv_Register[BlNrA] = variable[BlNrA];
		if (Block_Aktiv_Register[BlNrA] == 0)
		  {
		  Block_FS_soll_Register[BlNrA] = 0;
		  }
		}

	// MC-Rückmeldung -- Anzeige Block_Aktiv_Register ----------------------------
	CheckSumme = 0;
	usart_write("Bl-Aktiv.....");
	for (BlNrA = 1; BlNrA <= 32; BlNrA++)
		{
		usart_write("%2i ", Block_Aktiv_Register[BlNrA]);
		if (((BlNrA % 8) == 0) && (BlNrA < 32))
		  {usart_write(" ");};
		CheckSumme  = CheckSumme + Block_Aktiv_Register[BlNrA];
		};
	usart_write("  CS=%i \r\n", CheckSumme);

	MC_Bahn_Timer_Register_aktualisieren();											// Timer-Register für MAIN aktualisieren
	}

//##################################################################################################################
// Bahn_Block_FS_max ==> maximale Geschindigkeit
// [0..32] Werte in Block-FS-max-Rregister schreiben
//##################################################################################################################
void set_MC_Bahn_Block_FS_max(void)
	{
	unsigned char BlNrMX;																// Block-Nummer [1..32]
	unsigned      CheckSumme;
```

```
1282
1283    for (BlNrMX = 1; BlNrMX <= 32; BlNrMX++)                    // Block-Nr [1..32] - Regelung ==> Variablen [0..x]
1284    {
1285    Block_FS_max_Register[BlNrMX] = variable[BlNrMX];
1286    if (Block_FS_max_Register[BlNrMX] > Fahrstufen_MAX)
1287     {
1288     Block_FS_max_Register[BlNrMX] = Fahrstufen_MAX;
1289     }
1290    }
1291
1292    MC_Bahn_Timer_Register_aktualisieren();                     // Timer-Register für MAIN aktualisieren
1293
1294    // MC-Rückmeldung -- Anzeige Block_FS_max_Register ----------------------------
1295    CheckSumme = 0;
1296    usart_write("Bl-FS-max....");
1297    for (BlNrMX = 1; BlNrMX <= 32; BlNrMX++)
1298    {
1299    usart_write("%2i ", Block_FS_max_Register[BlNrMX]);
1300    if (((BlNrMX % 8) == 0) && (BlNrMX < 32))
1301     {usart_write(" ");};
1302    CheckSumme  = CheckSumme + Block_FS_max_Register[BlNrMX];
1303    }
1304    usart_write("  CS=%i \r\n", CheckSumme);
1305    }
1306
1307    //#################################################################################################################
1308    // Bahn_Block_FS_min ==> minimale Geschwindigkeit
1309    // [0..32] Werte in Block-FS-min-Rregister schreiben
1310    //#################################################################################################################
1311    void set_MC_Bahn_Block_FS_min(void)
1312    {
1313    unsigned char BlNrMI;
1314    unsigned      CheckSumme;
1315
1316    for (BlNrMI = 1; BlNrMI <= 32; BlNrMI++)                    // Block-Nr [1..32] - Regelung ==> Variablen [0..x]
1317    {
1318    Block_FS_min_Register[BlNrMI] = variable[BlNrMI];
1319    if (Block_FS_min_Register[BlNrMI] > Fahrstufen_MAX)          //Achtung Variablen beginnen mit dem Index 0
1320     {
1321     Block_FS_min_Register[BlNrMI] = Fahrstufen_MAX - 1;
1322     }
1323    }
1324    MC_Bahn_Timer_Register_aktualisieren();                     // Timer-Register für MAIN aktualisieren
1325
1326    // MC-Rückmeldung -- Anzeige Block_FS_min_Register ----------------------------
1327    CheckSumme = 0;
1328    usart_write("Bl-FS-min....");
1329    for (BlNrMI = 1; BlNrMI <= 32; BlNrMI++)
1330    {
1331    usart_write("%2i ", Block_FS_min_Register[BlNrMI]);
1332    if (((BlNrMI % 8) == 0) && (BlNrMI < 32))
1333     {usart_write(" ");};
1334    CheckSumme  = CheckSumme + Block_FS_min_Register[BlNrMI];
1335    };
1336    usart_write("  CS=%i \r\n", CheckSumme);
1337    }
1338
1339    //#################################################################################################################
1340    // Bahn-Block-Delay ==> Startverzögerung
1341    // [0..32] Werte in Block-Delay-Register schreiben
1342    //#################################################################################################################
1343    void set_MC_Bahn_Block_Delay(void)
```

```
	{
	unsigned char BlNrD;											// Block-Nummer [1..32]
	unsigned     CheckSumme;

	for (BlNrD = 1; BlNrD <= 32; BlNrD++)  					// Block-Nr [1..32] - Regelung ==> Variablen [0..x]
		{
		Block_Delay_Register[BlNrD] = variable[BlNrD];
		if (Block_Delay_Register[BlNrD] > 80)				
		  {
		  Block_Delay_Register[BlNrD] = 80;
		  }
		}

	// MC-Rückmeldung -- Anzeige Block_Delay_Register ----------------------------
	CheckSumme = 0;
	usart_write("Bl-Delay.....");
	for (BlNrD = 1; BlNrD <= 32; BlNrD++)
		{
		usart_write("%2i ", Block_Delay_Register[BlNrD]);
		if (((BlNrD % 8) == 0) && (BlNrD < 32))
		  {usart_write(" ");};
		CheckSumme  = CheckSumme + Block_Delay_Register[BlNrD];
		};
	usart_write("  CS=%i \r\n", CheckSumme);
	}

//##########################################################################################################
// Block-FS-Delay1 ==> beschleunigen
// [0..32] Werte in Block-FS-Delay-Register schreiben
//##########################################################################################################
void set_MC_Bahn_Block_FS_Delay1(void)
	{
	unsigned char BlNrD1;											// Block-Nummer [0..34]
	unsigned     CheckSumme;

	for (BlNrD1 = 1; BlNrD1 <= 32; BlNrD1++)  				// Block-Nr [1..32] - Regelung ==> Variablen [0..x]
		{
		Block_FS_Delay1_Register[BlNrD1] = variable[BlNrD1];      		// Achtung Variablen beginnen bei Index 0
		if (Block_FS_Delay1_Register[BlNrD1] > 80)	
		  {
		  Block_FS_Delay1_Register[BlNrD1] = 80;
		  }
		}

	// MC-Rückmeldung -- Anzeige Block_FS_Delay1_Register ---------------------------
	CheckSumme = 0;
	usart_write("Bl-Delay-1...");
	for (BlNrD1 = 1; BlNrD1 <= 32; BlNrD1++)
		{
		usart_write("%2i ", Block_FS_Delay1_Register[BlNrD1]);
		if (((BlNrD1 % 8) == 0) && (BlNrD1 < 32))
		  {usart_write(" ");};
		CheckSumme  = CheckSumme + Block_FS_Delay1_Register[BlNrD1];
		};
	usart_write("  CS=%i \r\n", CheckSumme);
	}

//##########################################################################################################
// Block-FS-Delay2 ==> bremsen
// [0..32] Werte in Block-FS-Delay-Register schreiben
//##########################################################################################################
void set_MC_Bahn_Block_FS_Delay2(void)
```

```
	{
	unsigned char BlNrD2;											// Block-Nummer [0..34]
	unsigned     CheckSumme;

	for (BlNrD2 = 1; BlNrD2 <= 32; BlNrD2++)   					// Block-Nr [1..32] - Regelung ==> Variablen [0..x]
		{
		Block_FS_Delay2_Register[BlNrD2] = variable[BlNrD2];
		if (Block_FS_Delay2_Register[BlNrD2] > 80)
		  {
		  Block_FS_Delay2_Register[BlNrD2] = 80;
		  }
		}

	// MC-Rückmeldung -- Anzeige Block_FS_Delay2_Register ----------------------------
	CheckSumme = 0;
	usart_write("Bl-Delay-2...");
	for (BlNrD2 = 1; BlNrD2 <= 32; BlNrD2++)
		{
		usart_write("%2i ", Block_FS_Delay2_Register[BlNrD2]);
		if (((BlNrD2 % 8) == 0) && (BlNrD2 < 32))
		  {usart_write(" ");};
		CheckSumme  = CheckSumme + Block_FS_Delay2_Register[BlNrD2];
		};
	usart_write("  CS=%i \r\n", CheckSumme);
	}

//###################################################################################################################
// Bahn_Block_FS_soll ==> vorgegebene Geschwindigkeit
// [0..32] Werte in Block-FS-soll-Rregister schreiben
//###################################################################################################################
void set_MC_Bahn_Block_FS_soll(void)
	{
	unsigned char BlNrS;
	unsigned     CheckSumme;

	for (BlNrS = 1; BlNrS <= 32; BlNrS++)   					// Block-Nr [1..32] - Regelung ==> Variablen [0..x]
		{
		Block_FS_soll_Register[BlNrS] = variable[BlNrS];
		if (Block_FS_soll_Register[BlNrS] > Fahrstufen_MAX)
		  {
		  Block_FS_soll_Register[BlNrS] = Fahrstufen_MAX;
		  }
		}

	if (Bahn_Automatik_Modus == 1)										// nur bei Einzelsteuerung übertragen
		MC_Bahn_Timer_Register_aktualisieren();							// Timer-Register für MAIN aktualisieren
	
	// MC-Rückmeldung -- Anzeige Block_FS_soll_Register ----------------------------
	CheckSumme = 0;
	usart_write("Bl-soll......");
	for (BlNrS = 1; BlNrS <= 32; BlNrS++)
		{
		usart_write("%2i ", Block_FS_soll_Register[BlNrS]);
		if (((BlNrS % 8) == 0) && (BlNrS < 32))
		  {usart_write(" ");};
		CheckSumme  = CheckSumme + Block_FS_soll_Register[BlNrS];
		};
	usart_write("  CS=%i \r\n", CheckSumme);
	}

//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@@ BM	-	MC-BM auf allen EB-Karten Belegmessung
```

```
1468  //@@-----------------------------------------------------------------------------
      -------------------------------
1469  /*
1470  void get_Bahn_Block_Test(void)
1471  ---->{
1472  ---->unsigned char
          BMBeNr;---------------------------------------------------------------->
          ---->// Beleg-Register-Nr
1473  ---->unsigned int
          Strecken_Karten_Adresse;----------------------------------------------->
          ---->---->// Adresse der Block- und Streckenkarten
1474
1475  ---->//@@ Belegwerte von Eisenbahn-Hardware auslesen
1476  ---->if (MC_Betrieb_Modus < 3)
1477  ---->---->{
1478  ---->---->//@@ TestLed zum Prüfen ob die Funktion ausgeführt wird !!
1479  ---->---->if (TestLed == 0)
1480  ---->---->---->{
1481  ---->---->---->PORTD |= _BV(6);
1482  ---->---->---->TestLed = 1;
1483  ---->---->---->}
1484  ---->---->else---->
1485  ---->---->---->{
1486  ---->---->---->PORTD &= 255-_BV(6);
1487  ---->---->---->TestLed = 0;
1488  ---->---->---->}
1489  ---->---->//@@ MC und Inteface Daten-Input
1490  ---->---->//@@ MC und Interface Richtung für Daten umschalten zur Belegmessung
1491  ---->---->PORTD |=
               _BV(7);---------------------------------------------------------->
               ---->---->---->// MC-D7 auf 1 => 74LS154 Encoder sperrt
1492  ---->---->DDRC =
               0x00;------------------------------------------------------------->
               ---->---->---->// MC-Port-C (Daten) als Eingang
1493  ---->---->PORTC =
               0xFF;------------------------------------------------------------->
               ---->---->---->// MC-Port-C (Daten) als Eingang - with pull-ups enabled
1494  ---->---->PORTD &=
               255-_BV(3);------------------------------------------------------->
               ---->---->---->// MC-Steuer-Port-D = [____.0___] => Interface-74LS245 Daten IN
1495  ---->---->PORTD &=
               255-_BV(4);------------------------------------------------------->
               ---->---->---->// MC-Steuer-Port-D = [___0.____] => Interface-74LS245 Daten aktiv
1496
1497  ---->---->//@@ Delay für 74LS245 Daten-Umschaltung und MC-Port-C (Daten) als Eingang -
               with pull-ups enabled
1498  ---->---->asm("nop");
1499  ---->---->asm("nop");
1500  ---->---->asm("nop");
1501  ---->---->asm("nop");
1502  ---->---->asm("nop");
1503  ---->---->asm("nop");
1504
1505  ---->---->//@@ Belegmeldung von 35 Karten auslesen
1506  ---->---->for (Strecken_Karten_Adresse = 35; Strecken_Karten_Adresse < 39;
               Strecken_Karten_Adresse++)---->// Karte-CS - Adresse-Port-A = [35..39]
1507  ---->---->---->{
1508  ---->---->---->PORTA =
                    Karte_CS;--------------------------------------------------->
                    ---->---->---->// MC-Port-A => Adresse [35..69] = Karte-CS ausgeben
1509  ---->---->---->PORTD &=
                    255-_BV(7);------------------------------------------------->
                    ---->---->---->// MC-Steuer-Port-D = [0___.____] => Karten-CS 74LS244 = 0 =
                    aktiv
1510  ---->---->---->//@@ Delay für 74LS245 Daten-Umschaltung und MC-Port-C (Daten) als
                    Eingang - with pull-ups enabled
1511  ---->---->---->asm("nop");
1512  ---->---->---->asm("nop");
1513  ---->---->---->asm("nop");
1514  ---->---->---->asm("nop");
1515  ---->---->---->asm("nop");
```

```
			asm("nop");
			//@@ Delay ENDE
			Beleg_Register[Strecken_Karten_Adresse - 35] = PINC;			// Einlesen von Karten-Adresse [35..39] * 8 Bit => Beleg-Array[0..34] speichern
			PORTD |= _BV(7);			// MC-Steuer-Port-D = [1___.____] => Karten 74LS244 = 1 sperrt
			}
		DDRC &= 0xFF;			// MC-Daten-Port-C als Ausgang
		}

	//@@ Belegwerte an PC senden
	usart_write("MC-BM ");
	for (BMBeNr = 1; BMBeNr <= 34; BMBeNr++)			// Register 0..34
		{usart_write("%3i ", Beleg_Register[BMBeNr]);};			// MC sendet an PC die Beleg-Register-Daten
	usart_write("\r\n");
	//@@ MC_Responce_Modus = 0;			// MC-Rückmeldung aus
	}
*/

//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@@ BM	-	MC-BM auf allen Streckenteile Belegmessung
//@@ Karten_CS 01..35 Daten schreiben in 74LS377
//@@ Karten_CS 36..69 Beleg-Daten lesen aus 74LS244
//@@ Steuerung-Gundzustand Port-D = 1xx1.1xxx
//@@ * D7 CS für D-FlipFlop	und 74ls244 => D7 = 1 aktiv
//@@ * D4 CS für Interface-Datentreiber 74LS245 => D4 = 0 aktiv
//@@ * D3 Richtung für Interface-Datentreiber 74LS245 D3 = 1 schreiben, D3 = 0 Lesen
//@@ * Bit auf 0 => &= 255-_BV(x);
//@@ * Bit auf 1 => |= _BV(x);
//@@---------------------------------------------------------------------------------------------------------------
void get_MC_Bahn_Strecken_Beleg(void)
	{
	unsigned char BMBeNr;			// Beleg-Register-Nr
	unsigned int Strecken_Karten_Adresse;			// Adresse der Block- und Streckenkarten

	//@@ Belegwerte von Eisenbahn-Hardware auslesen
	if (MC_Betrieb_Modus < 3)
		{
		//@@ TestLed zum Prüfen ob die Funktion ausgeführt wird !!
		if (TestLed == 0)
			{
			PORTD |= _BV(6);
			TestLed = 1;
			}
		else
			{
			PORTD &= 255-_BV(6);
			TestLed = 0;
			}
	
		//@@ MC und Inteface Output
		PORTD |= _BV(7);			// MC-D7 auf 1 => 74LS154 Encoder sperrt
		PORTD |=
```

```
_BV(3);                                        // MC-D3 auf 1 => 74LS245 Interface-Daten output
        PORTD &= 255-_BV(4);                   // MC-D4 auf 0 => 74LS245 Interface-Daten aktiv
        DDRC = 0xFF;                           // MC-Daten Port-C als Ausgang

        //@@ alle 32 Blockabschnitte und alle 248 Streckenabschnitte auf 0
        //@@ 74LS154 erzeugt CS mit fallender Flanke und 74LS377 übernimmt Daten
        for (Strecken_Karten_Adresse = 0; Strecken_Karten_Adresse < 35; Strecken_Karten_Adresse++)    // 35 Karte-CS - Adressen-Port-A = [0..34]
            {
            PORTA = Strecken_Karten_Adresse;   // 74LS154 CS für Kartenadresse [0..34] ausgeben
            PORTC = 0x00;                      // MC-Daten-Port-C = [0000.0000] alle Streckenteile auf 0
            PORTD &= 255-_BV(7);               // MC-Steuer-Port-D = [0___.____] => Karten-CS 74LS377 = 0
            asm("nop");                        // !!!! mindestens 4 * NOP
            asm("nop");
            asm("nop");
            asm("nop");
            asm("nop");
            PORTD |= _BV(7);                   // MC-Steuer-Port-D = [1___.____] => CS für 74LS377 wieder auf 1
                                               // Wert im 74LS377 speichern
            }

        //@@ MC und Inteface Daten-Input
        //@@ MC und Interface Richtung für Daten umschalten zur Belegmessung
        PORTD |= _BV(7);                       // MC-D7 auf 1 => 74LS154 Encoder sperrt
        DDRC = 0x00;                           // MC-Port-C (Daten) als Eingang
        PORTC = 0xFF;                          // MC-Port-C (Daten) als Eingang - with pull-ups enabled
        PORTD &= 255-_BV(3);                   // MC-Steuer-Port-D = [____.0___] => Interface-74LS245 Daten IN
        PORTD &= 255-_BV(4);                   // MC-Steuer-Port-D = [___0.____] => Interface-74LS245 Daten aktiv

        //@@ Delay für 74LS245 Daten-Umschaltung und MC-Port-C (Daten) als Eingang - with pull-ups enabled
        asm("nop");
        asm("nop");
        asm("nop");
        asm("nop");
        asm("nop");
        asm("nop");

        //@@ Belegmeldung von 35 Karten auslesen
        for (Strecken_Karten_Adresse = 35; Strecken_Karten_Adresse < 69; Strecken_Karten_Adresse++)    // Karte-CS - Adresse-Port-A = [35..69]
            {
            PORTA = Karte_CS;
```

```
			// MC-Port-A => Adresse [35..69] = Karte-CS ausgeben
			PORTD &= 255-_BV(7);			// MC-Steuer-Port-D = [0___.____] => Karten-CS 74LS244 = 0 = aktiv
			//@@ Delay für 74LS245 Daten-Umschaltung und MC-Port-C (Daten) als Eingang - with pull-ups enabled
			asm("nop");
			asm("nop");
			asm("nop");
			asm("nop");
			asm("nop");
			asm("nop");
			//@@ Delay ENDE
			Beleg_Register[Strecken_Karten_Adresse - 35] = PINC;			// Einlesen von Karten-Adresse [35..69] * 8 Bit => Beleg-Array[0..34] speichern
			PORTD |= _BV(7);			// MC-Steuer-Port-D = [1___.____] => Karten 74LS244 = 1 sperrt
			}
		}
	
	//@@ Belegmeldung von 35 Karten auslesen
	DDRC &= 0xFF;			// MC-Daten-Port-C als Ausgang
	//	MC_Responce_Modus = 10;

	//@@ Belegwerte an PC senden
	usart_write("MC-BM ");
	for (BMBeNr = 1; BMBeNr <= 35; BMBeNr++)			// Register 0..34
		{usart_write("%3i ",Beleg_Register[BMBeNr]);};			// MC sendet an PC die Beleg-Register-Daten
	usart_write("\r\n");
	//@@ MC_Responce_Modus = 0;			// MC-Rückmeldung aus
	}

//##############################################################################################################
//@@ KM	-	MC-KM auf allen EB-Karten Kurzschlussmessung
//--------------------------------------------------------------------------------------------------------------
void get_MC_Bahn_Strecken_Kurzschluss(void)
	{
	unsigned char KMBeNr;			// Kurzschluss-Register-Nr
	unsigned int Strecken_Karten_Adresse;			// Adresse der Block- und Streckenkarten

	//@@ Belegwerte von Eisenbahn-Hardware auslesen
	if (MC_Betrieb_Modus < 6)
		{
		//@@ TestLed zum Prüfen ob die Funktion ausgeführt wird !!
		if (TestLed == 0)
			{
			PORTD |= _BV(6);
			TestLed = 1;
			}
		else	
			{
			PORTD &= 255-_BV(6);
			TestLed = 0;
			}
```

```
		//@@ MC und Inteface Daten-Input
		//@@ MC und Interface Richtung für Daten umschalten zur Belegmessung
		PORTD |= _BV(7);										// MC-D7 auf 1 => 74LS154 Encoder sperrt
		DDRC = 0x00;											// MC-Port-C (Daten) als Eingang
		PORTC = 0xFF; 										// MC-Port-C (Daten) als Eingang - with pull-ups enabled
		PORTD &= 255-_BV(3);									// MC-Steuer-Port-D = [____.0___] => Interface-74LS245 Daten IN
		PORTD &= 255-_BV(4);									// MC-Steuer-Port-D = [___0.____] => Interface-74LS245 Daten aktiv

		//@@ Delay für 74LS245 Daten-Umschaltung und MC-Port-C (Daten) als Eingang - with pull-ups enabled
		asm("nop");
		asm("nop");
		asm("nop");
		asm("nop");
		asm("nop");
		asm("nop");

		//@@ Belegmeldung von 35 Karten auslesen
		for (Strecken_Karten_Adresse = 35; Strecken_Karten_Adresse < 39; Strecken_Karten_Adresse++)		// Karte-CS - Adresse-Port-A = [35..39]
			{
			PORTA = Karte_CS;									// MC-Port-A => Adresse [35..69] = Karte-CS ausgeben
			PORTD &= 255-_BV(7);								// MC-Steuer-Port-D = [0___.____] => Karten-CS 74LS244 = 0 = aktiv
			//@@ Delay für 74LS245 Daten-Umschaltung und MC-Port-C (Daten) als Eingang - with pull-ups enabled
			asm("nop");
			asm("nop");
			asm("nop");
			asm("nop");
			asm("nop");
			asm("nop");
			//@@ Delay ENDE
			Kurzschluss_Register[Strecken_Karten_Adresse - 35] = PINC;				// Einlesen von Karten-Adresse [35..39] * 8 Bit => Beleg-Array[0..34] speichern
			PORTD |= _BV(7);									// MC-Steuer-Port-D = [1___.____] => Karten 74LS244 = 1 sperrt
			}
		DDRC &= 0xFF;											// MC-Daten-Port-C als Ausgang
		}

	//@@ Kurzschlusswerte an PC senden
	usart_write("MC-KM ");
	for (KMBeNr = 1; KMBeNr <=  34; KMBeNr++) 							// Register 0..34
		{usart_write("%3i ",Kurzschluss_Register[KMBeNr]);};					// MC sendet an PC die Kurzschluss-Register-Daten
	usart_write("\r\n");
	//@@ MC_Responce_Modus = 0;									// MC-Rückmeldung aus
```

```
1695    ⟶}
1696
1697
1698
1699    //#############################################################################################
        ################################
1700    // Bahn-Timer-Register⟶-⟶für MAIN laden
1701    //-----------------------------------------------------------------------------------------
        -------------------------------
1702    void MC_Bahn_Timer_Register_aktualisieren(void)
1703    ⟶{
1704    ⟶unsigned char TBlNr;
1705    ⟶
1706    //⟶usart_write("=> Timer FS\r\n");
1707
1708
1709    //##############################
1710    // PROBLEM mit HS, NS, SB ...
1711    ⟶Bahn_Automatik_Modus = 1;
1712
1713
1714    // Einelsteuerung ------------------------------------------------------------
1715    ⟶if (Bahn_Automatik_Modus == 0)
1716    ⟶⟶{
1717    ⟶⟶⟶for (TBlNr = 1; TBlNr <= 32; TBlNr++)  ⟶⟶⟶⟶⟶⟶⟶// Block-Nr
            [1..32] - Regelung
1718    ⟶⟶⟶⟶{
1719    ⟶⟶⟶⟶if (Block_Status_Register[TBlNr] == 1)
1720    ⟶⟶⟶⟶⟶{
1721    ⟶⟶⟶⟶⟶Timer_Block_FS_soll_Register[TBlNr] = Block_FS_soll_Register[TBlNr];
1722    ⟶⟶⟶⟶⟶Timer_Block_FS_Delay_Register[TBlNr] =
                Block_FS_Delay1_Register[TBlNr];⟶// mit FS-Delay1
1723    ⟶⟶⟶⟶⟶}
1724    ⟶⟶⟶⟶else
1725    ⟶⟶⟶⟶⟶{
1726    ⟶⟶⟶⟶⟶Timer_Block_FS_soll_Register[TBlNr] = 0;
1727    ⟶⟶⟶⟶⟶Timer_Block_FS_ist_Register[TBlNr] = 0;
1728    ⟶⟶⟶⟶⟶Timer_Block_FS_Delay_Register[TBlNr] = 0;
1729    ⟶⟶⟶⟶⟶}
1730    ⟶⟶⟶⟶}⟶
1731    ⟶⟶⟶}⟶
1732    ⟶⟶
1733    ⟶⟶// Automatiksteuerung
        ------------------------------------------------------------------------------
        --
1734    ⟶⟶if (Bahn_Automatik_Modus != 0)
1735    ⟶⟶⟶{
1736    ⟶⟶⟶for (TBlNr = 1; TBlNr <= 32; TBlNr++)  ⟶⟶⟶⟶⟶⟶⟶// Block-Nr
            [1..32] - Regelung
1737    ⟶⟶⟶⟶{
1738    ⟶⟶⟶⟶if (Block_Status_Register[TBlNr] == 1)⟶⟶⟶⟶⟶⟶⟶// wenn
              Block "GRÜN"
1739    ⟶⟶⟶⟶⟶{
1740    ⟶⟶⟶⟶⟶switch (Block_Aktiv_Register[TBlNr])
1741    ⟶⟶⟶⟶⟶⟶{
1742    ⟶⟶⟶⟶⟶⟶// ==> Block-FS = 0
                  ------------------------------------------------------
1743    ⟶⟶⟶⟶⟶⟶case 0 : ⟶Timer_Block_FS_soll_Register[TBlNr] = 0;
1744    ⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶Timer_Block_FS_Delay_Register[TBlNr] =
                  0;⟶⟶⟶⟶⟶⟶⟶⟶⟶// mit FS-Delay = 0
1745    ⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶break;
1746    ⟶⟶⟶⟶⟶⟶// ==> Beschleunigen ----------------------------------------
1747    ⟶⟶⟶⟶⟶⟶case 1 :⟶Timer_Block_FS_soll_Register[TBlNr] =
                  Block_FS_max_Register[TBlNr];⟶⟶⟶// mit FS-MAX
1748    ⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶Timer_Block_FS_Delay_Register[TBlNr] =
                  Block_FS_Delay1_Register[TBlNr];⟶// mit FS-Delay1
1749    ⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶break;
1750    ⟶⟶⟶⟶⟶⟶// ==> Bremsen ----------------------------------------------
1751    ⟶⟶⟶⟶⟶⟶case 2 : ⟶Timer_Block_FS_soll_Register[TBlNr] =
                  Block_FS_min_Register[TBlNr];⟶⟶⟶// mit FS-MIN
1752    ⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶⟶Timer_Block_FS_Delay_Register[TBlNr] =
```

```
Block_FS_Delay2_Register[TBlNr];	// mit FS-Delay2
								break;
						// ==> FS-MAX ------------------------------------------------
						case 5 :	Timer_Block_FS_soll_Register[TBlNr] = Block_FS_max_Register[TBlNr];		// mit FS-MAX
								Timer_Block_FS_ist_Register[TBlNr] = Block_FS_max_Register[TBlNr];			// mit FS-MAX
								Timer_Block_FS_Delay_Register[TBlNr] = 0;									// mit Timer_Block_FS_Delay = 0
								break;
						}
					}
				}	
			}	
		}


//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//
// Weichenbetrieb (BM = 3 )
//
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

//==============================================================================================================
// WeichenDaten ==> Reset
//==============================================================================================================
void set_MC_Bahn_Weichen_Reset(void)
	{
	// Weichen Daten INIT = alle Weichen Daten =[00]
	for (Weichen_Nr = 1; Weichen_Nr < 9; Weichen_Nr++)
		{
		WeichenDaten_links[Weichen_Nr]  = 0;
		WeichenDaten_rechts[Weichen_Nr] = 0;
		}
	usart_write("Weichen_Daten = [00] !\r\n");

	// Weichen-Daten = [00] an alle Weichenschalter ausgeben ----------------------------
	DDRC = 0xFF;																		// MC-Port-C Daten als Ausgang
	DDRD = 0xFF;																		// MC-Port-D Control als Ausgang
	PORTC = 0x00;																		// MC-Port-C = [00] alle WeichenDaten = 0
	// CS D-FlipFlop für alle WeichenSchalter
	PORTD  &= 255 - _BV(6);															// MC-Port-D6 = [0] Weichen link = CS 1 - 0 - 1
	PORTD |= _BV(6);																	// MC-Port-D6 = [1]
	PORTD  &= 255 - _BV(7);															// MC-Port-D6 = [0] Weichen rechts = CS 1 - 0 - 1
	PORTD |= _BV(7);																	// MC-Port-D6 = [1]

	usart_write("Weichen[1..64] Daten = [00] !\r\n");
	}

```

```
1800  //=============================================================================
      ============================
1801  // Weichen schalten
1802  // WeichenNummern vom PC = 1..64 werden zur Ausgabe auf MC = 0..63 umgesetzt
1803  //=============================================================================
      ============================
1804  void set_MC_Bahn_Weichen(void)
1805  {
1806  // WeichenDaten einlesen (PC = 1..64)
      ------------------------------------------------------------
1807  Weichen_Nr = variable[1];
1808  WeichenDaten_NEU[Weichen_Nr] = variable[2];
1809  // WeichenNummern vom PC = 1..64 werden zur Ausgabe auf MC = 0..63 umgesetzt
1810  PORTC |= _BV(Weichen_Nr - 1);                                              
      // MC-Port-Cx = [1]
1811
1812  if (WeichenDaten_NEU[Weichen_Nr] == 0)
1813  {
1814  // wenn WeichenDaten_NEU == [0] dann Weichen links
1815  // CS D-FlipFlop für WeichenSchalter Links => Port D6 = [0] => [1]
        ------------------------
1816  PORTD &= 255 - _BV(6);                                                     
        // MC-Port-D6 = [0] Weichen link = CS 1 - 0 - 1
1817  PORTD |=
        _BV(6);                                                                
        // MC-Port-D6 = [1]
1818  usart_write("Weiche %2i => links gestellt !\r\n", Weichen_Nr);
1819  }
1820  else
1821  {
1822  // wenn WeichenDaten_NEU == [1] dann Weichen rechts
1823  // CS D-FlipFlop für WeichenSchalter Rechts => Port D7 = [0] => [1]
        ------------------------
1824  PORTD &= 255 -
        _BV(7);                                                                //
        MC-Port-D7 = [0] Weichen rechts = CS 1 - 0 - 1
1825  PORTD |=
        _BV(7);                                                                
        // MC-Port-D7 = [1]
1826  usart_write("Weiche %2i => rechts gestellt !\r\n", Weichen_Nr);
1827  }
1828
1829  // Weichen Rückstellung Delay -------------------------------------
1830  for (Weichen_TimerDelay = Weichen_Delay; Weichen_TimerDelay > 0;
      Weichen_TimerDelay--)
1831  {
1832  long_delay(10);
1833  }
1834
1835  // 3. Weichen_Rückstellung ----------------------------
1836  // Alle Weichen-Infos = [00]
1837  PORTC =
      0x00;                                                                    //
      MC-Port-C = [00]
1838  // CS D-FlipFlop für alle WeichenSchalter
1839  PORTD &= 255 - _BV(6);                                                   //
      MC-Port-D6 = [0] Weichen link = CS 1 - 0 - 1
1840  PORTD |=
      _BV(6);                                                                  //
      MC-Port-D6 = [1]
1841  PORTD &= 255 - _BV(7);                                                   //
      MC-Port-D6 = [0] Weichen rechts = CS 1 - 0 - 1
1842  PORTD |=
      _BV(7);                                                                  //
      MC-Port-D6 = [1]
1843  //usart_write("Weichen_Rückstellung !\r\n");
1844  }
1845
1846  //=============================================================================
      ============================
1847  // Weichenantrieb Delay
```

```
1848  //=============================================================================================
      ============================
1849  void set_MC_Bahn_Weichen_Delay(void)
1850  ⟶{
1851  ⟶Weichen_Delay = variable[1];
1852  ⟶if (variable[1] < 255)
1853  ⟶⟶{Weichen_Delay = variable[1];}
1854  ⟶else
1855  ⟶⟶{Weichen_Delay = 255;}
1856
1857  ⟶Weichen_MC_TimerDelay = Weichen_Delay + 1000;
1858  ⟶usart_write("Weichen_Delay = %8i\r\n", Weichen_Delay);
1859  ⟶}
1860  ⟶
1861
1862
1863
1864  //  ENDE
      -----------------------------------------------------------------------------------------------
      ----------------------------
```