'kompiliert mit BASCOM V2.0.7.7.

'---------------------------------

'Ausstehende Korrekturen/Verbesserungen:

'---------------------------------------

'- beachte, daß angesteckter Programmieradapter Fehlfunktionen auslöst, da scheinbar Tasten gedrückt sind

'- Vergesse nicht abschließend $EEPROMHEX einzukommentieren

'- PWM-Werte über 1023 beschneiden

'- scroll PWM-Output mit Encoder beim Einschalten oder im Betrieb

'- PWM-Out auch in FM_Tune_Status_Freq_LCD (CALL-SUB)

'- kurze Frequenz-Anzeige in Radiotext auch bei stehendem Radiotext oder Ping-Pong-Scrolling

'- Prüfe AM-Sendernamen-Eingabe/Speicherung/Löschung/Einlesen

'- tausche kurzen und langen S5-Druck (Memory STORE/RECALL aus)

'- Prüfe Property-Verstellung

'- Prüfe RS232-Steuerung

'- prüfe, ob PTY-Name, Ländercode, Regionalcode, Language-Code (evtl. nicht im Kabel) richtig dekodiert werden

'- evtl. Band-Rollover doch wieder einführen für Scan???

'- FM-Bandstart alternativ auf 64MHz setzen. Mal gucken, was da los ist :-)

'- Amateurfunkbänder einbauen?

'- I2C-Befehle verkürzen mit I2CRECEIVE, I2CSEND

'Ausstehende Verbesserungen RDS:

'------------------------------

'- Linefeed LF in Radiotext auswerten (bisher nur NOPs)

'- RT-Empfang kontinuierlich auch nach Erst-Dekodierung einer Nachricht, solange wie kein A/B-Flagwechsel?

'- PIN_code im Klartext auswerten?

'- TMC-Dekodierung implementieren (Gruppe 8)?

'Versionshistorie:

'----------

'V1.0.0.:

'--------

'- FM-Deemphasis auf "1"(=50µs für Europa) anstelle Default-Wert "2" (=75µs für USA etc.) !!!

'- Frequenzsuchraster FM : 50Khz (anstelle Default-Wert 100 khz)

'- Max. Tune-Error FM 20kHz (empfohlen lt. Datenblatt) anstelle Default-Wert 30 kHz

'- Letzte eingestellte Frequenz/Band/AM-FM-Modus beim Einschalten aus EEPROM einlesen und bei Veränderungen jeweils in EEPROM speichern

'- 50 FM-Speicher, 50 AM-Speicher + Namen

'- S2/S4 länger als 2s gedrückt --> FM/AM-Bandscan und automatische Abspeicherung der gefundenen Sender In Speicher 1...n (max. 30); Speicher 1 dann aufrufen

'- Beim Speichern eines AM-Senders wird Name (10 Zeichen) mit Drehencoder+Memory-Taste eingegeben und nach 10 Zeichen automatisch mitgespeichert

'- Falls Property-EEPROM-Speicher=$FF --> lese Default-Wert aus Si4735 aus und speichere in Property(n)

'- Einstellung der je acht Properties durch Drücken von S5 (Memory) länger als 2s

' - Auswahl der Property mit Drehencode

' - Bestätigen mit kurzem Tastendruck von S5 (Memory)

' - Variation der Property mit Drehencoder innerhalb der erlaubten Grenzen

' - Bestätigen und Abspeichern der Property mit kurzem Tastendruck von S5 (Memory)

'- FM-Anzeige:

' - Ergänzung "S" = Stereo , "R" = RDS-Daten vorhanden (LCD-Zeile 2, Spalte 15/16)

'- Ausgabe (sehr vieler) RDS-Informationen (meist) im Klartext über RS-232

'V1.0.1.:

'--------

'Anzeige der Betriebsspannung bei Start

'Anzeige der HW/FW/Rev. nach Powerup

'Anzahl KW-Bänder erweitert

'KW-Band bei Umschaltung im Klartext ausgeben

'Kommentare auf LCD bei allen Aktionen

'V1.0.2.:

'--------

'FM-Senderspeicher auf 50 erweitert

'AM-Senderspeicher auf 50 erweitert

'EEPROM enthält einige ausgewählte gültige LW,MW-Frequenzen und Namen

'EEPROM enthält gültige Default-Werte für Properties

'V1.0.3.:

'--------

'RDS: AF-Speicherung korrigiert

'RDS: Länder und Regionalcodes explizit

'RDS: Gruppe 1A und 1B-Dekodierung implementiert

'RDS: Language-Code im Klartext

'RDS: Datumsberechnung aus MJD implementiert

'RDS: Gruppe 10A-Dekodierung implementiert (PTYN)

'RDS: Gruppe 9A-Hex-Ausgabe implementiert (EWS)

'V1.0.7.:

'--------

'RDS: Inverses "R" für RDS+Radiotext-Empfang

'V1.0.8.:

'--------

'Codetabelle für Sonderzeichen, insbes. Umlaute

'RS232 Befehl "j" für einzelne RDS-Info-Teilbereiche jetzt mit Subvariable --> jedes gesetzte Bit (0...6) = 1 Teilberich der gesamten RDS-Info

'Ping-Pong Scrollen bei Radiotext-Länge <25, sonst Scrollen von links n. rechts und Repeat

'bem Scrollen: Start und Ende länger stehen lassen (3x Schleifendurchlauf)

'geänderte PWM-Ausgabe (anstelle RSSI): Stereo-Signal (R3 Bit 0..6[%])

'PWM-Ausgabe Frequenz-Offset "Ratio-Mitte" (FM_RSQ_Status, R7 Bit0:7 [kHz]), wählbar durch Tastendruck S2 beim Einschalten

'Sonderzeichen "ST" für STEREO-Signal definiert, Ausgabe auf Zeile 2, Spalte 15

'Sonderzeichen "RT" für RADIOTEXT-Signal definiert, Ausgabe auf Zeile 2, Spalte 16; ersetzt inverses R

'V1.0.9.:

'--------

'RAM-Variablen optimiert

'EON-Dekodierung eingebaut

'V1.1.0.:

'--------

'IRS optimiert

'zugelassene Blockerrors testweise auf 2 = 3...5 korrigierbare Errors gesetzt

'über RS232 wird nur ein Datensatz ausgegeben

'falls bei Bandscan kein Sender gefunden wird --> kein Memory_Scroll-Modus

'alternative PWM-Ausgabe Frequenz-Offset "Ratio-Mitte" (FM_RSQ_Status, R7 Bit0:7 [kHz]), wählbar durch Tastendruck S2 beim Einschalten

'alternative Frequenz-Schrittweite manuell: 10kHz (FM), 1kHz (AM), wählbar durch Tastendruck S3 beim Einschalten

'V1.1.1.:

'--------

'S2 beim Einschalten gedrückt: Toggelt alternative PWM-Ausgabe Frequenz-Offset "Ratio-Mitte"

'S3 beim Einschalten gedrückt: Toggelt alternative Frequenz-Schrittweite manuell: 10kHz (FM), 1kHz (AM) / 50 kHz (FM) / 9/9/5 kHz (AM)

'EEPROM speichern und lesen der obigen Flags

'zugelassene Blockerrors wieder auf 1 = 1...2 korrigierbare Errors zurückgesetzt

'Betriebsspannungsmessung und Firmware-Anzeige aus Platzgründen auskommentiert

'EEPROM-Memory-Map:

'------------------

' 1 = Anzahl belegter FM-Senderspeicher($FF = unbelegt) "Fm_memorymax"

' 2...101 = 50 Gespeicherte AM-Frequenzen "Fm_memory(n)"

'102...201 = 50 Gespeicherte FM-Frequenzen "Am_memory(n)"

'202...751 = 50 Sendernamenspeicher f. AM-Sender; je 10+1 Bytes "Am_text(n)"

'995...999   = Anzahl belegter Senderspeicher in LW, MW, KW1, KW2, KW3

'1000   = Letzter eingestellter Modus (AM/FM)

'1001...1002 = Letzte Eingestellte FM-Frequenz

'1003...1004 = Letzte eingestellte AM-Frequenz

'1005   = Letztes eingestelltes AM-Band

'1006...1021 = 16 Bytes Tuning-Properties "Property_dat(n)"

'1022   = Flag für FM-PWM-Ausgabe

'1023   = Flag für manuelle Schrittweite

```
$regfile = "m328pdef.dat"                    'ATMega328P-PU: 32k Flash 2k RAM 1k EEPROM

$crystal = 8000000                           '8 MHz

$hwstack = 100                               'je 2 Byte für Rücksprungadressen CALL/GOSUB/FUNCTION; je 34 Byte pro (multiple) IRS

$swstack = 100                               'je 2 Byte für (lokale Variable + Variablen, die an ein Unterprogramm übergeben werden)

$framesize = 100                             ' (mit Byval übergebene + lokale Variable) + 24 Bytes für Umwandlungsroutinen (PRINT/LCD, INPUT, FORMAT(FP-Variable), STR$


$map                                         'Report-File um die Label-Adressen erweitern


'$sim                                        'Befehle für Simulation, damit Wait-Zyklen nicht abgearbeitet werden

'$dbg                                        'zur Stackanalyse


'$prog &HFF , &HD7 , &HD7 , &HFD                                'Fuse bytes.


$baud = 38400                                '38.400 Baud über Hardware Rs-232 an FTDI


$lib "I2C_TWI.LBX"                           'nur bei Benutzung der Hardware I2C-Ports




'PIN-Configuration:

'------------------


'INPUT:

'------
```

```
'PB2 = Pin 16 Taster S2

'PB3 = Pin 17 Taster S3

'PB4 = Pin 18 Taster S4

'PB5 = Pin 19 Taster S5

'PC1 = Pin 24 Encoder

'PC2 = Pin 25 Encoder

'PC0 = Pin 23 ADC für Lautstärke-Poti

'PD0 = Pin 2  RxD



'*** Taster ***

Config Pinb.2 = Input : S2 Alias Pinb.2 : Set Portb.2      'PB2=S2 Pullup einschalten

Config Pinb.3 = Input : S3 Alias Pinb.3 : Set Portb.3      'PB2=S2 Pullup einschalten

Config Pinb.4 = Input : S4 Alias Pinb.4 : Set Portb.4      'PB2=S2 Pullup einschalten

Config Pinb.5 = Input : S5 Alias Pinb.5 : Set Portb.5      'PB2=S2 Pullup einschalten


'*** Encoder ***

Config Pinc.1 = Input : Set Portc.1                  'Encoder-Portpins Pullups einschalten

Config Pinc.2 = Input : Set Portc.2


Config Pinc.0 = Input                          'ADC0 für Lautstärke-Poti


Config Pind.0 = Input                          'RxD für Hardware RS-232



'OUTPUT:

'-------

'PB0 = Pin 14 Verkürzungsspule

'PB1 = Pin 15 Verkürzungsspule
```

```
'PC3 = Pin 26 Verkürzungsspule
'PC4 = Pin 27 I2C-SDA: Si4735 Data
'PC5 = Pin 28 I2C_SCL: Si4735 Clock
'PD1 = Pin 3  TxD

'PD2...PD7 = EADOGM162:RS,E,DB4...DB7

Config Pinb.0 = Output                  'Spule MF
Config Pinc.3 = Output                  'Spule LF
Config Pinb.1 = Output                  'PWM1A-Ausgang für S-Meter

Config Pind.1 = Output                  'TxD für Hardware RS-232



'******************
'*** Konstanten ***
'******************



'*****************
'*** Variablen ***
'*****************


Dim N As Byte , N1 As Byte , N2 As Byte          'HilfsVariable Schleifenzähler
```

```
Dim I As Word , I1 As Word , I2 As Word                'IndexVariable

Dim T As Word                                          'Schleifendurchlaufzähler Main-Loop

Dim Lcdrefresh As Byte                                 'Schleifenzähler; nach 5 Durchläufen x 200 Mainloop-
Durchläufen Frequenz und Tuingdetails erneuern


Dim D As Byte                                          'Hilfs-Variable "Daten"

Dim Lo As Byte , Hi As Byte                            'Lo und High-Byte f. diverse Zwecke


Dim Text1 As String * 16                               'allgemeiner Textausgabe-String

Dim Byte1 As Byte , Byte2 As Byte , Word1 As Word , Word2 As Word , Long1 As Long , Float1 As Single



'*** Poti, Taster, Encoder ***


Dim Pot As Word                                        'Poti-Wert aus ADC

Dim Dif As Integer                                     'Differenz zum letzten Poti-Wert

Dim T1 As Word , T2 As Word , T3 As Word , T4 As Word      'Verzögerungszähler für lang gedrückte Tasten

Dim Encoder1 As Integer , Encoder1old As Integer           'gelesene Encoder-Bits

Dim Code1 As Byte , Code1old As Byte                   'Encoder-Codes


'*** Frequenzen, Bänder, Schrittweiten ***


Dim Am_fm_flag As Byte                                 'Flag für AM-/FM-Modus: 1=AM, 0=FM

Dim Freq_fm As Word                                    'FM-Frequenz

Dim Freq_am As Word                                    'AM-Frequenz

Dim Freq As Word                                       'Frequenz

Dim Band As Byte                                       'AM-Frequenzband

Dim Band_start(17) As Word , Band_end(17) As Word          'Band_start/-end-Frequenzen

Dim Band_name As String * 16                           'Bandname
```

```
Dim Am_lower_limit As Word : Am_lower_limit = 149          'absolute AM-Grenzfrequenzen

Dim Am_upper_limit As Word : Am_upper_limit = 23000

Dim Seek_step As Byte                        'Suchschrittweite in kHz

Dim Freq_step_fine_flag As Byte                    'Flag, ob FM-Step=10kHz, AM-Step=1 kHz (Einschalten S3 gedrückt)


'*** Senderspeicher ***


Dim Am_memory(50) As Word                     '30 AM-Sendespeicher (Frequenz, 2 Byte)

Dim Am_memorypos As Byte                      'aktueller AM-Speicherplatz

Dim Am_memorymin As Byte                      'min. Speicherplatz in gewähltem Speicherbereich

Dim Am_memorymax As Byte , Scan_memorymax As Byte          'Anzahl benutzter Speicherplätze AM-Sender;

Dim Lw_memorymax As Byte , Mw_memorymax As Byte , Kw1_memorymax As Byte , Kw2_memorymax As Byte , Kw3_memorymax As Byte

Dim Am_text(50) As String * 10                'AM-Sendernamen-Speicher a´ 10 Byte

Dim Am_textpos As Byte                        'Positionszeiger für AM-Sendernamen-Eingabe

Dim Am_textaktiv As Bit                       'Flag für AM-Sendernamen-Eingabe

Dim Fm_memory(50) As Word                     '30 FM-Sendespeicher (Frequenz, 2 Byte)

Dim Fm_memorypos As Byte                      'aktueller FM-Speicherplatz

Dim Fm_memorymax As Byte                      'Anzahl benutzter Speicherplätze FM-Sender

Dim Memory_aktiv_flag As Bit                  'im Speicher-Scroll-Modus


'*** Si4735 ***


Dim Rssi As Word                              'Signalstärke dBuv

Dim Snr As Word                               'SNR (dB)

Dim Stereo_signal As Byte                     'Stereo-Signal in %

Dim Freq_offset As Byte                       '+/- Frequenz-Abweichung in kHz
```

```
Dim Pwm_output As Byte                    'Flag für PWM-Ausgang (0=Stereo-Signal, 1=Freq.-Offset, 2=RSSI, 3=SNR)

Dim C As Long                             'Antennten-Kapazität

Dim Fin As Long                           'Eingabe-Frequenz

Dim H As Byte , L As Byte                 'HilfsVariable High-/Low-Byte

Dim Vol As Byte                           'Volume

Dim Status As Byte                        'Statusbyte aus Si4735

Dim Bltf As Bit                           'Statusbyte für erreichen der Bandgrenze bei Suchlauf

Dim Valid As Bit                          'Flag für gültige Frequenz (über Pegel- und SNR-Schwelle)

Dim R1 As Byte , R2 As Byte , R3 As Byte , R4 As Byte     '12* gelesene Response-Bytes aus Si4735

Dim R5 As Byte , R6 As Byte , R7 As Byte , R8 As Byte

Dim R9 As Byte , R10 As Byte , R11 As Byte , R12 As Byte

Dim Property_adr As Word                  'Property-Adresse

Dim Dat As Byte , Dat_word As Word        'Datenbyte (u.a. Property-Low-Byte)

Dim Property_dat(16) As Byte              '16 Byte Tuning-Parameter f. Si4735

Dim Property_adress(16) As Word

Dim Property_max(16) As Byte

Dim Property_min(16) As Byte

Dim Property_default(16) As Byte

Dim Property_aktiv As Byte                'Flag für Encoder-Eingabe Properties (1=Property-Scroll-Modus, 2=Property-Einstell-Modus)



'*** RDS-Variablen ***


Dim Rds_flag As Bit                       'RDS-Signal vorhanden

Dim Rds_block_errors As Byte : Rds_block_errors = 1     'max. zulässige RDS-Block-Fehler: 0=0 Errors, 1=1...2 errors correctable, 2=3...5 errors correctable, 3=uncorrectable

Dim Block_errors_block_a As Byte , Block_errors_block_b As Byte , Block_errors_block_c As Byte , Block_errors_block_d As Byte     'Errors Block A,B,C,D
```

```
Dim Service As Byte                          'Gruppennummer
Dim Tp As Byte , Ta As Byte                  'Flag für Travel-Program, Travel Announcement
Dim Pi As Word                               '16 bit Programm-Identification-Nummer
Dim Ms_flag As Bit                           'Music/Speech
Dim Di_counter As Byte                       'Decoder Information-Zähler modulo 4
Dim Di(4) As Byte                            'Decoder-Informationen
Dim Pty As Byte , Ptyn As Byte               'Program-Type,Program-Type Name
Dim Pty_name As String * 8 , Pty_name_flag As Byte   'PTYN aus Gruppe 10A

Dim Ps_name As String * 8 , Ps_name_last As String * 8   'Sendername aus RDS, 8 Zeichen
Dim Ps_counter As Byte , Ps_flag As Bit      'Zähler für decodierte 2Byte-PS Blocks, Flag für
komplette Dekodierung

Dim Af_flag As Byte                          'Flag für gefundene AFs, 1=gefundene, 2=Liste fertig
Dim Af(25) As Word                           'Alternate Frequencies
Dim Freq_af As Word
Dim Af_max As Byte                           'Anzahl von AF
Dim Af_number As Byte                        'laufende Nummer AF
Dim Lw_mw_af As Bit                          'Flag für folgende LW-MW-Frequenz in Block D
Dim Method_b As Bit                          'Flag für AF-Methode B
Dim Reg_flag As Bit                          'Flag für Regionalprogramm/-Frequenzen

Dim Eon_flag As Byte                         'Flag für EON-Dekodierung: 0 oder $FF = nicht empfangen; 1=
Start empfangen, 2=vollständig empfangen
Dim Ps_eon As String * 8                     'EON:PS
Dim Pty_eon As Byte                          'EON:PTY
Dim Tp_eon As Bit , Ta_eon As Bit            'EON:TP/TA
Dim Pi_eon As Word , Pic1_eon As Byte , Pic2_eon As Byte    'EON:PI
```

```
Dim Pin_code_eon As Word                    'EON:PIN

Dim Af_eon(12) As Word                      'EON: AF/Tuned-Freq. - Mapped Freq.

Dim Vc_counter_eon As Word                  'Zähler für Adress-Segment

Dim Link_info As Word                       'Linkage-Information


Dim La As Bit                               'Linkage Actuator-Flag

Dim Vc As Byte                              'Variant Code

Dim Ecc As Byte                             'Extended Country Code

Dim Lc As Word                              'Language Code

Dim Ews As Word , Ews0 As Byte , Ews1 As Byte , Ews2 As Byte , Ews3 As Byte , Ews4 As Byte
'Emergency Warning System

Dim Pin_code As Word                        'Program-Identification-Number

Dim Tmc_ident As Word                       'TMC-Idenfication


Dim Radiotext_on As Bit : Radiotext_on = 1  'Flag, ob RDS-Text angezeigt werden soll, oder nicht

Dim Radiotext_present As Bit                'Flag für Radiotext in RDS-Daten

Dim Radiotext_first_aqu As Bit              'Flag für Radiotext Erstaquisition (Grupppe 2A oder 2B in
RDS-Daten enthalten)

Dim Radiotext_start_decode As Byte          'Flag, daß RT-Dekodierung gestarrtet hat

Dim Segment As Byte , Position As Byte      'Segment-Nummer des letzten empfangen egmentes

Dim Char_counter As Byte                    'Zeichenzähler

Dim Radiotext As String * 64 , Radiotext_new As String * 64 'Radiotext, max. 64 Zeichen

Dim Radiotext_ab_flag As Byte , Radiotext_ab_flag_old As Byte   'Toggle-Bit für neue Radiotext-Info

Dim Radiotext_ready As Bit                  'Radiotext anzeigen --> Carriage Return oder alle Gruppen
gesendet in Radiotext

Dim Radiotext_display_pos As Byte           'Aktuelle, rechte Stelle des RT

Dim Radiotext_length As Byte , End_position As Byte     'Länge des Radiotext-Strings

Dim Radiotext_lf_pos As Byte                'Position des nächsten LF

Dim Display_string As String * 16           '16-Byte String für LCD Zeile 1

Dim Scroll_counter As Byte                  'Verzögerungszähler für 1. und letztes Zeichen beim Scrolling
```

```
Dim Direction As Bit                    '0=nach rechts, 1=nach links
Dim Replace_codes(127) As Byte          'Sonderzeichen-Tabelle
Dim Rds_over_rs232_control As Byte      'Control-Byte für RDS-Ausgabe An/Aus (an RS-232)

Dim Pos As Byte                         'Positionszeiger in Ausgabestring
Dim Ch As String * 1                    '1 Zeichen für String
Dim Tx As String * 2                    '1 Doppelzeichen für String
Dim Freq_string As String * 6           'Frequenzausgabe-String "108.45" für FM oder "1234" für
AM

Dim Minuten As Integer , Utc_min As Byte    'Minuten
Dim Stunden As Integer , Utc_hour As Byte   'Stunden
Dim Utc_offset As Byte
Dim Mjd As Long                         'Modified Julian Date
Dim Year As Integer , Month As Byte , Day As Byte , Wd As Long
Dim K As Byte , Ks As Single , Ms As Single , Ys As Single  'Hilfsvariablen

'*** RS-232 Variablen für I2C-Steuerung vom PC ***

Dim Command As Byte                     'eingelesenes Kommando-Byte
Dim Bytesout As Byte
Dim Bytesin As Byte
Dim Data_i2c(10) As Byte                'zu lesende/schreibende I2C-Daten




'********************************
'*** SUB-/FUNCTION Definitionen ***
```

```
'*********************************

Declare Function Si4735_get_property(byval Prop_adr As Word) As Word       'Hole Property-Wert Nr. N (1...16)


Declare Sub Si4735_get_status_byte()                    'Status-Byte holen. Bit0 =

Declare Sub Si4735_wait_for_seek_tune_cplt()            'Warte, bis Status-Bytet.0 = 1 = STC = Seek/Tune Complete --> bereit zum nächsten Befehlsempfang

Declare Sub Si4735_wait_for_clear_to_send()             'Warte bis Status-Byte.7 = CTS = "Clear to Send"

Declare Sub Si4735_set_rx_volume()                      'Property "Volume" setzen

Declare Sub Si4735_set_property(byval Prop_adr As Word , Byval Prop_value_high As Byte , Byval Prop_value_low As Byte)       'Property setzen

Declare Sub Si4735_print_property_lcd()                 'Property-Wert Nr. N auf LCD ausgeben

Declare Sub Si4735_init()                               'Ruft Init=Powerup, Si4735_set_rx_volume, FM_Tune_Freq, Am_tune_status_freq_print_lcd, Rds_init auf; Setzt Properties 1-8

Declare Sub Si4735_get_rev()                            'Hole IC, Firmware und Rev. und schreibe auf LCD


Declare Sub Am_tune_freq()

Declare Sub Am_seek_freq_up()

Declare Sub Am_seek_freq_down()

Declare Sub Am_tune_status_freq_print_lcd()

Declare Sub Am_stop_seek_tune()

Declare Sub Am_rsq_status()

Declare Sub Am_seek_step()                              'Setze AM-Frequenzschritt

Declare Sub Call_am_memory(am_memory_number As Byte)

Declare Sub Call_fm_memory(fm_memory_number As Byte)

Declare Sub Write_ammemories_eeprom(stored_memories As Byte , Byval Start_memory As Word)
'Schreibe Anzahl gespeicherter Sender in AM-Bändern ins EEPROM

Declare Sub Read_ammemories_eeprom(stored_memories As Byte , Byval Start_memory As Word)
```

```
Declare Sub Fm_tune_freq()

Declare Sub Fm_seek_freq_up()

Declare Sub Fm_seek_freq_down()

Declare Sub Fm_tune_status_freq_print_lcd()

Declare Sub Fm_stop_seek_tune()

Declare Sub Fm_rsq_status()

Declare Sub Lcd_freq(byval Fm_freq As Word , Byval Y_pos As Byte , Byval X_pos As Byte)      'Drucke FM_Freq in LCDPosition Y,X

Declare Sub Lcd_scanning                    'drucke "Scanning..." in untere LCD-Zeile, lösche "mxx " rechts oben


Declare Sub Eeprom_save()                   'Frequenzdaten in EEPROM sichern


Declare Sub Rds_init()                      'RDS-Initialisierung

Declare Sub Rds_reset()                     'RDS-Variablen zurücksetzen nach SYNC-Lost oder neuem Sender

Declare Sub Rds_get_status_and_data()           'hole 1 RDS-Gruppe + RDS-Stausbytes R1...R14 aus Si4735

Declare Sub Rds_decoding()                  'Ausgabe RDS-Sendername und RDS-Zeit auf 2.Display-Zeile bzw. Radiotext

Declare Sub Rds_calc_af(af_index As Byte , Af_word As Word) 'Berechne Frequenz (FM/LW/MW) einer gültigen AF

Declare Sub Rds_char_code(chr_code As Byte)         'Ersetze RDS-ASCII-Code mit EADOGM-Code

Declare Sub Make_freq_string(byval Af_freq As Word)     'erzeuge Freq_string für formatierte AM-/FM-Frequenz


Declare Sub Radiotext_decoded()             '1x Radiotext-Message dekodiert

Declare Sub Radiotext_scroll()              'Scrolle Radiotext um 1 Zeichen nach rechts

Declare Sub Radiotext_start()               'Bringe Radiotext erneut in Anfangsposition
```

```
'Declare Sub Blockerrors_service()                    'Isoliere Blockerrors der 4 Blocks + Gruppennummer
Declare Sub Clear_lcd_lowerline()                     'untere LCD-Zeile löschen
Declare Sub Clear_lcd_upperline()                     'obere LCD-Zeile löschen
Declare Sub Reset_flags()                             'Eingabeflags zurücksetzen
Declare Sub Reset_encoder()

Declare Sub F_control()
Declare Sub Mam_control()
Declare Sub Mfm_control()
Declare Sub Pc_control_i2c()
Declare Sub Properties()
Declare Sub Rds_print_control()                       'RDS-Ausgabe über RS-232 in Teilen ein- oder ausschalten (pro Bit 1 Segment, max. 8 Teilbereiche)



'**********************
'*** Initialisation ***
'**********************

Open "com1:" For Binary As #1                         'Kanal 1/Com1 für RS-232 öffnen (38400 Baud, 8N1)

Config Scl = Portc.5                                  'Entspricht Hardware-I2C; (evtl. andere I2C-Routine benutzen)
Config Sda = Portc.4
I2cinit
Config Twi = 400000                                   'I2C-Bus mit 400kHz
```

```
Config Adc = Single , Prescaler = Auto , Reference = Off    'AD-Wandler für Volume-Poti Abfrage; A_ref = ext = 3,3V

Start Adc


Config Timer1 = Pwm , Prescale = 8 , Pwm = 10 , Compare A Pwm = Clear Down       'TIMER1 als 10bit-PWM-Ausgabe für S-Meter (oder Stereo-Blend/Frequency-Offset)

Start Timer1

Pwm1a = 0


Config Timer0 = Timer , Prescale = 8                    'TIMER0 als Interruptquelle für Encoder-Abfrage


'On Ovf0 Tim0_isr Saveall

On Ovf0 Tim0_isr Nosave


Enable Timer0


Enable Interrupts


Config Lcdpin = Pin , Db4 = Portd.4 , Db5 = Portd.5 , Db6 = Portd.6 , Db7 = Portd.7 , E = Portd.3 , Rs = Portd.2

Config Lcd = 16 * 2 , Chipset = Dogm162v3


Initlcd

Cursor Off Noblink



'Deflcdchar 0 , 1 , 14 , 14 , 1 , 11 , 13 , 14 , 31                    'Inverses "R"

Deflcdchar 0 , 24 , 20 , 24 , 20 , 20 , 7 , 2 , 2          '"RT" = Radiotext-Symbol

'Deflcdchar 7 , 6 , 8 , 6 , 1 , 14 , 4 , 4 , 4                    '"ST" = Stereo-Symbol (alternativ)

Deflcdchar 7 , 12 , 16 , 12 , 2 , 12 , 7 , 2 , 2          '"ST" = Stereo-Symbol
```

```
Deflcdchar 1 , 2 , 5 , 5 , 6 , 5 , 5 , 6 , 8              'ß
Deflcdchar 2 , 32 , 4 , 14 , 4 , 32 , 14 , 32 , 32       '+/-
Deflcdchar 3 , 14 , 17 , 23 , 25 , 25 , 23 , 17 , 14     'Copyright
Deflcdchar 4 , 24 , 25 , 2 , 4 , 8 , 16 , 27 , 27        'Promille
Deflcdchar 5 , 4 , 14 , 21 , 4 , 4 , 4 , 4 , 32          'Pfeil n. oben
Deflcdchar 6 , 32 , 4 , 2 , 31 , 2 , 4 , 32 , 32         'Pfeil n. rechts

'Deflcdchar 7 , 4 , 4 , 4 , 4 , 21 , 14 , 4 , 32                'Pfeil n. unten
'Deflcdchar 2 , 16 , 15 , 15 , 17 , 30 , 30 , 1 , 31            'Inverses "S"
'Deflcdchar N , 32 , 16 , 8 , 4 , 2 , 1 , 32 , 32              '\
'Deflcdchar N , 15 , 20 , 20 , 23 , 20 , 20 , 15 , 32          'OE
'Deflcdchar N , 32 , 32 , 10 , 21 , 22 , 20 , 11 , 32          'oe

Cls                                     'LCD-Einschalt-Meldung
'Cursor Off Noblink



'*** Band_start/-end-Frequenzen und Bandname ***

Restore Band_data

For N = 1 To 17
   Read Band_start(n)
   Read Band_end(n)
Next N
```

```
'*** Property-Grenzwerte einlesen ***

Restore Property_data

For N = 1 To 16                         'Tuning-Parameter f. Si4735 aus  zurücklesen EEPROM-Adresse 1006...1023
   Read Property_adress(n)                  'Property-Adressen, Min./Max.Werte aus Data-Zeilen einlesen
   Read Property_min(n)
   Read Property_max(n)
   Read Property_default(n)
Next N


'*** Sonderzeichen-Ersatztabelle ab ASCII 128 für EADOGM-Zeichensatz vs. RDS-Zeichensatz laden ***

Restore Special_characters
For N = 1 To 127
   Read Replace_codes(n)
Next N


'*** Clear Memory = Taste S5 ("Memory") beim Einschalten gedrückt ***

If S5 = 0 Then                          'Falls beim Einschalten S5 ("Memory") länger als 0,3s gedrückt --> alle Tuning-Parameter+Senderspeicher löschen
```

```
  D = 0
  Writeeeprom D , 0                    'Anzahl gespeicherter AM-/FM-Sender = 0
  Writeeeprom D
  Writeeeprom D , 995                  '5xAM-Speicher
  Writeeeprom D
  Writeeeprom D
  Writeeeprom D
  Writeeeprom D

  Text1 = Space(10)
  For I = 202 To 741 Step 11           'AM-Senderspeichernamen löschen
    Writeeeprom Text1 , I
  Next I

  For N = 1 To 16                      'Tuning-Parameter mit Default-Wert füllen und ins EEPROM
schreiben
    Property_dat(n) = Property_default(n)
    D = Property_default(n)
    I = N + 1005
    Writeeeprom D , I
  Next N

End If

'*** letze Einstellwerte aus EEPROM einlesen und prüfen, ob gültig ***

Readeeprom Am_fm_flag , 1000           'letzen AM/FM Modus aus EEPROM zurücklesen
```

```
Readeeprom Freq_fm                    'letzte Frequenzen zurücklesen'

Readeeprom Freq_am

Readeeprom Band


If Am_fm_flag > 1 Or Freq_fm < Band_start(17) Or Freq_fm > Band_end(17) Or Freq_am < Band_start(1) Or Freq_am > Band_end(16) Then

  Am_fm_flag = 0                    'Starte mit FM-Modus

  Freq_fm = 10080                   'FM-Default-Wert = 100.80 MHz

  Freq_am = 549                     'AM-Default-Frequenz= DLF

  Band = 2

End If


If Freq_am < Band_start(band) Or Freq_am > Band_end(band) Then      'Falls AM-Frequenz außerhalb Bandgrenzen...

  Freq_am = Band_start(band)                '--> AM-Frequenz auf Bandstart-Anfang

End If



Readeeprom Fm_memorymax , 1                 'Anzahl gespeicherter FM-Sender aus EEPROM-Adresse 0 einlesen

If Fm_memorymax > 50 Then Fm_memorymax = 0          'falls EEPROM-Speicher unbeschrieben --> Anzahl gespeicherter FM-Sender=0 setzen



If Fm_memorymax > 0 Then                    'falls FM-Senderspeicher EEPROM nicht leer --> FM-Senderspeicher einlesen

  For N = 1 To Fm_memorymax

    I = N * 2                       'EEPROM-Adresse für FM-Frequenz berechnen

    I = I + 100

    Readeeprom Freq , I             'FM-Speicherfrequenzen einlesen
```

```
    Fm_memory(n) = Freq
  Next N
End If

Readeeprom Lw_memorymax , 995                    '5x genutzte AM-Speicher-Endadressen
Readeeprom Mw_memorymax , 996
Readeeprom Kw1_memorymax , 997
Readeeprom Kw2_memorymax , 998
Readeeprom Kw3_memorymax , 999

Call Read_ammemories_eeprom(lw_memorymax , 1)          'AM-Speicherfrequenzen und Speichernamen zurücklesen
Call Read_ammemories_eeprom(mw_memorymax , 11)
Call Read_ammemories_eeprom(kw1_memorymax , 21)
Call Read_ammemories_eeprom(kw2_memorymax , 31)
Call Read_ammemories_eeprom(kw3_memorymax , 41)

'*** letzte Tuningparameter (Properties) aus EEPROM einlesen ***

For N = 1 To 16                                  'Tuning-Parameter f. Si4735 aus  zurücklesen EEPROM-Adresse 1006...1023
  I = N + 1005
  Readeeprom Dat , I
  Property_dat(n) = Dat

  If Dat < Property_min(n) Or Dat > Property_max(n) Then  'eingelesener EEPROM-Wert außerhalb Wertebererich
```

```
      Property_dat(n) = Property_default(n)
   End If
Next N


Readeeprom Pwm_output , 1022                    'Flag für PWM-Ausgabe FM einlsen
If Pwm_output > 3 Then Pwm_output = 0            'auf Gültigkeit prüfen


Readeeprom Freq_step_fine_flag , 1023            'Flag für manuelle Schrittweite einlesen
If Freq_step_fine_flag > 1 Then Freq_step_fine_flag = 0     'auf Gültigkeit prüfen


Lcd "DSP-Radio V1.1.1"
Home Lower
Lcd "  Extended RDS"
Wait 1


'*** Ändere PWM-Ausgabe auf Freq-Offset, falls Taste S2 ("FM") beim Einschalten gedrückt ***


Cls
Lcd "PWM Analog-Out:" : Home Lower
```

```
If S2 = 0 Then                          'Falls beim Einschalten S2 ("FM") gedrückt --> PWM-Ausgabe aus FM-Freq-Offset anstelle Stereo-Signal
  Incr Pwm_output : If Pwm_output > 3 Then Pwm_output = 0
  Writeeeprom Pwm_output , 1022         'Flag für PWM-Ausgabe FM in EEPROM schreiben
End If

Select Case Pwm_output
      Case 0 : Print "FM Stereo-Level"
      Case 1 : Print "FM Freq-Offset"
      Case 2 : Print "Feldstaerke"
      Case 3 : Print "SNR"
End Select

If Pwm_output = 1 Then
  Lcd "FM Freq-Offset"
Else
  Lcd "FM Stereo-Level"
End If

Wait 2

If S3 = 0 Then                          'Falls beim Einschalten S3 ("Band") gedrückt --> manueller Frequenzschritt
  Toggle Freq_step_fine_flag.0
  Writeeeprom Freq_step_fine_flag , 1023    'Flag für manuelle Frequenzschrittweite iin EEPROM schreiben
End If
```

```
Cls
If Freq_step_fine_flag = 1 Then
  Lcd "FM-Step:10 kHz"
  Lowerline : Lcd "AM-Step: 1 kHz"
Else
  Reset Freq_step_fine_flag
  Lcd "FM-Step:50 kHz"
  Lowerline : Lcd "AM-Step: 5 kHz"
End If

Wait 2



'(
Pot = Getadc(14) : Pot = Getadc(14)                    '1.1V Bandgap-Referenz messen
Float1 = Pot : Float1 = Float1 / 1024 : Float1 = 1.1 / Float1
Cls : Lcd "Betriebsspannung"
Lowerline : Lcd "      " ; Fusing(float1 , "#.#") ; " V" : Wait 1

Call Si4735_get_rev()                    'Hardware/Firmware Daten holen und anzeigen
')
'Vol = 45                                 'Default Volume Wert
```

```
'*** Si 4735 initialisieren ***
Call Si4735_init                        '--> Starte FM/AM-Modus in Si4735



'*****************                                    k
'*****************
'*** MAIN-LOOP ***
'*****************
'*****************

Do



  '************ RS-332 Control**************************
  '*** 1 Steuerzeichen einlesen und zu SUB verzweigen ***
  '*****************************************************


   D = Inkey(#1)
   Select Case D

        Case 102 : Call F_control              '"f", Freq
        Case 109 : Call Mam_control             '"m", Memory AM
        Case 110 : Call Mfm_control             '"n", Memory FM
        Case 112 : Call Properties              '"p", Property
        Case 105 : Call Pc_control_i2c          '"i", I2C command
        Case 106 : Call Rds_print_control       '"j": RDS output On (Sendername, Zeit, Radiotext
ausgeben)
```

```
      Case 107 : Rds_over_rs232_control = 0          '"k": RDS output Off
      Case 114 : Print Rssi                          '"r": RSSI
      Case 115 : Print Snr                           '"s": SNR

  End Select

  Waitms 1



'************************
'*** Encoder auswerten ***
'************************


'*** Encoder: Property-Anzeige/Einstell-Modus ***

If Encoder1old <> Encoder1 Then

   Disable Timer0

   If Property_aktiv <> 0 Then                   'im Property-Einstell-Modus?

     'Properties scrollen

     If Property_aktiv = 1 Then                  'im Property-Scroll-Modus?
```

```
        If Encoder1 > 0 Then Incr N                 'Property-Adresse erhöhen oder verkleinern und auf Intervall
9...16 bringen
        If Encoder1 < 0 Then Decr N

        If N > 16 Then N = 9                        'Property-Scroll nur im Intervall 1...8 (FM) oder 9...16 (AM)
        If N = 0 Then N = 8
        If N = 8 And Am_fm_flag = 1 Then N = 16
        If N = 9 And Am_fm_flag = 0 Then N = 1

      Elseif Property_aktiv = 2 Then                'im Property-Setz-Modus (Property_aktiv=2)? --> Je nach
Property Max./Min.Werte einhalten

        Dat = Property_dat(n)

        If Encoder1 > 0 Then                        'Property-Wert erhöhen oder verkleinern und auf zulässiges
Intervall bringen

          If Dat < Property_max(n) Then
            Incr Dat
          Else
            Dat = Property_min(n)                   'Rollover
          End If

        End If

        If Encoder1 < 0 Then

          If Dat > Property_min(n) Then
            Decr Dat
          Else
```

```
            Dat = Property_max(n)
          End If

        End If

        Property_dat(n) = Dat
        Property_adr = Property_adress(n)            'Neuen Property-Wert setzen
        Call Si4735_set_property(property_adr , &H00 , Dat)

      End If

      Call Si4735_print_property_lcd                 'Property auf LCD ausgeben
      Call Reset_encoder



  '*** Encoder: AM-Modus ***

  Elseif Am_fm_flag = 1 Then                         'im AM Modus?

'     Disable timer0

      If Am_textaktiv = 1 Then                       'AM-Text Senderspeichernamen-Eingabe?

        'Namenseingabe AM-Sender

        Ch = Mid(am_text(am_memorypos) , Am_textpos , 1)      'aktuelles Zeichen aus Am_text isolieren
        I = Asc(ch)
        If Encoder1 > 0 Then Incr I                  'ASCII-Wert 1 erhöhen oder erniedrigen
```

```
        If Encoder1 < 0 Then Decr I
        If I = 126 Then I = 32                  'Bei Überlauf auf "SPACE" zurücksetzen
        If I = 31 Then I = 125                  'Bei Unterlauf auf "~" zurücksetzen
        Ch = Chr(i)                             'ASCII-Wert in Zeichen zurückverwandeln
        Mid(am_text(am_memorypos) , Am_textpos , 1) = Ch      'Zeichen in Am_text zurückspeichern
        I = Am_textpos + 6
        Locate 2 , I
        Lcd Ch                                  'neues Zeichen ausgeben
        Locate 2 , I

        Call Reset_encoder

      Elseif Memory_aktiv_flag = 1 Then         'im Memorymodus? --> Speicher durchscrollen

       'Speicher-Scrolling

       If Am_memorymax > 0 Then                 'Scrolling nur, falls überhaupt mind. 1 Sender gespeichert

         If Encoder1 > 0 Then Incr Am_memorypos      'Speicherzeiger +/- 1
         If Encoder1 < 0 Then Decr Am_memorypos

        If Am_memorypos > Am_memorymax Then Am_memorypos = Am_memorymin      'Rollover bei
Überschreiten
        If Am_memorypos < Am_memorymin Then Am_memorypos = Am_memorymax      'Rollover bei
Unterschreiten

         Call Call_am_memory(am_memorypos)

       End If
```

```
    Else                                'sonst --> Frequenz je nach Band um 1/5/9 kHz erhöhen

     'Frequenz-Erhöhung

      Freq = Encoder1 * 5                    '5 kHz-Default-Schrittweite

      If Freq_step_fine_flag = 0 Then
        If Band < 3 Then Freq = Encoder1 * 9        'Im Grobmodus: LW/MW --> 9 kHz
      Else
        Freq = Encoder1                     'Im Feinmodus --> 1 KHz
      End If

      Freq_am = Freq_am + Freq

      If Freq_am < Am_lower_limit Then Freq_am = Am_lower_limit       'absolute Frequenzgrenzen
checken
      If Freq_am > Am_upper_limit Then Freq_am = Am_upper_limit

      Call Am_tune_freq                     'Frequenz einstellen

      Call Am_tune_status_freq_print_lcd

      Call Reset_encoder

    End If
```

```
    '*** Encoder: FM-Modus ***

    Elseif Am_fm_flag = 0 Then                    'im FM-Modus, dann...

      '     Disable Timer0                        'Encoder verändert --> Frequenz- oder Memory-
Inc/Decr

        If Memory_aktiv_flag = 1 Then             'im Speichermodus? --> Speicher durchscrollen

          If Fm_memorymax > 0 Then

            If Encoder1 > 0 Then Incr Fm_memorypos
            If Encoder1 < 0 Then Decr Fm_memorypos

            If Fm_memorypos > Fm_memorymax Then Fm_memorypos = 1
            If Fm_memorypos < 1 Then Fm_memorypos = Fm_memorymax

            Call Call_fm_memory(fm_memorypos)

          End If

        Else                                      'sonst --> FM-Frequnez um +/-50 kHz verändern

          If Freq_step_fine_flag = 1 Then         'Frequenz-Schrittweite berechnen
            Freq = Encoder1
          Else
            Freq = Encoder1 * 5
          End If
```

```
        Freq_fm = Freq_fm + Freq

        If Freq_fm < Band_start(17) Then Freq_fm = Band_end(17)      'FM-Bandgrenzen checken -->
Rollover
        If Freq_fm > Band_end(17) Then Freq_fm = Band_start(17)

        Call Fm_tune_freq                       'und einstellen

        Call Fm_tune_status_freq_print_lcd

        Call Reset_encoder

      End If

   End If

 End If


 '*********************************
 '*** Display periodisch erneuern ***
 '*********************************

 '*** Frequenz, Tuning-Details und ggf. RDS/Radiotext auf Display***


 If Property_aktiv = 0 Then                     'falls gerade nicht im Property-Einstell-Modus und RDS-Data
present...
```

```
    Call Rds_decoding()                    'RDS-Loop; hole die letzten Gruppen aus FIFO-Speicher, sobald
mind. 14 Gruppen present
  End If



  Incr T                                   'Schleifenzähler für Mainloop bis 200

  If T = 100 Then T = 0

  If T = 25 Then                           'bei T=50: Frequenz auf Display, falls Radiotext_ready=0

    If Valid = 0 Then                      'Bei noch ungültiger Frequenz...

      If Am_fm_flag = 0 Then               'FM-Modus?

        If Radiotext_ready = 1 Then
          If Scroll_counter > 0 Then       'Verzögerungszähler
            Decr Scroll_counter
          Else
            Call Radiotext_scroll          'Scrolle Radiotext um 1 Zeichen weiter
          End If
        Else
          Call Fm_tune_status_freq_print_lcd    '--> FM-Fequenz Auf Display (wenn kein Radiotext
decodiert)
        End If

      Else                                 'AM-Modus?
        Call Am_tune_status_freq_print_lcd      '--> Am-Status/Frequenz-ausgabe
      End If
```

```
  End If

End If

If T = 50 Then                          'bei T=100: Tuning-Details und ggf. RDS auf Display

  If Am_fm_flag = 0 Then                  'FM-Modus

    Call Fm_rsq_status                    '--> FM-Tuning-Details

    If Radiotext_ready = 1 Then             'Radiotext-Anzeige
      If Scroll_counter > 0 Then            'Verzögerungszähler
        Decr Scroll_counter
      Else
        Call Radiotext_scroll               'Scrolle Radiotext um 1 Zeichen weiter
      End If
    End If

  Else                                  'AM-Modus
    Call Am_rsq_status                      '-->AM-Tuningdetails auf LCD
  End If

End If

If T = 0 Or T = 75 And Radiotext_ready = 1 Then

  If Radiotext_ready = 1 Then             'Radiotext-Anzeige
    If Scroll_counter > 0 Then            'Verzögerungszähler
```

```
      Decr Scroll_counter
    Else
      Call Radiotext_scroll                  'Scrolle Radiotext um 1 Zeichen weiter
    End If
  End If

End If



'*****************************
'*** Tastendrücke auswerten ***
'*****************************

'*** S2 = FM --> Seek-FM Up/Down oder Band-Scan FM ***

  If S2 = 0 Then

    Call Reset_flags                       'Eingabeflags zurücksetzen
    Call Rds_reset

    If Am_fm_flag = 1 Then                     'falls noch nicht im FM-Modus...
      Am_fm_flag = 0                        ' --> FM-Modus aktivieren
      T1 = 0
      Call Si4735_init

    Else

      T1 = 0
      Waitms 5                             'debounce
```

```
Do
   Waitms 1
   Incr T1
Loop Until S2 = 1

If T1 < 500 Then                        'short (<500ms)--> Seekup

   'FM-Seekup

   Valid = 0
   Call Fm_seek_freq_up                 'Stop bei Erreichen der Bandgrenze
   Call Clear_lcd_lowerline
   Waitms 500
   Fm_tune_status_freq_print_lcd

Elseif T1 > 499 And T1 < 2000 Then      'long (500...2000ms) --> Seekdown

   'FM-Seekdown

   Valid = 0
   Call Fm_seek_freq_down
   Call Clear_lcd_lowerline
   Waitms 500
   Fm_tune_status_freq_print_lcd

Else                                    'very long (2000ms Scan FM-Band and Store 1...n)

   'FM-Bandscan
```

```
Fm_memorymax = 0 : Fm_memorypos = 0 : Memory_aktiv_flag = 0      'Speicher auf Null setzen
Freq_fm = Band_start(17)                  'stelle Band_start FM ein (z.B. 87,50 MHz)
Call Fm_tune_freq                         'Tune auf Band_start
Bltf = 0                                  'Wraparound-Indikator

Cls : Lcd "FM-Bandscan" : Wait 1 : Cls

Do                                        'Schleife für kompletten Bandsuchlauf
  Call Fm_seek_freq_up                    'FM-Suchmodus aufwärts starten
  Call Fm_tune_status_freq_print_lcd      'gefundene Frequenz aus Si4735 holen und anzeigen
  Call Fm_rsq_status                      'RSSI und SNR anzeigen
  Incr Fm_memorymax                       'Memoryzeiger + 1
  Fm_memory(fm_memorymax) = Freq_fm       'Frequenz abspeichern

Loop Until Bltf <> 0 Or Fm_memorymax = 50     'solange, bis Bandgrenze erreicht oder 30 Speicher
voll

If Fm_memorymax > 0 Then                  'Sender gefunden...

  If Bltf = 1 Then Decr Fm_memorymax      'Bei Bandgrenze wird letzter Speicher doppelt belegt

  Fm_memorypos = 1                        'Zeiger auf 1. gefundenen Sender in Memory stellen
  Memory_aktiv_flag = 1
  Call Clear_lcd_lowerline
  Freq_fm = Fm_memory(fm_memorypos)
  Call Fm_tune_freq
  Locate 1 , 14                           '"Mnn" in Zeile 1, Spalte 14 schreiben
  Lcd "M" ; Fm_memorypos ; " "
```

```
        End If

      End If

    End If

  End If



'*** S3 = AM Band Up/Down  ***

  If S3 = 0 Then

    Call Reset_flags                    'Eingabeflags zurücksetzen
    Call Rds_reset

    T2 = 0
    Waitms 5                            'debounce

    Do
      Waitms 1
      Incr T2
    Loop Until S3 = 1

    If Am_fm_flag = 0 Then              'Falls noch nicht im AM-Modus --> AM-Modus starten
      Am_fm_flag = 1
      Call Si4735_init
```

```
Else
  If T2 < 500 Then                    'short-->nächst höheres Band / long--> nächst niedrigeres Band
    Incr Band
  Else
    Decr Band
  End If
End If

If Band > 16 Then Band = 1            'Bringe BAND auf Intervall 1...16
If Band = 0 Then Band = 16            'mit Rollover

Call Am_stop_seek_tune                'Stoppe evtl. Tuning

Select Case Band                      'Setze Anfang der Memmorybereichs
      Case 1 : Am_memorypos = 1 : Am_memorymax = Lw_memorymax
      Case 2 : Am_memorypos = 11 : Am_memorymax = Mw_memorymax
      Case 3 To 6 : Am_memorypos = 21 : Am_memorymax = Kw1_memorymax
      Case 7 To 10 : Am_memorypos = 31 : Am_memorymax = Kw2_memorymax
      Case 11 To 16 : Am_memorypos = 41 : Am_memorymax = Kw3_memorymax
End Select

Am_memorymin = Am_memorypos           'Halte min. Speicherplatz im Speicherbereioch fest

Band_name = Lookupstr(band , Band_names)
Cls : Lcd Band_name : Lowerline
Lcd Band_start(band) ; "-" ; Band_end(band) ; " kHz" : Wait 1
Cls
```

```
    Freq_am = Band_end(band) : Hi = High(freq_am) : Lo = Low(freq_am)      'Setze Suchlauf-Ende auf Bandende
    Call Si4735_set_property(&H3401 , Hi , Lo)

    Freq_am = Band_start(band) : Hi = High(freq_am) : Lo = Low(freq_am)      'Setze Suchlauf-Anfang auf Anfang aktuelles Band
    Call Si4735_set_property(&H3400 , Hi , Lo)

    Call Am_tune_freq                       'Starte mit Band_start-Frequenz
    Waitms 100
    Call Am_tune_status_freq_print_lcd

    If Band < 3 Then                        'Setze Suchlauf-Schrittweite
      Seek_step = 9                         '9kHz in LW und MW
    Else
      Seek_step = 5                         '1kHz in KW
    End If

    Call Am_seek_step()

  End If

'*** S4 = AM Seek --> Seek-AM Up/Down oder Band-Scan FM ***

  If S4 = 0 Then

    Call Reset_flags                        'Eingabeflags zurücksetzen
    Call Rds_reset
```

```
If Am_fm_flag = 1 Then                    'schon im AM-Modus?

  T3 = 0
  Waitms 5                                'debounce
  Do
    Waitms 1
    Incr T3
  Loop Until S4 = 1

  If T3 < 500 Then                        'short (<0,5s)

    'AM-Seekup

    Valid = 0
    Call Am_seek_freq_up
    Call Clear_lcd_lowerline
    Waitms 500
    Am_tune_status_freq_print_lcd

  Elseif T3 > 499 And T3 < 2000 Then      'long (500...2000ms) --> Seekdown

    'AM-Seekdown

    Valid = 0
    Call Am_seek_freq_down
    Call Clear_lcd_lowerline
    Waitms 500
    Am_tune_status_freq_print_lcd
```

```
Else                                'very long (>2000ms) Scan AM-Band and Store 1...n)

 'AM-Band-Scan

  If Band = 1 Then                  'LW: von Speicher 1...10 füllen
    Scan_memorymax = 0              'Speicher-Startposition - 1
    Am_memorypos = 10               'Speicher-Maximum

  Elseif Band = 2 Then              'MW: von Speicher 11...20 füllen
    Scan_memorymax = 10
    Am_memorypos = 20

  Elseif Band > 2 And Band < 7 Then     'KW: Tropenbänder 3-6 = 120-60m von Speicher 21...30
füllen
    Scan_memorymax = 20
    Am_memorypos = 30

  Elseif Band > 6 And Band < 11 Then    'KW: Bänder 7-10 = 49-25m von Speicher 31...40 füllen
    Scan_memorymax = 30
    Am_memorypos = 40

  Elseif Band > 10 And Band < 17 Then   'KW: Bänder 11-16 = 22-11m von Speicher 41...50 füllen
    Scan_memorymax = 40
    Am_memorypos = 50

  End If

  Freq_am = Band_start(band)            'stelle Band_start-Frequenz AM-Band ein
```

```
Call Am_tune_freq

Bltf = 0                              'Wraparound-Flag zurücksetzen

Band_name = Lookupstr(band , Band_names)
Cls : Lcd "Scan " ; Band_name : Wait 1 : Cls

Do

  Call Am_seek_freq_up                'AM-Suchmodus aufwärts starten
  Call Am_tune_status_freq_print_lcd  'gefundene Frequenz aus Si4735 holen und anzeigen
  Call Am_rsq_status                  'RSSI und SNR anzeigen

  Incr Scan_memorymax                 'Memoryzeiger + 1

  Am_memory(scan_memorymax) = Freq_am 'Frequenz abspeichern
  Am_text(scan_memorymax) = Space(10) 'Sendername löschen

Loop Until Bltf <> 0 Or Scan_memorymax = Am_memorypos  'solange, bis Bandgrenze erreicht

'  If Bltf = 1 Then Decr Scan_memorymax                'Bei Bandgrenze wird letzter Speicher doppelt belegt
Am_memorypos = Am_memorypos - 9       '--> Zeiger auf Anfang des 10er-Speicerhblocks legen

If Valid = 0 Then                     'letzter Sender ungültig --> kein Sender gefunden
  Freq_am = Band_start(band)          'Frequenz auf Bandanfang setzen
Else
  Freq_am = Am_memory(am_memorypos)
End If
```

```
      Call Clear_lcd_lowerline

      Call Am_tune_freq
      Call Am_tune_status_freq_print_lcd

      If Valid = 0 Then                    'falls M1 keine gültige Frequenz --> keine Frequenz gefunden
        Memory_aktiv_flag = 0
      Else
        Memory_aktiv_flag = 1
        Home Lower
        Lcd "M" ; Am_memorypos ; " "
      End If

    End If

  Else                                     'noch nicht im AM-Modus

    Am_fm_flag = 1                         'AM-Modus einschalten
    T = 0
    Call Si4735_init

  End If

End If

'*** S5: Memory  ***
```

```
'je einmal lang drücken (lang) toggelt Recall Modus an/aus

'Speicherauswahl im Recall-Modus mit Dreh-Encoder

'falls Recall-Modus aktiv und dann kurzer Tastendruck für "Store" --> Recall-Modus wird ausgeschaltet
'und Frequenz nicht (!) gespeichert

'2s drücken --> Property-Einstell-Modus starten (Scroll mit Encoder)
'        --> kurz drücken --> gewählte Property mit Encoder verändern
'        --> kurz drücken --> Property-Einstell-Modus beenden

'5s drücken --> alle Daten ins EEPROM speichern

  If S5 = 0 Then

    T4 = 0
    Waitms 5                          'debounce

    Do
     Waitms 1
     Incr T4
    Loop Until S5 = 1

   If T4 > 5000 Then                  'longer > 5000 ms --> alle Daten ins EEPROM

     Call Rds_reset

     Cls : Lcd "All data stored" : Lowerline : Lcd "into EEPROM" : Wait 2
```

```
    Writeeeprom Fm_memorymax , 1                'Anzahl belegter FM-Speicher ins EEPROM
    Writeeeprom Lw_memorymax , 995
    Writeeeprom Mw_memorymax , 996
    Writeeeprom Kw1_memorymax , 997
    Writeeeprom Kw2_memorymax , 998
    Writeeeprom Kw3_memorymax , 999

    Call Write_ammemories_eeprom(lw_memorymax , 1)   'Speichere alle AM-Senderspeicher ins
EEPROM
    Call Write_ammemories_eeprom(mw_memorymax , 11)
    Call Write_ammemories_eeprom(kw1_memorymax , 21)
    Call Write_ammemories_eeprom(kw2_memorymax , 31)
    Call Write_ammemories_eeprom(kw3_memorymax , 41)

    If Fm_memorymax > 0 Then

      For N = 1 To Fm_memorymax                 'FM-Frequenzspeicher ins EEPROM
        I = N * 2
        I = I + 100
        Freq = Fm_memory(n)
        Writeeeprom Freq , I
      Next N

    End If

    For N = 1 To 16                           'Properties ins EEPROM
      I = N + 1005
      Dat = Property_dat(n)
      Writeeeprom Dat , I
```

```
      Next N

    Elseif T4 > 2000 Then                    'Sehr langer Tastendruck (>2s) --> Properties-Setz-Modus

      Call Rds_reset

      Property_aktiv = 1                     'Property-Aktiv-Flag setzen

      If Am_fm_flag = 0 Then N = 1 Else N = 9      'Property No.1/9 als Start (ja nach AM/FM-Modus)

      Call Si4735_print_property_lcd         'Property anzeigen

    Elseif T4 > 500 Then                     'Langer Tastendruck (>500ms)= MEMORY-Recall Modus
einschalten

      Call Rds_reset

      If Memory_aktiv_flag = 0 Then          'noch nicht im Memorymodus?

        If Am_fm_flag = 1 Then               'AM-Modus --> "M" auf Display
          If Am_memorymax >= Am_memorypos Then    'gültige Speicher?
            Am_memorypos = Am_memorymin
            Call Call_am_memory(am_memorypos)
            Memory_aktiv_flag = 1            ' --> Memorymodus aktiv setzen
          End If

        Else                                 'FM-Modus
          If Fm_memorymax >= Fm_memorypos Then    'gültige Speicher
            Fm_memorypos = 1
```

```
        Call Call_fm_memory(fm_memorypos)
        Memory_aktiv_flag = 1                ' --> Memorymodus aktiv setzen
      End If

    End If

  Else

    Memory_aktiv_flag = 0                    'sonst: Memorymodus schon aktiv --> Memorymodus löschen

    If Am_fm_flag = 1 Then                   'AM-Modus
      Home Lower                             '"Mnn"-Anzeige auf Display löschen
      Lcd "   "
    Else
      Locate 1 , 14
      Lcd "   "
    End If

  End If

Else                                         'kurzer Tastendruck (<500ms) --> STORE MEMORY/STORE PROPERTY

  If Property_aktiv = 2 Then                 '--> im Property-Setz-Modus?

    Property_aktiv = 0                       'Property-Modus wieder ausschalten
    Call Clear_lcd_lowerline                 '2. LCD-Zeile wieder löschen

  Elseif Property_aktiv = 1 Then             'im Property-Scroll-Modus?
```

```
  Property_aktiv = 2                    '--> in Property-Setz-Modus wechseln

Elseif Memory_aktiv_flag = 0 Then       'Memorymodus nicht aktiv

  If Am_fm_flag = 1 Then                'AM-Modus?

    If Am_textaktiv = 0 Then            'nicht im AM-Text-Eingabemodus?

      'nächsten Speicher anwählen
      Incr Am_memorypos                 'Speicherzähler+1 mit Rollover auf ersten Speicherplatz
      If Am_memorypos > Am_memorymax Then Am_memorypos = Am_memorymin
      Am_memory(am_memorypos) = Freq_am 'Sender-Frequenz speichern (AM)
      Am_text(am_memorypos) = Space(10) 'Sendernamen löschen
      Home Lower                        '"Mnn" auf Display schreiben
      Lcd "M" ; Am_memorymax ; " "

      Am_textaktiv = 1                  'Texteingabe für AM-Sendernamen aktivieren
      Am_textpos = 1
      Locate 2 , 7                      'Cursor vor Start der Am_text-Eingabe setzen
      Cursor On Blink                   'Cursor anschalten

    Else

      'Texteingabemodus
      Incr Am_textpos                   'Zeichenzähler+1

      If Am_textpos = 11 Then           'letztes Zeichen erreicht?
        Am_textaktiv = 0                'Am_text-Eingabeflag zurücksetzen
        Cursor Off Noblink              'Cursor wieder ausschalten
```

```
          End If


        End If

      Else                              'FM-Mode


        Incr Fm_memorymax

        If Fm_memorymax > 50 Then Fm_memorymax = 50  'Speicherzähler+1 und bei Überlauf auf max.Wert belassen setzen

        Fm_memory(fm_memorymax) = Freq_fm          'Sender-Frequenz speichern (AM)

        Locate 1 , 14

        Lcd "M" ; Fm_memorymax ; " "


      End If

    Elseif Memory_aktiv_flag = 1 Then              'sonst: Memorymodus schon aktiv -->


      Memory_aktiv_flag = 0                    'Memorymodus ausschalten


      If Am_fm_flag = 1 Then                   'AM-Modus --> LCD-Display "Mnn" löschen

        Home Lower

        Lcd "   "

      Else

        Locate 1 , 14                        'LCD-Display "Mnn" löschen

        Lcd "   "

      End If


    End If
```

```
    End If


  End If



'*** Volume-Poti holen und setzen***


  Pot = Getadc(0)                    'Hole Poti-ADC
  Pot = Pot / 16                     'schneide 4 LSB ab
  Dif = Pot - Vol                    'Berechne Differenz zu aktuellem Vol-Wert
  Dif = Abs(dif)
  If Dif > 1 Then                    'nur bei Veränderung-->
    Vol = Pot                        'setze neuen Volume-Wert
    Call Si4735_set_rx_volume
  End If


Loop



'**********************
'**********************
'*** Ende MAIN-LOOP ***
'**********************
'**********************
```

```
'*********************
'*********************
'*** SUBs und IRSs ***
'*********************
'*********************
```

```
'*************************************************************************
'*** Lese Si4734 Hardware-Code/-rev, Firmware, Lib und gebe auf LCD aus ***
'*************************************************************************

Sub Si4735_get_rev()

   '*** POWERUP ***
   I2cstart
   I2cwbyte 34
   I2cwbyte &H01                        'CMD = 0x01 = Powerup

   '*** Set AM-/FM-Mode ***
```

```
If Am_fm_flag = 0 Then
  I2cwbyte &B00010000                    'FM-Modus
Else
  I2cwbyte &B00010001                    'AM-Modus
End If

'*** Set Analog-/Digital Outputs ***
I2cwbyte &B00000101                      'Analog Outputs On, Digital Outputs Off
I2cstop

Waitms 120                               '110ms Wartezeit nach Powerup lt. Datenblatt
'*** GET REVISION ***
I2cstart
I2cwbyte 34
I2cwbyte &H10                             'CMD = 0x10 = Get_Rev.
I2cstop

Waitms 1

I2cstart
I2cwbyte 35                              'Read-Mode
I2crbyte Status , Ack
I2crbyte R1 , Ack
I2crbyte R2 , Ack
I2crbyte R3 , Ack
I2crbyte R4 , Ack
I2crbyte R5 , Ack
I2crbyte R6 , Ack
I2crbyte R7 , Nack
```

```
   I2cstop

   Cls
   Lcd "Si47" ; R1 ; " Rev.:" ; R6
   Lowerline : Lcd "FW:" ; R2 ; "." ; R3 ; " Lib:" ; R7
   Wait 2




End Sub



'****************************
'*** Initialisiere SI4735 ***
'*** FM/AM-Modus starten ****
'****************************


Sub Si4735_init()

   Cls

   '*** POWERDOWN ***
   I2cstart
   I2cwbyte 34
   I2cwbyte &H11                        'CMD 0x11=POWERDOWN
   I2cstop
```

```
Waitms 1

'*** POWERUP ***
I2cstart
I2cwbyte 34
I2cwbyte &H01                          'CMD = 0x01 = Powerup

'*** Set AM-/FM-Mode ***
If Am_fm_flag = 0 Then
  I2cwbyte &B00010000                  'FM-Modus
Else
  I2cwbyte &B00010001                  'AM-Modus
End If

'*** Set Analog-/Digital Outputs ***
I2cwbyte &B00000101                    'Analog Outputs On, Digital Outputs Off
I2cstop

Waitms 120                             '110ms Wartezeit nach Powerup lt. Datenblatt


'*** Lade und setze Properties ***
If Am_fm_flag = 0 Then                 'Property-Indices für Schleife
  N1 = 1 : N2 = 8                      'FM
Else
  N1 = 9 : N2 = 16                     'AM
End If

For N = N1 To N2                       'Read back Properties 1...8
```

```
    Property_adr = Property_adress(n)
    Dat = Property_dat(n)

    If Dat > Property_max(n) Then                   'Falls Property außerhalb Wertebereich...
      Dat_word = Si4735_get_property(property_adr)    'lese eingestellten (Default-) Wert
      Property_dat(n) = Low(dat_word)               'und speichere in Property_dat(N)
    Else
      Call Si4735_set_property(property_adr , &H00 , Dat)      'sonst: Property setzen
    End If

  Next N

  Call Si4735_set_rx_volume                       'Set Volume


  If Am_fm_flag = 0 Then                          'FM-Modus

    Call Si4735_set_property(&H1402 , &H00 , &H05)      'Setze FM_Seek_Freq_Spacing Suchraster-FM auf 50kHz anstelle 100kHz (Default)

    Call Si4735_set_property(&H1108 , &H00 , 20)        'Setze FM_Max_Tune_Error 20kHz anstelle 30kHz (Default)

    Cls : Lcd "       FM" : Lowerline
    Call Lcd_freq(band_start(17) , 2 , 1)
    Locate 2 , 7 : Lcd "-"

    Call Lcd_freq(band_end(17) , 2 , 8)
```

```
    Locate 2 , 14 : Lcd "MHz" : Wait 1
    Cls

    Call Fm_tune_freq                       'Set Frequency

    Call Fm_tune_status_freq_print_lcd      'Tune-Status auf Display

    Call Rds_init                           'RDS initialisieren

  Else                                      'AM-Modus einschalten

    Band_name = Lookupstr(band , Band_names)
    Cls : Lcd Band_name : Lowerline
    Lcd Band_start(band) ; "-" ; Band_end(band) ; " kHz" : Wait 1
    Cls

    Call Am_tune_freq
    Waitms 500

    Call Am_tune_status_freq_print_lcd

  End If

End Sub



'*************************************************************************
'*** Si4735: Statusbyte holen                                        ***
```

```
'*** Bit 7 = CTS = Clear to send next commands                    ***
'*** Bit 6 = ERR = Error                                          ***
'*** Bit 3 = RSQINT = Received Signal Quality measurement has been triggered ***
'*** Bit 2 = RDSINT = RDS-Interrupt                               ***
'*** Bit 0 = STCINT = Seek/Tune Complete                          ***
'*************************************************************************

Sub Si4735_get_status_byte()                    'Abfrage, ob STC-Bit gesetzt (Seek/Tune Complete), also Si4735 bereit zum Befehlsempfanf

   I2cstart
   I2cwbyte 34
   I2cwbyte &H14                                'CMD $0x14 = Si4735_Get_status_byte
   I2cstop

   Waitms 1

   I2cstart
   I2cwbyte 35                                  'Read-Mode
   I2crbyte Status , Nack                       'STATUS-Byte: D7=CTS, D6=ERR, D3=RSQINT, D2=RDSINT, D0=STCINT
   I2cstop

'    If Status.2 = 1 Then Rds_flag = 1 Else Rds_flag = 0

End Sub
```

```
'*****************************************
'*** Si4735: Warte bis Statusbit.0 STC=1 ***
'*** Bit 0 = STCINT = Seek/Tune Complete ***
'*****************************************

Sub Si4735_wait_for_seek_tune_cplt()

   Do
     Call Si4735_get_status_byte
   Loop Until Status.0 = 1 Or S2 = 0 Or S3 = 0 Or S4 = 0 Or S5 = 0 Or Encoder1old <> Encoder1

End Sub



'*****************************************
'*** Si4735: Warte bis Statusbit.7 CTS=1 ***
'*** Bit 0 = CTS = Clear to send nxt cmd ***
'*****************************************

Sub Si4735_wait_for_clear_to_send()

   Do
     Call Si4735_get_status_byte
   Loop Until Status.7 = 1

End Sub
```

```
'*************************
'*** Si4735: Set-Volume ***
'*************************

Sub Si4735_set_rx_volume()

    Call Si4735_set_property(&H4000 , &H00 , Vol)

End Sub




'************************************
'*** Si4735: Properties-Word setzen ***
'************************************

Sub Si4735_set_property(byval Prop_adr As Word , Byval Prop_value_high As Byte , Byval Prop_value_low As Byte )

   Local High_adr As Byte
   Local Low_adr As Byte
   Local High_value As Byte
   Local Low_value As Byte

   Call Si4735_wait_for_clear_to_send()

   High_adr = High(prop_adr)
   Low_adr = Low(prop_adr)
```

```
    I2cstart
    I2cwbyte 34
    I2cwbyte &H12                          'CMD=0x12=Set Property
    I2cwbyte &H00
    I2cwbyte High_adr
    I2cwbyte Low_adr
    I2cwbyte Prop_value_high
    I2cwbyte Prop_value_low
    I2cstop

    Waitms 15                              '10ms t(complete) laut Datenblatt

End Sub

'************************************
'*** Si4735: Properties-Word holen ***
'************************************

Function Si4735_get_property(byval Prop_adr As Word) As Word

    Local High_adr As Byte
    Local Low_adr As Byte
    Local High_value As Byte
    Local Low_value As Byte

    Call Si4735_wait_for_clear_to_send()
```

```
   High_adr = High(prop_adr)
   Low_adr = Low(prop_adr)



   I2cstart
   I2cwbyte 34
   I2cwbyte &H13                          'CMD 0x13=CMD Si4735_get_property
   I2cwbyte &H00
   I2cwbyte High_adr
   I2cwbyte Low_adr
   I2cstop

   Waitms 1

   I2cstart
   I2cwbyte 35                          'Read-Mode
   I2crbyte Status , Ack                   'STATUS-Byte: D7=CTS, D6=ERR, D3=RSQINT, D2=RDSINT,
D0=STCINT
   I2crbyte High_value , Ack                'Always $00
   I2crbyte High_value , Ack                'High und Lowbyte der Property einlesen
   I2crbyte Low_value , Nack
   I2cstop
   Waitms 1

   Si4735_get_property = Makeint(low_value , High_value)

End Function
```

```
'***********************************************
'*** Gebe Property-Wert(n) auf LCD Zeile 2 aus ***
'***********************************************

Sub Si4735_print_property_lcd()                    'Property N auf LCD bringen (2. LCD-Zeile)

   Local Property_string As String * 8

   Property_string = Lookupstr(n , Property_names)

   Home Lower
   Dat = Property_dat(n)
   Lcd "P" ; N ; "  " ; Dat ; "    "                '"P nn Propertywert" in 2. Zeile schreiben
   Locate 2 , 9 : Lcd Property_string

End Sub



'*********************************************
'*** Si4735: Suchschritt AM 1/5/9 kHz setzen ***
'*********************************************

Sub Am_seek_step()

   Call Si4735_set_property(&H3402 , &H00 , Seek_step)

End Sub
```

```
'************************************
'*** Si4735: AM-Frequenz einstellen ***
'************************************

Sub Am_tune_freq()

  If Freq_am > 500 Then                    'Verkürzungsspulen setzen:
    If Freq_am > 2000 Then
      Portb.0 = 1                  'SW
      Portc.3 = 0
    Else
      Portb.0 = 0                  'MW
      Portc.3 = 1
    End If
  Else
    Portb.0 = 0                    'LW
    Portc.3 = 0
  End If

  H = High(freq_am)
  L = Low(freq_am)

  I2cstart
  I2cwbyte 34
  I2cwbyte &H40                        'CMD 0x40 = Tune AM-Frequency
  I2cwbyte &H00
  I2cwbyte H
```

```
   I2cwbyte L
   I2cwbyte &H00                    'Tuning-Cap automatic
   I2cstop

   Waitms 85                        '80ms bis Seet-Tune Complete lt. Datenblatt

   Call Si4735_wait_for_seek_tune_cplt        'Warte bis STC-Bit gesetzt

End Sub



'************************************
'*** Si4735: AM-Suchmodus aufwärts ***
'************************************

Sub Am_seek_freq_up()

   Call Am_stop_seek_tune             'Cancel Seek

   I2cstart
   I2cwbyte 34
   I2cwbyte &H41                    'CMD=0x41
   I2cwbyte &B00001000                'Bit3=Seek Up/Down; Bit2=Wrap/Stop bei Erreichen der
Bandgrenze
   I2cstop

   Waitms 85                        '80ms bis Seet-Tune Complete lt. Datenblatt
```

```
   Call Clear_lcd_lowerline : Lowerline : Lcd "Scanning..."

   Call Si4735_wait_for_seek_tune_cplt          'Warte bis STC-Bit gesetzt

End Sub


'***********************************
'*** Si4735: AM-Suchmodus abwärts ***
'***********************************

Sub Am_seek_freq_down()

   Call Am_stop_seek_tune                  'Cancel Seek

   I2cstart
   I2cwbyte 34
   I2cwbyte &H41                  'CMD=0x41
   I2cwbyte &B00000000                  'Bit3=Seek Up/Down; Bit2=Wrap/Stop bei Erreichen der Bandgrenze
   I2cstop

   Waitms 85                  '80ms bis Seet-Tune Complete lt. Datenblatt

   Call Lcd_scanning

   Call Si4735_wait_for_seek_tune_cplt          'Warte bis STC-Bit gesetzt

End Sub
```

```
'****************************************
'*** Lese AM-Senderspeicher ins EEPROM ***
'****************************************

Sub Read_ammemories_eeprom(stored_memories As Byte , Byval Start_memory As Word)

   Local End_memory As Word

   If Stored_memories >= Start_memory Then

      End_memory = Stored_memories                    'Schleifenzähler-Endwert = Start + Anzahl Speicher -1
'      End_memory = End_memory + Start_memory
'      Decr End_memory

      For I1 = Start_memory To End_memory
         I = I1 * 2
         Readeeprom Freq , I                          'AM-Frequenz ins EEPROM
         Am_memory(i1) = Freq

         I = I1 * 11
         I = I + 191
         Readeeprom Text1 , I
         Am_text(i1) = Text1                          'AM-Sendernamen ins EEPROM
      Next I1
```

```
   End If

End Sub




'*******************************************
'*** Speichere AM-Senderspeicher ins EEPROM ***
'*******************************************

Sub Write_ammemories_eeprom(stored_memories As Byte , Byval Start_memory As Word)

   Local End_memory As Word

   If Stored_memories >= Start_memory Then

      End_memory = Stored_memories                  'Schleifenzähler-Endwert = Start + Anzahl Speicher -1
'     End_memory = End_memory + Start_memory
'     Decr End_memory

      For I1 = Start_memory To End_memory
         I = I1 * 2
         Freq = Am_memory(i1)
         Writeeeprom Freq , I
                                        'AM-Frequenz ins EEPROM
         I = I1 * 11
         I = I + 191
         Text1 = Am_text(i1)
```

```
        Writeeeprom Text1 , I                   'AM-Sendernamen ins EEPROM
      Next I1

   End If

End Sub



'************************************
'*** Si4735: AM-Suchlauf abbrechen ***
'************************************

Sub Am_stop_seek_tune()                         'Stopt aktuellen AM-Suchlauf

   I2cstart
   I2cwbyte 34
   I2cwbyte &H42                        'CMD=0x42
   I2cwbyte &B00000011                      'Bit1=1 Cancel Seek; Bit0=1 Clear Seek/Tune Interrupt Status Bits
   I2cstop

   Waitms 1

   Call Lcd_scanning

   Call Si4735_wait_for_clear_to_send

End Sub
```

```
'************************************
'*** Si4735: FM-Frequenz einstellen ***
'************************************

Sub Fm_tune_freq()

   Call Rds_reset

   Call Fm_stop_seek_tune                    'Cancel Seek

   H = High(freq_fm)
   L = Low(freq_fm)

   I2cstart
   I2cwbyte 34
   I2cwbyte &H20                    'CMD=0x20
   I2cwbyte &H00                    'Always $00
   I2cwbyte H
   I2cwbyte L
   I2cwbyte &H00                    'Antenna-Cap=Automatic
   I2cstop

   Waitms 65                        '60ms bis Seek-Tune-Complete laut Datenblatt

   Call Si4735_wait_for_seek_tune_cplt          'Warte bis STC-Bit gesetzt
```

```
   Cls



End Sub



'************************************
'*** Si4735: FM-Suchmodus aufwärts ***
'************************************

Sub Fm_seek_freq_up()

   Call Fm_stop_seek_tune                    'Cancel Seek

   I2cstart
   I2cwbyte 34
   I2cwbyte &H21                             'CMD=0x21
   I2cwbyte &B00001000                       'Bit3=Seek Up/Down; Bit2=Wrap/Stop bei Erreichen der
Bandgrenze
   I2cstop

   Waitms 65                                 '60ms bis Seet-Tune Complete lt. Datenblatt

   Call Lcd_scanning

   Call Si4735_wait_for_seek_tune_cplt       'Warte bis STC-Bit gesetzt

End Sub
```

```
'***********************************
'*** Si4735: FM-Suchmodus abwärts ***
'***********************************

Sub Fm_seek_freq_down()

   Call Fm_stop_seek_tune                    'Cancel Seek

   I2cstart
   I2cwbyte 34
   I2cwbyte &H21                             'CMD=0x21
   I2cwbyte &B00000000                       'Bit3=Seek Up/Down; Bit2=Wrap/Stop bei Erreichen der
Bandgrenze
   I2cstop

   Waitms 65                                 '60ms bis Seet-Tune Complete lt. Datenblatt

   Call Lcd_scanning

   Call Si4735_wait_for_seek_tune_cplt       'Warte bis STC-Bit gesetzt

End Sub

'***********************************
'*** Si4735: FM-Suchlauf abbrechen ***
```

```
'************************************

Sub Fm_stop_seek_tune()

   Call Rds_reset

   I2cstart
   I2cwbyte 34
   I2cwbyte &H22                        'CMD=0x22
   I2cwbyte &B00000011                      'Bit1=1 Cancel Seek; Bit0=1 Clear Seek/Tune Interrupt Status Bits
   I2cstop

   Waitms 1

   Call Si4735_wait_for_clear_to_send

End Sub



'************************************************************
'*** Si4735: Frequenz/Tuningdetails FM in R1...R7 auslesen ***
'*** Frequenz auf LCD ausgeben, falls gültig            ***
'************************************************************

Sub Fm_tune_status_freq_print_lcd()                 'Frequenz holen und ausgeben

   Call Si4735_get_status_byte                  '
```

```
   I2cstart

   I2cwbyte 34                          'Adresse Si4736 für Write

   I2cwbyte &H22                         'CMD = 0x22 = FM_Tune_Status

'   I2cwbyte &B00000000                              'D1=1 Cancel Seek; D0=1 Clear Seek/Tune Complete Interrupt Status Indicator

   I2cwbyte &B00000011                    'Original-Version --> Cancel Seek, Clear Seek-Tune-Complete Interrupt Bit

   I2cstop


   Waitms 1


   I2cstart

   I2cwbyte 35                          'Adresse Si4735 für Read

   I2crbyte Status , Ack                  'Statusbyte (Bit 7=CTS)

   I2crbyte R1 , Ack                     'Bit7=kompletter Suchlauf (Bandlimit) nach FM_Seek; Valid_channel=Bit 0

   I2crbyte R2 , Ack                     'Frequenz

   I2crbyte R3 , Ack                     'Frequenz

   I2crbyte R4 , Ack                     'RSSI

   I2crbyte R5 , Ack                     'S/N

   I2crbyte R6 , Ack

   I2crbyte R7 , Nack                     'Tuning-Capacitor

   I2cstop


   Bltf = R1.7                          'Statusbit für Erreichen der Bandgrenze bzw. Wraparound


   Freq = 256 * R2

   Freq = Freq + R3
```

```
   Rssi = R4

   Snr = R5

   Valid = R1.0

   If Valid = 1 Then                    'Falls Frequenz Valid --> ausgelesene Frequenz in Freq_fm
übernehmen; Daten in EEPROM sichern
      Freq_fm = Freq
      Call Reset_encoder
      Call Eeprom_save
   End If

'    If Radiotext_ready = 0 Then                    'falls keine RDS-Anzeige
      Call Lcd_freq(freq_fm , 1 , 1)          '--> Frequenz-Anzeige
'    End If

'    Pwm1a = Rssi * 12                    'RSSI als PWM ausgeben

   Waitms 1

   Call Si4735_wait_for_clear_to_send

End Sub

'****************************************
'*** Drucke FM_Freq in LCDPosition Y,X ***
```

```
'****************************************

Sub Lcd_freq(byval Fm_freq As Word , Byval Y_pos As Byte , Byval X_pos As Byte)

  Call Make_freq_string(fm_freq)                    'erzeuge Frequenz-String
  Locate Y_pos , X_pos                              '--> Frequenz in MHz + 2 Nachkommastellen auf Display
  Lcd Freq_string

End Sub



'****************************************
'*** Erzeuge formatierten Frequenz-String ***
'****************************************

Sub Make_freq_string(byval Af_freq As Word)

Local Freq_int As Word
Local Freq_remainder As Word
Local Short_string As String * 2

  If Af_freq > 6000 Then                            'FM-Frequenz in 10 kHz-Einheiten

    Freq_int = Af_freq / 100 : Freq_remainder = Af_freq Mod 100       'formatiere auf " 88.56" für MHz-Ausgabe
    Freq_string = Str(freq_int) : Short_string = Str(freq_remainder)

    If Freq_int < 100 Then
      Freq_string = " " + Freq_string
```

```
    End If

    Freq_string = Freq_string + "."

    If Freq_remainder < 10 Then
      Freq_string = Freq_string + "0"
    End If

    Freq_string = Freq_string + Short_string

  Else                                  'AM-Frequenz in  kHz

    Freq_string = Str(af_freq)

  End If

End Sub




'************************************************
'*** Drucke "Scanning..." in untere LCD-Zeile ***
'*** Lösche Mxx Feld                  ***
'************************************************

Sub Lcd_scanning()

   Cls
```

```
'    Call Clear_lcd_lowerline
   Lowerline : Lcd "Scanning..."
'    Locate 1 , 13 : Lcd "    "                              'lösche letzte 4 Zeichen rechts oben

End Sub


'************************************
'** RSSI und SNR holen und ausgeben ***
'************************************


Sub Fm_rsq_status()

   I2cstart
   I2cwbyte 34
   I2cwbyte &H23                          'CMD 0x23=FM_RSQ_STATUS
   I2cwbyte &B00000000                        'Bit0=0 --> Interrupt Status Preserved
   I2cstop

   Waitms 1

   I2cstart
   I2cwbyte 35
   I2crbyte Status , Ack                    'Bit 7=CTS;
   I2crbyte R1 , Ack                      'div. Statusbits
   I2crbyte R2 , Ack                      'div. Statusbits; Bit0=Fm_freq_valid_flag Channel
   I2crbyte R3 , Ack                      'div. Statusbits; Bit7=Pilotton
   I2crbyte R4 , Ack                      'RSSI:0...127dBuV
```

```
   I2crbyte R5 , Ack                        'SNR:=9...127dB
   I2crbyte R6 , Ack                        'Stereo-Block_errors_block_nd 0...100%
   I2crbyte R7 , Nack                        'Signed Frequency Offset
   I2cstop

   Rssi = R4
   Snr = R5

'(
   If Status.2 = 1 Then
     Rds_flag = 1                                  'RDS_INT Flag setzen
   Else
     Rds_flag = 0
   End If
')

   Incr Lcdrefresh                          'nur bei jedem n. Schleifendurchlauf

   If Lcdrefresh > 4 Then

     Lcdrefresh = 0

'     If Radiotext_present = 0 Then                              'falls keine Radiotext-Anzeige
     If Radiotext_ready = 0 Then                  'falls keine Radiotext-Anzeige

         Call Clear_lcd_upperline
         Call Lcd_freq(freq_fm , 1 , 1)           'Frequenz auf Display
         Locate 1 , 8
         Lcd Rssi ; " "                       'RSSI und
```

```
      Locate 1 , 11
      Lcd Snr ; " "                        'SNR auf Display
      If Memory_aktiv_flag = 1 Then
        Locate 1 , 14                      '"Mnn" in Zeile 1, Spalte 14 schreiben
        Lcd "M" ; Fm_memorypos
      End If

    End If

    If Property_aktiv = 0 Then

      Locate 2 , 15                        'STEREO-Anzeige-Symbol

      If R3.7 = 1 Then Lcd Chr(7) Else Lcd " "       'Pilotton vorhanden -->Sonderzeichen für ST=Stereo

      Locate 2 , 16                        'RDS/RADIOTEXT-Anzeige-Symbol

'     If Radiotext_present = 1 Or Radiotext_ready = 1 Then
      If Radiotext_present = 1 Then
        Lcd Chr(0);                        'Sonderzeichen "RT" für Radiotext
      Elseif Rds_flag = 1 Then                   '"R" für nur RDS (ohne Radiotext)
        Lcd "R";
      Else
        Lcd " ":
      End If

    End If

  End If
```

```
   Stereo_signal = R3 And &B01111111                  'Bit 0...6 von R3 = Stereo-Signal in %
   Freq_offset = R7                                   'Frequency-Offset in kHZ [-128...+128kHz]

   Select Case Pwm_output
        Case 0 : Pwm1a = Stereo_signal * 8             'PWM1A als Stereo-Signal setzen 0...100% = 0...1023
        Case 1 : Pwm1a = Freq_offset * 4               'PWM1A als Freq_Offset setzen +/-128kHz = 0...1023
        Case 2 : Pwm1a = Rssi * 12                     'Signalstärke an PWM-Ausgang 0..85dBµV = 0...1023
        Case 3 : Pwm1a = Snr * 12                      'SNR 0...85dB = 0...1023
   End Select

   Waitms 1

   Call Si4735_wait_for_clear_to_send

End Sub



'*************************************************************
'*** Si4735: Frequenz/Tuningdetails AM in R1...R7 auslesen ***
'*** Frequenz auf LCD ausgeben, falls gültig              ***
'*************************************************************

Sub Am_tune_status_freq_print_lcd()

   I2cstart
   I2cwbyte 34
   I2cwbyte &H42                                      'CMD=0x42
```

```
'    I2cwbyte &B00000000                    'Bit1=0 Continue Seek; Bit0=Clear Seek/Tune Interrupt Status Bits
    I2cwbyte &B00000011                     'Original-Version --> Cancel Seek, Clear Seek-Tune-Complete Interrupt Bit
    I2cstop

    Waitms 1

    I2cstart
    I2cwbyte 35                             'Adresse Si4735 für Read
    I2crbyte Status , Ack                   'Statusbyte (Bit 7=CTS)
    I2crbyte R1 , Ack                       'Bit7=kompletter Suchlauf (Bandlimit) nach AM_Seek; Fm_freq_valid_flag=Bit 0
    I2crbyte R2 , Ack                       'Frequenz
    I2crbyte R3 , Ack                       'Frequenz
    I2crbyte R4 , Ack                       'RSSI
    I2crbyte R5 , Ack                       'S/N
    I2crbyte R6 , Ack                       'Tuning-Capacitor
    I2crbyte R7 , Nack                      'Tuning-Capacitor
    I2cstop

    Bltf = R1.7                             'Statusbit für Erreichen der Bandgrenze

    Freq = 256 * R2
    Freq = Freq + R3

    Rssi = R4

    Snr = R5
```

```
   C = 256 * R6

   C = C + R7

   C = C * 95

   C = C / 1000

   C = C + 7


   Valid = R1.0                          'R1.0 = Valid AM-Frequency


   Home Upper


   Lcd Freq

   If Freq < 10000 Then Lcd " "

   If Freq < 1000 Then Lcd "  "

   Locate 1 , 13

   Lcd C ; " "


   If Valid = 1 Then                     'Falls Frequenz Valid --> Frequenz übernehmen Daten in EEPROM
sichern

      Freq_am = Freq

      Call Eeprom_save

   End If


   Pwm1a = Rssi * 12                     'Signalstärke an PWM-Ausgang übergeben


   Call Si4735_wait_for_clear_to_send


End Sub
```

```
'************************************
'** RSSI und SNR holen und ausgeben ***
'************************************

Sub Am_rsq_status()

   I2cstart
   I2cwbyte 34
   I2cwbyte &H43                        'CMD 0x43
   I2cwbyte &B00000000                      'Bit0=0 Interrupt Status preserved
   I2cstop

   Waitms 1

   I2cstart
   I2cwbyte 35
   I2crbyte Status , Ack
   I2crbyte R1 , Ack                    'div. Statusbits
   I2crbyte R2 , Ack                    'div. Statusbits; Bit0=Fm_freq_valid_flag Channel
   I2crbyte R3 , Ack
   I2crbyte R4 , Ack                    'RSSI:0...127dBuV
   I2crbyte R5 , Nack                    'SNR:=9...127dB
   I2cstop

   Rssi = R4
   Snr = R5
'  Incr Lcdrefresh
```

```
'   If Lcdrefresh > 5 Then
'     Lcdrefresh = 0
      Locate 1 , 7
      Lcd Rssi ; " "
      Locate 1 , 10
      Lcd Snr ; " "
'   End If
    If Am_textaktiv = 1 Then                    'Im Texteingabemodus -->
      I = Am_textpos + 6
      Locate 2 , I
    End If

    Pwm1a = Rssi * 12
    Waitms 1
    Call Si4735_wait_for_clear_to_send

End Sub




'************************************************
'*** Stelle AM-Speicher ein und bringe auf LCD ***
'************************************************

Sub Call_am_memory(am_memory_number As Byte)

    Home Lower                                  '"Mnn" in 2. Zeile schreiben
    Lcd "M" ; Am_memorypos ; "  "
```

```
   Freq_am = Am_memory(am_memory_number)              'Frequenz aus Speicher-Array laden
   Call Am_tune_freq

   Call Am_tune_status_freq_print_lcd

   Locate 2 , 7
   Text1 = Am_text(am_memory_number)                  'Sendername in 2. Zeile ab Spalte 7 schreiben
   Lcd Text1

   Call Reset_encoder

End Sub



'************************************************
'*** Stelle FM-Speicher ein und bringe auf LCD ***
'************************************************

Sub Call_fm_memory(fm_memory_number As Byte)

   Freq_fm = Fm_memory(fm_memory_number)
   Call Fm_tune_freq

   Call Fm_tune_status_freq_print_lcd
```

```
   Locate 1 , 14                    '"Mnn" in Zeile 1, Spalte 14 schreiben
   Lcd "M" ; Fm_memorypos ; " "

   Call Reset_encoder

End Sub


'*********************************
'*** Si4735: RDS initialisieren ***
'*********************************

Sub Rds_init()

   'RDS_INT wird gesetzt, wenn RDS-Sync = RDS-Signal vorhanden

   Call Si4735_set_property(&H1500 , &H00 , &B00000100)    'Property=RDS_Int_Source; RDSINT-Set bei... Bit2=RDS-SYNC, Bit1=RDS_SYNC_Lost, Bit0=RDS_Recv

   Call Si4735_set_property(&H1501 , &H00 , 14)          'Property=RDS_Int_FIFO_Count; DAT=14=min. Number of received Groups before RDSRECV-Bit is set

   Select Case Rds_block_errors
         Case 0 : Call Si4735_set_property(&H1502 , &B00000000 , 1)      'mur 1-2 korrigierbare Blockfehler zulassen
         Case 1 : Call Si4735_set_property(&H1502 , &B01010101 , 1)
         Case 2 : Call Si4735_set_property(&H1502 , &B10101010 , 1)
         Case 3 : Call Si4735_set_property(&H1502 , &B11111111 , 1)
   End Select
```

```
Call Rds_reset()


End Sub



'*****************************************************************
'*** Si4735: RDS-Status + RDS-Daten in Status, R1...R12 holen ***
'*****************************************************************


Sub Rds_get_status_and_data()


  I2cstart
  I2cwbyte 34                          'Adresse Si4735 für Write=34
  I2cwbyte &H24                         'CMD $0x24 zur Adressierung RDS-Status-/Daten
'  I2cwbyte &B00000000                                'Preserve RDS-Interrupt-Flag
  I2cwbyte &B00000001                      'Preserve RDS-Interrupt-Flag
   I2cstop


  Waitms 1


  I2cstart
  I2cwbyte 35                          'Adresse Si4735 für Read=35
  I2crbyte Status , Ack                    'Statusbyte einlesen:B7=CTS-Flag, B6=ERR-Flag, B3=RSQINT-Flag,
B2=RDSINT-Flag, B0=STCINT-Flag                                    '


  I2crbyte R1 , Ack                        'Statusbyte einlesen:B2=RDS_SYNC_FOUND-Flag,
B1=RDS_SYNC_LOST-Flag, B0=RDS_RECV-Flag
  I2crbyte R2 , Ack                        'Statusbyte einlesen:B2=GRP_LOST-Flag, B0=RDS_SYNC-Flag
```

```
  I2crbyte R3 , Ack                    'RDS_FIFO_USED = Anzahl der verbleibenden Gruppen im RDS-
FIFO-Speicher

  I2crbyte R4 , Ack                    'RDS-Gruppen in R4...R11 (8 Bytes)
  I2crbyte R5 , Ack
  I2crbyte R6 , Ack
  I2crbyte R7 , Ack
  I2crbyte R8 , Ack
  I2crbyte R9 , Ack
  I2crbyte R10 , Ack
  I2crbyte R11 , Ack
  I2crbyte R12 , Nack                   'je 2 Bit für Anzahl der korrigierten RDS-Fehler in Gruppe A,B,C,D
  I2cstop


  Waitms 1

  Call Si4735_wait_for_clear_to_send

End Sub


'************************************************
'*** RDS-Dekodierung, LCD-Anzeige, RS-232-Anzeige ***
'************************************************

Sub Rds_decoding()

    Do                                 'Solange je eine Gruppe auf FIFO holen, bis FIFO leer
```

```
Call Si4735_get_status_byte                    'Statusbit 7 holen

If Status.7 = 1 Then                           'Falls CTS=1 (Si4735 bereit)...

 Call Rds_get_status_and_data                  'hole RDS-Daten

 If R2.0 = 1 Then Rds_flag = 1 Else Rds_flag = 0    'RDS-Flag = aktueller RDS-Sync-Status

 If R3 > 0 Then                                'FIFO-Speicher nicht leer (>0)

   Block_errors_block_a = R12 And &B11000000       'Anzahl RDS-Fehler in Block A isolieren
   Block_errors_block_b = R12 And &B00110000       'Anzahl RDS-Fehler in Block B isolieren
   Block_errors_block_c = R12 And &B00001100       'Anzahl RDS-Fehler in Block C isolieren
   Block_errors_block_d = R12 And &B00000011       'Anzahl RDS-Fehler in Block D isolieren

   Service = R6 And &B11111000                  'Gruppennummer = Bit 7...4 + Typ = Bit 3 aus R6 isolieren



   '*** GRUPPE 0A/0B (PI/TA/MS_Flag/DI/AF/PS=Sendername) ***

   If Service = &B00000000 Or Service = &B00001000 Then       'Gruppe 0A/0B (Bit7..3/R6=0) -->
Sendername

     If R7.4 = 1 Then Ta = 1 Else Ta = 0         'Travel-Announcement-Flag
     If R7.3 = 1 Then Ms_flag = 1 Else Ms_flag = 0       'Music-Speech-Flag
```

```
    Pos = R7 And &B00000011                 'Adresse für DI und PS-Namenszeichen isolieren Bit 0,1 in
R7


    Byte1 = Pos Xor &B00000011              'Invertiere Adresse für DI-Adressierung
    Incr Byte1                              '+1, da DI-Tabelle 1-based


    Incr Di_counter : If Di_counter = 0 Then Di_counter = 4     'DI-Counter>=4  als Zeichen für
vollständig dekodierte DI


    If R7.2 = 1 Then                        'DI-Flag = Bit2 in R7
      Di(byte1) = 1
    Else
      Di(byte1) = 0
    End If


    'PS-Name
    Set Ps_counter.pos                      'Setze n=pos. Bit in PS-Counter


    Pos = Pos * 2                           'Programmname dekodieren und in Ps_name-String bringen
    Pos = Pos + 1
    Call Rds_char_code(r10)                 'RDS ASCII-Code in EADOGM-Code umrechnen
    Ch = Chr(r10)                           '2 Zeichen aus R10 und R11 (Block D)
    Mid(ps_name , Pos , 1 ) = Ch            'in Ps_name-String
    Call Rds_char_code(r11)                 'RDS ASCII-Code in EADOGM-Code umrechnen
    Ch = Chr(r11)
    Incr Pos
    Mid(ps_name , Pos , 1 ) = Ch


    If Ps_counter = &B00001111 Then         'Ausgabe auf LCD, wenn alle 4*2 Zeichensätzen
durchlaufen
```

```
    Home Lower
    Lcd Ps_name
    Ps_counter = 0                  'Setze PS-Counter zurück
    Ps_flag = 1
    Ps_name_last = Ps_name
  End If


  'AF in Block C (R8,R9) nur in Gruppe 0A

  If Service = &B00000000 Then          'Alternate Frequencies nur in Gruppe 0A!

    'First 'AF-Byte

    Lw_mw_af = 0                    'LW/MW-Flag zurücksetzen

    If R8 = 224 Then                'keine AF vorhanden

      Af_flag = 0                   '..> AF-Flag = 0

    Elseif R8 > 224 And R8 < 250 And Af_flag <> 2 Then      'Startbyte: = n weitere Alternate
Frequencies vorhanden?

      Af_flag = 1                   'AF-Flag auf Empfang setzen
      Af_number = 0                 'laufender AF_Index
      Af_max = R8 - 224             'Anzahl AFs = Start der AF-Liste

    Elseif R8 > 0 And R8 < 205 And Af_flag = 1 Then      'Code zwischen 1...204 => gültige  AF
```

```
            If Af_number < 26 Then Incr Af_number  'Zähler nur inkrementieren,solange <= 25

'             Word1 = Af(af_number)
            Call Rds_calc_af(r8 , Word1)          'Dekodiere 1 AF
            Af(af_number) = Word1

            If Af_number >= Af_max Then Af_flag = 2       'Setze AF-Flag auf voll, falls AF=AF_max

          Elseif R8 = 250 Then                  'LW/MW-Frequenz folgt

            Lw_mw_af = 1                        '--> Flag für Block D setzen, daß Frequenz = LM/AM-Frequenz ist

          End If

          'Second AF-Byte

          If Af_number < 26 Then Incr Af_number     'Zähler nur inkrementieren,solange <= 25

'             Word1 = Af(af_number)
          Call Rds_calc_af(r9 , Word1)            '2. AF dekodieren
          Af(af_number) = Word1

          If Af_number >= Af_max Then Af_flag = 2   'Setze AF-Flag auf voll, falls AF=AF_max

        End If

      End If
```

```
'*** GRUPPE 1A/1B (PI, Slow-labeling-codes etc.) ***

If Service = &B00010000 Or Service = &B00011000 Then       'Gruppe 1A/1B (Bit7..3/R6=0)

  Pin_code = Makeint(r11 , R10)

  If Service = &B00010000 Then              'Gruppe 1A --> div. Info-Bits
    Word1 = Makeint(r9 , R8)
    Word1 = Word1 And &B0000111111111111
    If R8.7 = 1 Then La = 1 Else La = 0      'Linkage actuator
    Vc = R8 And &B01110000                   'Variant-Code isolieren
    Shift Vc , Right , 4
    If Vc = 0 Then                       'ECC
      Ecc = R9
    Elseif Vc = 1 Then                   'TMC-Identification
      Tmc_ident = Word1
    Elseif Vc = 3 Then                   'Language Code
      Lc = Word1
    Elseif Vc = 7 Then                   'Emergency warning system
      Ews = Word1
    End If
  End If

End If
```

```
'*** GRUPPE 2A/2B (Radiotext) ***

If Service = &B00100000 Or Service = &B00101000 And Radiotext_on = 1 Then 'Gruppe 2A/2B = Radiotext1 (bis zu 64/32 Zeichen!)

Radiotext_present = 1

Radiotext_ab_flag = R7 And &B00010000 'A/B-Toggle-Flag für neuen Text isolieren = Bit 4, Block B, R7

If Radiotext_first_aqu = 0 Then '1. Durchlauf?

Radiotext_ab_flag_old = Radiotext_ab_flag Xor &B00010000 'setzt A/B-Flag künstlich auf aktuelles Toggeln

Radiotext_first_aqu = 1

End If

If Radiotext_ab_flag <> Radiotext_ab_flag_old Then 'A/B Flag getoggelt? --> neue Nachricht beginnt

Radiotext_ab_flag_old = Radiotext_ab_flag

Radiotext_start_decode = 1 '-->setze Flag auf neue Dekodierung

End If

Segment = R7 And &B00001111 'Zeichenposition "POS" = 4*(Bit0...3) von R7

If Radiotext_start_decode = 1 And Segment = 0 Then 'Zeichenanfang gefunden?

Radiotext_start_decode = 2 'Flag auf Beginn Dekodierung setzen

Char_counter = 0

End If
```

```
If Radiotext_start_decode = 2 Then       'Bedingung zum Erstellen des RT-Strings erfüllt

  If Service = &B00100000 Then          'Gruppe 02A

    Position = Segment
    Shift Position , Left , 2           'Position=Segment*4

    If R8 = 13 And Radiotext_start_decode = 2 Then      '=RETURN?
      Call Radiotext_decoded            'Aussprung in RT-Ready
    Else
      Incr Position : Incr Char_counter      'erhöhe Positions- und Gesamt-Zeichenzähler
      Call Rds_char_code(r8)            'RDS ASCII-Code in EADOGM-Code umrechnen
      Mid(radiotext_new , Position , 1) = Chr(r8)      '1. Zeichen in Radiotext-String an richtige
Position schreiben

    If R9 = 13 And Radiotext_start_decode = 2 Then
      Call Radiotext_decoded
    Else
      Incr Position : Incr Char_counter      'erhöhe Positions- und Gesamt-Zeichenzähler
      Call Rds_char_code(r9)       'RDS ASCII-Code in EADOGM-Code umrechnen
      Mid(radiotext_new , Position , 1) = Chr(r9)      '1. Zeichen in Radiotext-String an richtige
Position schreiben

    If R10 = 13 And Radiotext_start_decode = 2 Then
      Call Radiotext_decoded
    Else
      Incr Position : Incr Char_counter
      Call Rds_char_code(r10)    'RDS ASCII-Code in EADOGM-Code umrechnen
```

```
                Mid(radiotext_new , Position , 1) = Chr(r10)       '1. Zeichen in Radiotext-String an richtige
Position schreiben


                If R11 = 13 And Radiotext_start_decode = 2 Then

                  Call Radiotext_decoded

                Else

                  Incr Position : Incr Char_counter

                  Call Rds_char_code(r11) 'RDS ASCII-Code in EADOGM-Code umrechnen

                  Mid(radiotext_new , Position , 1) = Chr(r11)       '1. Zeichen in Radiotext-String an
richtige Position schreiben

                End If

              End If

            End If

          End If


        Else                        'Gruppe 02B:


          Position = Segment

          Shift Position , Left , 1          'Position=Segment*2


          If R10 = 13 And Radiotext_start_decode = 2 Then

            Call Radiotext_decoded

          Else

            Incr Position : Incr Char_counter       'erhöhe Positions- und Gesamt-Zeichenzähler

            Call Rds_char_code(r10)         'RDS ASCII-Code in EADOGM-Code umrechnen

            Mid(radiotext_new , Position , 1) = Chr(r10)       '1. Zeichen in Radiotext-String an richtige
Position schreiben


            If R11 = 13 And Radiotext_start_decode = 2 Then

              Call Radiotext_decoded
```

```
                Else
                  Incr Position : Incr Char_counter      'erhöhe Positions- und Gesamt-Zeichenzähler
                  Call Rds_char_code(r11)      'RDS ASCII-Code in EADOGM-Code umrechnen
                  Mid(radiotext_new , Position , 1) = Chr(r11)      '1. Zeichen in Radiotext-String an richtige Position schreiben
                End If
              End If

            End If

            If Service = &B00100000 And Position = 64 And Radiotext_start_decode = 2 Then Call Radiotext_decoded      'nach 64/32 Zeichen automatisch CR=1 setzen
            If Service = &B00101000 And Position = 32 And Radiotext_start_decode = 2 Then Call Radiotext_decoded      'Gruppe 02B

          End If

        End If



        '*** GRUPPE 4A (UTC/MJD) ***

        If Service = &B01000000 Then                'Gruppe 4A --> UTC-Zeit/Datum

          Byte1 = R7 And &B00000011                 'MJD berechnen
          Mjd = Byte1 : Shift Mjd , Left , 15
          Word1 = Makeint(r9 , R8)
          Shift Word1 , Right , 1
          Long1 = Word1
```

```
Mjd = Mjd + Long1


Utc_offset = R11 And &B00111111            'UTC-Offset berechnen


Lo = R11                           'Minuten berechnen
Shift Lo , Right , 6
Hi = R10 And &B00001111
Shift Hi , Left , 2
Utc_min = Hi + Lo
Minuten = Utc_min


Lo = R10                           'Stunden berechnen
Shift Lo , Right , 4
Hi = R9 And &B00000001
Shift Hi , Left , 4
Utc_hour = Hi + Lo
Stunden = Utc_hour


If R11.0 = 1 Then                      '1/2-Stunden Addition
  If R11.5 = 1 Then
    Minuten = Minuten - 30
    If Minuten < 0 Then
      Minuten = Minuten + 60
      Stunden = Stunden - 1
    End If
  Else
    Minuten = Minuten + 30
    If Minuten > 60 Then
      Minuten = Minuten - 60
```

```
        Stunden = Stunden + 1
      End If
    End If
End If

Shift Utc_offset , Right , 1              'von 1/2-Stunden-Einheiten auf volle Stunden umrechnen
If R11.5 = 1 Then
  Stunden = Stunden - Utc_offset
Else
  Stunden = Stunden + Utc_offset
End If

If Stunden > 23 Then
  Stunden = Stunden - 24
  Decr Mjd
End If

If Stunden < 0 Then
  Stunden = Stunden + 24
  Incr Mjd
End If

Locate 2 , 10                          'Cursor auf Zeile 2, 10. Spalte setzen
Tx = Str(stunden)
If Stunden < 10 Then Tx = "0" + Tx
Lcd Tx ; ":";                          'Stunden ":" Minuten auf Display ausgeben
Tx = Str(minuten)
If Minuten < 10 Then Tx = "0" + Tx
Lcd Tx
```

```
        End If



'(
        '*** GRUPPE 9A (EWS) ***

        If Service = &B10010000 Then                    'Gruppe 10A --> Erweiterung des PTY-Typs

          If Ews <> &HFFFF Then                         'EWS-Daten isolieren
            Ews0 = R7 And &B00011111
            Ews1 = R8 : Ews2 = R9 : Ews3 = R10 : Ews4 = R11
          End If

        End If
')



        '*** GRUPPE 10A (PTYN) ***

        If Service = &B10100000 Then            'Gruppe 10A --> Erweiterung des PTY-Typs

          If Pty_name_flag < 2 Then             'Solange Schleife noch nicht 2x durchlaufen ist
            Incr Pty_name_flag                  'Setze Flag auf erkanntes PTYN

            Byte1 = R7 And &B00000001           'Isoliere Block C, Bit0 als HI/LO Schalter für Position in 8-
Zeichen-String
            Shift Byte1 , Left , 3              '*4
```

```
    Call Rds_char_code(r8)              'RDS ASCII-Code in EADOGM-Code umrechnen
    Ch = Chr(r8) : Pos = 1 + Byte1
    Mid(pty_name , Pos , 1 ) = Ch
    Call Rds_char_code(r9)              'RDS ASCII-Code in EADOGM-Code umrechnen
    Ch = Chr(r9) : Pos = 2 + Byte1
    Mid(pty_name , Pos , 1 ) = Ch
    Call Rds_char_code(r10)             'RDS ASCII-Code in EADOGM-Code umrechnen
    Ch = Chr(r10) : Pos = 3 + Byte1
    Mid(pty_name , Pos , 1 ) = Ch
    Call Rds_char_code(r11)             'RDS ASCII-Code in EADOGM-Code umrechnen
    Ch = Chr(r11) : Pos = 4 + Byte1
    Mid(pty_name , Pos , 1 ) = Ch
  End If

End If



'*** GRUPPE 14A (EON) ***

If Service = &B11100000 Then            'Gruppe 14A --> EON-Info

  Tp_eon = R7.4                 'TP_EON
  Pic1_eon = R10                'Program Identification Code in Block D
  Pic2_eon = R11

  Vc_counter_eon = R7 And &B00001111        'Variant-Code = Adressierung Datenbytes isolieren

  If Eon_flag = &HFF Then Eon_flag = 0      'Start-Flag für EON-Daten vorhanden setzen
```

```
If Eon_flag = 0 And Vc_counter_eon = 0 Then Eon_flag = 1       'EON-Dekodierung starten

If Eon_flag = 1 Then

 'PS (Sendername) dekodieren

  If Vc_counter_eon < 4 Then

    Pos = Vc_counter_eon * 2
    Incr Pos
    Call Rds_char_code(r8)              'RDS ASCII-Code in EADOGM-Code umrechnen
    Ch = Chr(r8)                        '2 Zeichen aus R8 und R9 (Block C)
    Mid(ps_eon , Pos , 1 ) = Ch         'in Ps_name-String
    Incr Pos
    Call Rds_char_code(r9)              'RDS ASCII-Code in EADOGM-Code umrechnen
    Ch = Chr(r9)
    Mid(ps_eon , Pos , 1 ) = Ch


  'AF und Tuning-Freq./Mapped Freq.

  Elseif Vc_counter_eon < 10 Then

    Pos = Vc_counter_eon - 4            'Index = 2*(VC-4)+1
    Shift Pos , Left , 1
    Incr Pos

    Lw_mw_af = 0                        'Setze LW/MW-Flag auf Null
```

```
  If R8 > 0 And R8 < 205 Then          'Code zwischen 1...204 => gültige  AF

    Call Rds_calc_af(r8 , Word1)
    Af_eon(pos) = Word1

  Elseif R8 = 250 Then                 'LW/MW-Frequenz folgt
    Lw_mw_af = 1
    Af_eon(pos) = 0                    '--> Flag setzen, daß nächste Frequenz = LM/AM-Frequenz ist

  Else
    Lw_mw_af = 0
    Af_eon(pos) = 0
  End If

  Incr Pos

  If R9 > 0 And R9 < 205 Then          'Code zwischen 1...204 => gültige  AF

    Call Rds_calc_af(r9 , Word1)
    Af_eon(pos) = Word1

  End If

'Linkage Information

Elseif Vc_counter_eon = 12 Then

  Link_info = Makeint(r9 , R8)
```

```
    'PTY_EON, TA

    Elseif Vc_counter_eon = 13 Then

      Pty_eon = R8 And &B11110000
      Swap Pty_eon
      Ta_eon = R9.0

    'PIN_EON

    Elseif Vc_counter_eon = 14 Then

      Pin_code_eon = Makeint(r9 , R8)

      Eon_flag = 2                    'Flag auf voll dekodiert setzen

    End If

  End If

End If

'*** GRUPPE 14B (EON) ***

If Service = &B11101000 Then              'Gruppe 14B --> EON-Info

  Ta_eon = R7.3                   'TA_EON
  Tp_eon = R7.4                   'TP_EON
  Pic1_eon = R10                   'Program Identification Code in Block D
```

```
      Pic2_eon = R11

    End If

    If Rds_over_rs232_control.0 = 1 Then          'RDS-Infosegment 1

      Print

      'aktuelle Gruppennummer und Typ anzeigen
      Byte1 = Service : Swap Byte1 : Byte1 = Byte1 And &B00001111
      Print "Gruppennummer: " ; Byte1 ; : If Service.3 = 0 Then Print "A" Else Print "B"

      'Blöcke anzeigen
      Print "BlockA: " ; Bin(r4) ; " " ; Bin(r5)
      Print "BlockB: " ; Bin(r6) ; " " ; Bin(r7)
      Print "BlockC: " ; Bin(r8) ; " " ; Bin(r9)
      Print "BlockD: " ; Bin(r10) ; " " ; Bin(r11)
      Print "Blockerrors: " ; Bin(r12)

    End If

  End If

End If
```

```
    '*** Bei Tastendruck, Encoder, oder RS-232-Input --> Ende Radiotext-Dekodierung ***
'(
    If S2 = 0 Or S3 = 0 Or S4 = 0 Or S5 = 0 Or Encoder1 <> Encoder1old Then
      Call Rds_reset
      Exit Do
    End If

    If Iscarwaiting(#1) = 1 Then Exit Do
')
  Loop Until R3 = 1 Or R3 = 0                    'Schleife, bis FIFO-Speicher leer


  '*** div. RDS-Infos auf RS-232 ausgeben ***


  Pi = Makeint(r5 , R4)                          'Isoliere PI-Code aus BlockA = R4 + R5

  If Rds_over_rs232_control.1 = 1 And Pi <> &HFFF Then   'RDS-Infosegment 2

    Print
    'PI-Code
    Print "PI = $" ; Hex(pi) ; " = %" ; Bin(r4) ; " " ; Bin(r5)

    'Travel-Program, Travel-Announcement
    Print "TP = ";
    If R6.2 = 1 Then
      Tp = 1
      Print "1 :" ;                             'Verkehrsfunksender
```

```
Else
  Tp = 0
  Print "0 : No ";
End If
Print "Travel Program"

Print "TA = " ; Ta ; " : " ;
If Ta = 0 Then Print "No ";
Print "Travel-Announcement" ;
If Ta = 1 And Tp = 0 Then Print " on EON" Else Print       'TA auf anderem Sender via EON

'Music/Speech
Print "Music or Speech: ";
If Ms_flag = 1 Then                         'MS
  Print "Music"
Elseif Ms_flag = 0 Then
  Print "Speech"
End If

'PTY (Europa)
Pty = R6 And &B00000011                     'PTY-Bits isolieren
Shift Pty , Left , 3
Ptyn = R7 And &B11100000
Shift Ptyn , Right , 5
Pty = Pty Or Ptyn
Print "Program-Type: " ; Pty ; " = ";

'PTY im Klartext ausgeben (Europäische Kodierung)
'empfohlene Ausgabe für 16-Zeichen Anzeige
```

```
  Text1 = Lookupstr(pty , Pty_codes)
  Print Text1 ; " ";

  'PTY_Name
  If Pty_name_flag > 1 Then                    'Klartext-Ergänzung des PTY
    Print Pty_name
    Pty_name_flag = 0                          'Flag zurücksetzen
  Else
    Print
  End If


  'Decoder-Information

  If Di_counter > 4 Then
    Print "Decoder-Information: " ;              'DI
    If Di(1) > 0 Then Print "Stereo," ; Else Print "Mono,";
    If Di(2) > 0 Then Print " Binaural Transmission,";
    If Di(3) > 0 Then Print " C" ; Else Print " Unc";
    Print "ompressed Transmission";
    If Di(4) > 0 Then Print ", Dynamic PTY or EON" Else Print ", Static PTY"
  End If

End If


If Rds_over_rs232_control.2 = 1 Then           'RDS-Infosegment 3
```

```
    Print


    If Ps_flag = 1 Then                         'Ausgabe PS nur bei voll dekodiertem PS
      Print "Program-Name: " ; Ps_name_last
      Ps_flag = 0
    End If


    Call Make_freq_string(freq_fm)
    Print "Frequenz: " ; Freq_string ; " MHz"


    If Af_flag = 2 And Af_max > 0 Then          'Alternate Frequencies ausdrucken, wenn Liste vollständig


      Word1 = Af(1)                             'AF1 zum Vergleich mit möglichen Frequenzpaaren in Methode B
laden
      Method_b = 0


      If Af_max > 2 Then                        'Prüfe auf Methode B
        If Af(2) = Af(1) Or Af(3) = Af(1) Then
          Method_b = 1
        Else
          Method_b = 0
          Reg_flag = 0                          'Regionalisierungs-Flag zurücksetzen
        End If
      End If


      N = 1                                     'N=laufender Index für AF-Speicherstelle
      N1 = 0                                    'N1=Index für tatsächlichen AF-Index (=N bei Methode A;
=Frequenzpaarnummer bei Methode B)
```

```
Do


  If N1 = 0 And Method_b = 1 Then
    Byte1 = Af_max / 2
    Print "Frequenzpaare: " ; Byte1
    Print "Tuned Frequency: ";            'Tuned-frequency bei Methode B (0. Paar)
  Elseif N1 = 0 And Method_b = 0 Then
    Print "Anzahl Alternativ-Frequenzen: " ; Af_max
    Print "1. Altern. Frequ.: " ;
  Else                              'Methode A oder Methode B ab 1.Paar
    If Method_b = 1 Then
      Print N1 ;
      Else
      Print N;
    End If
    Print ". Altern. Frequ.: " ;
  End If


  If Method_b = 0 Then                  'Methode A
    Freq_af = Af(n)
  Else                              'Methode B
    If N1 = 0 Then                      'Hauptfrequenz bei Paarindex N1=0
      Freq_af = Af(1)
    Else                            'AF-Paare dekodieren
      N = N1 * 2                        'N = 1.AF-Index für Frequenzpaar N1
      N2 = N + 1                        'N2= 2.AF-Index für Frequenzpaar N1


      If Af(n) < Af(n2) Then Reg_flag = 0 Else Reg_flag = 1       'Teste auf Regional-Frequenz
```

```
            If Af(n) = Af(1) Then Freq_af = Af(n2) 'Teste, welche Frequenz die AF, und welche die
Wiederholung der Hauptfrequenz ist
              If Af(n2) = Af(1) Then Freq_af = Af(n)
            End If
            Incr N                          'damit >AF_max wird nach Erreichen des letzten Frequenzpaares
          End If

          Incr N : Incr N1                  'Paarindex erhöhen

'           Word1 = Af(n)
          Call Make_freq_string(freq_af)

'           If Af(n) < 1600 Then                        'LW/MW-Frequenz
          If Freq_af < 1600 Then            'LW/MW-Frequenz
            Print Freq_string ; " kHz";
          Else                              'FM-Frequenz
            Print Freq_string ; " MHz";
          End If

          If Reg_flag = 1 Then Print " Regional" Else Print

        Loop Until N > Af_max

        Af_flag = 0                         'Flag wieder zurücksetzen, damit nächste AF-Liste bei Methode B
geholt werden kann

      End If
```

```
'(
    'PI-Codes (Europa)

    Byte1 = R4 : Swap Byte1 : Byte1 = Byte1 And &B00001111          '4 MSB von R4=BlockA isolieren
    If Byte1 <> 0 Then

      Decr Byte1                                    '-1, da Tabelle 0-based
      Print "Country-Code: ";

      If Ecc = &HFF Then                            'keine ECC-Daten aus Gruppe1A

        For N = 0 To 60 Step 15
          Byte2 = Byte1 + N                         'Berechne Tabellenadrese
          Text1 = Lookupstr(byte2 , Country_codes )
          Print Text1 ; " or ";
        Next N
        Print

      Else                                          'genau spezifiertes Land, weil ECC vorhanden
        Byte2 = Ecc And &B00001111                  '4 LSB=Index isolieren
        Byte2 = Byte2 * 15
        Byte2 = Byte2 + Byte1
        Text1 = Lookupstr(byte2 , Country_codes )
        Print Text1
      End If

      Byte1 = R4 And &B00001111                     '4 LSB von R8=BlockC isolieren
      Print "Area-Coverage-Code: ";
      Select Case Byte1
```

```
        Case 0 : Print "Local"
        Case 1 : Print "International"
        Case 2 : Print "National"
        Case 3 : Print "Suparnational"
        Case Else
            Byte2 = Byte1 - 3
            Print "Regional R" ; Byte2
End Select

Print "Program Reference Number: " ; R5

'Language Codes (Europe)
Print
If Lc <> &HFF Then
   Print "Language: ";
   If Lc < &H30 Then
      Text1 = Lookupstr(lc , Language_codes )
   Elseif Lc < &H40 Then
      Text1 = "Reserved"
   Elseif Lc = &H7D Then
      Text1 = "Armenian"
   Elseif Lc = &H7B Then
      Text1 = "Azerbaijan"
   Elseif Lc = &H79 Then
      Text1 = "Belorussian"
   Elseif Lc = &H71 Then
      Text1 = "Georgian"
   Elseif Lc = &H7D Then
      Text1 = "Armenian"
```

```
        Elseif Lc = &H63 Then
          Text1 = "Macedonian"
        Elseif Lc = &H60 Then
          Text1 = "Moldavian"
        Elseif Lc = &H56 Then
          Text1 = "Russian"
        Elseif Lc = &H54 Then
          Text1 = "Serbo-Croat"
        Elseif Lc = &H49 Then
          Text1 = "Ukrainian"
        Elseif Lc = &H40 Then
          Text1 = "Background Sound"
        End If
        Print Text1
      End If

    End If
')

  End If

  If Rds_over_rs232_control.3 = 1 And Stunden <> &HFFFF Then       'RDS-Infosegment 4

    'Zeit + Datum
    Print

    Print "Local Time: " ;
    If Stunden < 10 Then Print "0";
    Print Stunden ; "h " ;
```

```
    If Minuten < 10 Then Print "0";
    Print Minuten ; "m"

    Print "Modified Julian Date: " ; Mjd

    Ys = Mjd : Ys = Ys - 15078.2 : Ys = Ys / 365.25 : Ys = Int(ys)
    Ms = Mjd : Ms = Ms - 14956.1 : Ks = Ys * 365.25 : Ks = Int(ks) : Ms = Ms - Ks : Ms = Ms / 30.6001 : Ms = Int(ms)
    Float1 = Mjd : Float1 = Float1 - 14956 : Float1 = Float1 - Ks : Ks = Ms * 30.6001 : Ks = Int(ks) : Float1 = Float1 - Ks
    Day = Float1
    If Ms = 14 Or Ms = 15 Then K = 1 Else K = 0
    Year = Ys : Year = Year + K : Year = Year + 1900
    K = K * 12
    Month = Ms : Month = Month - 1 : Month = Month - K
    Wd = Mjd + 2 : Wd = Wd Mod 7
    Text1 = Lookupstr(wd , Day_names )
    Print "Date: " ; Text1 ; ", " ; Day ; "." ; Month ; "." ; Year

  End If


  If Rds_over_rs232_control.4 = 1 And Radiotext_ready = 1 Then      'RDS-Infosegment 5

    'Radiotext
    Print
    Print "Radiotext (Länge): " ; Radiotext_length
    Print Left(radiotext , Radiotext_length)          'LF und CR werden automtisch bei RS-232 richtig interpretiert
```

```
    End If



'(
    If Rds_over_rs232_control.5 = 1 And Ews <> &HFFFF Then                  'RDS-Infosegment 6


      'EWS
      Print
      Print "Emergency Warning: $" ; Hex(ews)
      If Ews0 <> &HFF Then                                    'EWS-Daten aus Gruppe 9A
        Print "EWS0...EWS4 = $" ; Hex(ews0) ; " " ; Hex(ews1) ; " " ; Hex(ews2) ; " " ; Hex(ews3) ; " " ;
Hex(ews4)
      End If


    End If
')



    If Rds_over_rs232_control.6 = 1 And Eon_flag = 2 Then  'RDS-Infosegment 7, wenn EON voll dekodiert


      'EON
       Print
       Eon_flag = 0                          'EON-Flag zurücksetzen für Neudecodierung


       Print "EON-Program-Name: " ; Ps_eon


      'PI-Code
      Pi_eon = Makeint(pic2_eon , Pic1_eon)
```

```
Print "PI-EON = $" ; Hex(pi_eon) ; " = %" ; Bin(pic1_eon) ; " " ; Bin(pic2_eon)



'PIN-Code

Print "PIN-EON = $" ; Hex(pin_code_eon)


'Travel-Program, Travel-Announcement

Print "EON-TP = " ; Tp_eon ; " : ";

If Tp_eon = 0 Then Print "No ";

Print "Travel Program"



Print "EON-TA = " ; Ta_eon ; " : ";

If Ta_eon = 0 Then Print "No ";

Print "Travel-Announcement";

If Ta_eon = 1 And Tp_eon = 0 Then                'TA auf anderem Sender via EON

  Print " on EON"

Else

  Print

End If


'EON_PTY


Text1 = Lookupstr(pty_eon , Pty_codes)


Print "EON-Program-Type: " ; Pty_eon ; " = " ; Text1


'EON_AF
```

```
For N = 0 To 5                          '6 Frequenzpaare

  Byte1 = N * 2 : Incr Byte1
  Word1 = Af_eon(byte1)

  If Af_eon(byte1) = Freq_fm Then       '1. AF_EON aus Frequenzpaar = eingestellte Frequenz?

    Incr Byte1                          'rechter Wert aus Frequenzpaar=mapped Frequency
    Call Make_freq_string(word1)
    Print "Mapped Frequency:  " ; Freq_string ;
    If Word1 < 1600 Then Print " kHz" Else Print " MHz"      '

  Else                                  'AF-Methode A

    If Word1 <> 0 Then                  'linke AF aus AF-Paar
      Call Make_freq_string(word1)
      Print "Altern. Frequency: " ; Freq_string ;
      If Word1 < 1600 Then Print " kHz " ; Else Print " MHz ";      '
    End If

    Incr Byte1
    Word1 = Af_eon(byte1)

    If Word1 <> 0 Then                  'rechte AF aus AF-Paar
      Call Make_freq_string(word1)
      Print "Altern. Frequency: " ; Freq_string ;
      If Word1 < 1600 Then Print " kHz" Else Print " MHz"      '
    End If
```

```
        End If

      Next N

      Eon_flag = &HFF                          'EON_Flag wieder zurücksetzen


   End If

   If Rds_over_rs232_control.7 = 1 And Pin_code <> &HFFFF Then       'RDS-Infosegment 8

      Print
      Print "PIN-Code:         $" ; Hex(pin_code)
      Print "Linkage actuator:  $" ; La
      Print "Linkage Information$" ; Hex(link_info)
      Print "Language Code:     $" ; Hex(lc)
      Print "Ext. Country Code: $" ; Hex(ecc)
      Print "TMC-Ident:         $" ; Hex(tmc_ident)
      Print "EWS-Channel:       $" ; Hex(ews)

   End If


   Rds_over_rs232_control = 0                    'Setze RS232-Ausgabe-Flag zurück

End Sub
```

```
'*********************************************
'*** Ersetze RDS-ASCII-Code mit EADOGM-Zeichencode
'*********************************************

Sub Rds_char_code(chr_code As Byte)


  If Chr_code > 127 Then                        'ASCII-Codes über 127?
    Reset Chr_code.7                           ' (Code - 127) = Tabellenindex
    Incr Chr_code
    Chr_code = Replace_codes(chr_code)
  End If

End Sub


'************************************
'*** Radiotext 1x fertig empfangen ***
'************************************

Sub Radiotext_decoded()

   If Char_counter = Position Then                  'Teste, ob Anzahl gezählter Zeichen = aktuelle
Positionszähler

     Radiotext_ready = 1                        'Flag auf fertig dekodierten RT-String
     Radiotext_start_decode = 3                   'Flag auf fertig dekodierten RT-String
     Radiotext = Radiotext_new                    'hole dekodierten String in "RADIOTEXT"-String
```

```
    Radiotext_length = Char_counter                 'Aktuelle Position des letzten Zeichens/Zeichenzählers =
Radiotext_Länge

    Call Radiotext_start                    '

'    Radiotext_lf_pos = Charpos(radiotext , 10)                 'Position des nächsten LF

    Home Upper : Lcd Display_string                 'RT-String in 1. Zeile LCD bringen

  Else

    Radiotext_start_decode = 1                  'Dekodier-Status auf Anfang zurücksetzen

  End If

End Sub



'*****************************************************
'*** Starte Radiotext-Anzeige mit ersten 16 Zeichen ***
'*****************************************************

Sub Radiotext_start()

Local Missing_spaces As Byte

  Call Clear_lcd_upperline                  'Am Beginn 1s Frequenzanzeige
```

```
   Call Fm_tune_status_freq_print_lcd
   Wait 1


   If Radiotext_length < 16 Then                    'Radiotext kürzer als 16 Zeichen?

     Radiotext = Left(radiotext , Radiotext_length)      'fülle RT mit Spaces auf 16 Zeichen auf
     Missing_spaces = 16 - Radiotext_length
     Radiotext = Radiotext + Space(missing_spaces)
     Radiotext_length = 16

   End If

   Radiotext_display_pos = 1                        'Zähler für Postion nächstes 16-Zeichen-Segment

   End_position = Radiotext_length - 14             'Letzte Zählerposition

   Display_string = Left(radiotext , 16)            'Isoliere erste LCD-Zeile des Display_strings

   Home Upper : Lcd Display_string                  'Drucke auf LCD aus

   Scroll_counter = 2 : Direction = 0               '2x stehen lassen, bevor weitergescrollt wird

End Sub


'****************************************
'*** Scrolle Radiotext-Lauftext auf LCD ***
```

```
'****************************************

Sub Radiotext_scroll()

   Local Switch_position As Byte
   Switch_position = End_position - 1

   If Radiotext_length > 16 Then                    'scrollen nur bei Länge>16

     'Ping-Pong-Scrollen
     If Radiotext_length < 25 Then

       If Direction = 0 Then                        'nach rechts

         Incr Radiotext_display_pos

         If Radiotext_display_pos = Switch_position Then Scroll_counter = 2      'rechtes Ende erreicht --> 2x stehen lassen

         If Radiotext_display_pos = End_position Then   'rechtes Ende erreicht
           Toggle Direction                         'Richtung umkehren
         Else
           Display_string = Mid(radiotext , Radiotext_display_pos , 16)      '16-Zeichen-String aus Radiotext ausschneiden
         End If

       Else                                         'nach links

         Decr Radiotext_display_pos
```

```
        If Radiotext_display_pos = 1 Then Scroll_counter = 2       'rechtes Ende erreicht 6x stehen lassen


        If Radiotext_display_pos = 0 Then            'rechtes Ende erreicht
          Toggle Direction                          'Richtung umkehren
        Else
          Display_string = Mid(radiotext , Radiotext_display_pos , 16)      '16-Zeichen-String aus Radiotext ausschneiden
        End If


      End If


      Home Upper
      Lcd Display_string                            'drucke auf LCD aus


    'normales Scrollen
    Else


      Incr Radiotext_display_pos


      If Radiotext_display_pos = Switch_position Then Scroll_counter = 2       '2x stehen lassen


      If Radiotext_display_pos = End_position Then      'Scrolling beendet?
        Call Radiotext_start
      Else
        Display_string = Mid(radiotext , Radiotext_display_pos , 16)      '16-Zeichen-String aus Radiotext ausschneiden
        Home Upper
        Lcd Display_string                          'drucke auf LCD aus
```

```
        End If

      End If

   End If

End Sub



'*****************************************
'*** Berechne Frequenz einer AF (FM/LW/MW) ***
'*****************************************

Sub Rds_calc_af(af_index As Byte , Af_word As Byte)

  If Af_index = 250 Then                    'LW/MW-Frequenz folgt im nächsten Byte

    Lw_mw_af = 1 : Af_word = 0

  Elseif Af_index > 0 And Af_index < 205 Then        'Code zwischen 1...204 => gültige  AF

    If Lw_mw_af = 0 Then                    'FM-Frequenz

      Af_word = Af_index + 875                'AF berechnen (in 100kHz-Einheiten)
      Af_word = Af_word * 10

    Else                                    'AM-Frequenz
```

```
      Lw_mw_af = 0                        'Flag wieder zurücksetzen

      Af_word = 9 * Af_index              '9kHz-Einheiten

      If Af_index < 16 Then               'LW
        Af_word = Af_word + 144
      Else                                'MW
        Af_word = Af_word + 387
      End If

    End If

  Else                                    'ungültige Frequenz --> 0 zurückgeben

    Af_word = 0

  End If

End Sub



'*****************************************************************
'*** RDS-Variablen zurücksetzen nach SYNC-Lost oder neuem Sender ***
'*****************************************************************

Sub Rds_reset()
```

```
   Tp = &HFF : Ta = &HFF                        'Travel Program, Travel Announcement-Flag

   Pi = &HFFFF                                  'Programm-Identification Code

   Di_counter = 0 : Di(1) = &HFF : : Di(2) = &HFF : Di(3) = &HFF : Di(4) = &HFF       'Decoder-Information-
Zähler

   Ps_name = Space(8) : Ps_name_last = Ps_name : Ps_flag = 0 : Ps_counter = 0

   Af_flag = 0 : Af_max = 0 : Lw_mw_af = 0              'div. Startdaten für Alternate Frequencies

   La = 0                                       'Linkage Actuator-Flag

   Vc = &HFF                                    'Variant Code

   Ecc = &HFF                                   'Extended Country Code

   Lc = &HFFFF                                  'Language Code

   Ews = &HFFFF : Ews0 = &HFF                       'Emergency Warning System

   Pin_code = &HFFFF                            'Program-Identification-Code

   Tmc_ident = &HFFFF                           'TMC-Identification

   Eon_flag = &HFF : Pin_code_eon = &HFFFF


   Radiotext_present = 0

   Radiotext_first_aqu = 0

   Radiotext_start_decode = 0 : Radiotext_ready = 0


   Minuten = &HFFFF : Stunden = &HFFFF : Utc_offset = &HFF : Mjd = &HFFFFFFFF       'Zeitvariablen
'Modified Julian Date

   Pty_name = Space(8) : Pty_name_flag = 0              'PTYN aus Gruppe 10A




End Sub
```

```
'*************************************************************************
'*** aktuell eingestellter Modus, Frequenz FM/AM und Band im EEPROM abspeichern ***
'*************************************************************************

Sub Eeprom_save()                       'Frequenzdaten in EEPROM sichern

   Writeeeprom Am_fm_flag , 1000
   Writeeeprom Freq_fm
   Writeeeprom Freq_am
   Writeeeprom Band

End Sub



'*******************************
'*** untere LCD-Zeile löschen ***
'*******************************

Sub Clear_lcd_lowerline()               'untere LCD-Zeile löschen

   Home Lower
   Lcd Space(16)

End Sub

'*******************************
'*** untere LCD-Zeile löschen ***
'*******************************
```

```
Sub Clear_lcd_upperline()                    'untere LCD-Zeile löschen

   Home Upper
   Lcd Space(16)

End Sub



'*****************************************
'*** Flags für Eingabemodus zurücksetzen ***
'*****************************************

Sub Reset_flags()                            'Eingabeflags zurücksetzen

   Memory_aktiv_flag = 0
   Property_aktiv = 0
   Am_textaktiv = 0
   Cursor Off

End Sub



'*************************************************
'*** Encoderwerte nach Veränderung auf Null setzen ***
'*************************************************

Sub Reset_encoder()                          'Encoder auf 0 zurücksetzen
```

```
   Encoder1 = 0
   Encoder1old = Encoder1
   Enable Timer0

End Sub




'*****************************************************
'************** RS-232: Frequenz-Einstellung *********
'*****************************************************


Sub F_control()

   Print "Tune"

   Input Fin

   If Am_fm_flag = 1 Then                    'im AM-Modus?
     If Fin >= 23000 Then                    'prüfe, ob Frequenz oberhalb 23MHz
       Am_fm_flag = 0                        '--> schalte auf FM-Modus um
       Call Si4735_init
     End If
   End If
```

```
   If Am_fm_flag = 0 Then                    'im FM-Modus?
'     If Fin < 000 Then                          'prüfe, ob Frequenz unterhalb 60MHz
'       Am_fm_flag = 1                            '--> schalte auf AM-Modus um
'       Call Si4735_init
'    End If
   End If

   If Am_fm_flag = 1 Then                    'setze AM-Frequenz

     Freq_am = Fin
     Call Am_tune_freq
     Call Am_tune_status_freq_print_lcd

   Else                                      'setze FM-Frequenz

     Fin = Fin / 10
     Freq_fm = Fin
     Call Fm_tune_freq
     Call Fm_tune_status_freq_print_lcd

   End If

End Sub



'****************************************
'*** RS-232: Senderspeicher AM eingeben ***
```

```
'****************************************

Sub Mam_control()

   Print "Memory AM"

   Input N                                'Speicherplatz holen
   Input Fin                              'Frequenz in kHz holen
   Input Text1                             'Sendername holen

   If N < 51 Then
     If N > Am_memorymax Then Am_memorymax = N
     Freq = Fin
     Am_memory(n) = Freq
     Text1 = Text1 + "        "
     Text1 = Left(text1 , 10)
     Am_text(n) = Text1
   End If

End Sub

'****************************************
'*** RS-232: Senderspeicher FM eingeben ***
'****************************************

Sub Mfm_control()
```

```
    Print "Memory FM"

    Input N
    Input Fin

    If N < 51 Then
      If N > Fm_memorymax Then Fm_memorymax = N
      Fin = Fin / 10
      Freq = Fin
      Fm_memory(n) = Freq
    End If

End Sub




'*********************************
'*** RS-232: Property eingeben ***
'*********************************

Sub Properties()

    Print "Property"

    Input N                                 'Property-Nummer holen

    If N < 17 Then
```

```
    Input Dat                          'Property-Wert holen                                    '

    Property_dat(n) = Dat
    Property_adr = Property_adress(n)
    Call Si4735_set_property(property_adr , &H00 , Dat)  'Property-Wert setzen

  End If

End Sub



'*****************************
'*** RS-232: I2C-Kommando  ***
'*****************************

Sub Pc_control_i2c()

 Print #1 , "I2C"

 Do

  Get #1 , Command                    'I2C write and read to SI4735

  If Command = 67 Then                 '"C"
    Input #1 , Bytesout
    Input #1 , Bytesin
    For N = 1 To Bytesout
      Inputhex #1 , Data_i2c(n)
```

```
  Next N
  I2cstart
  I2cwbyte 34
  For N = 1 To Bytesout
    I2cwbyte Data_i2c(n)
  Next N
  I2cstop
  If Bytesin > 0 Then
    Waitms 1
    I2cstart
    I2cwbyte 35
    While Bytesin > 1
      Bytesin = Bytesin - 1
      I2crbyte D , Ack                    'Binärausgabe der gelesenen Bytes vom I2C-Bus
      Print #1 , Bin(d)
    Wend
    I2crbyte D , Nack
    Print #1 , Bin(d)                     'Binärausgabe des letzten gelesenen Bytes vom I2C-Bus
    I2cstop
  End If
End If

If Command = 65 Then                      'LCD Line 1
  Input #1 , Text1
  Home Upper
  Lcd Text1
End If

If Command = 66 Then                      'LCD Line 2
```

```
      Input #1 , Text1
      Home Lower
      Lcd Text1
    End If

  Loop

End Sub



'********************
'*** RDS-Ausgabe ***
'********************

Sub Rds_print_control()

    Print : Print "RDS-Data over RS-232 --> Enter Control Byte: ";

    Input Rds_over_rs232_control : Print                '

End Sub









'*****************************************
```

```
'*** TIMER0-Interrupt für Encoder-Abfrage ***
'*****************************************

Tim0_isr:

$asm
      push      r16
      push      r17
      push      r20
      push      r21
      push      r24
      push      r26
      push      r27
      in        r24,sreg
      push      r24

$end Asm

 Code1 = Pinc
 Code1 = Code1 And &B00000110                    'PC2 und PC1 isolieren

 If Code1.2 < Code1old.2 Then
  If Code1.1 = 1 Then Encoder1 = Encoder1 + 1
  If Code1.1 = 0 Then Encoder1 = Encoder1 - 1
 End If

 Code1old = Code1

$asm
```

```
    pop     r24
    Out Sreg , R24
    pop     r27
    pop     r26
    pop     r24
    pop     r21
    pop     r20
    pop     r17
    pop     r16

$end Asm


Return


End









'******************
'******************
'*** DATA-Werte ***
'******************
'******************
'******************
```

```
$eeprom
$eepromhex                          'Intel HEX-Format für EEP-File für USBTiny-Programmer
'***EEPROMHEX-Direktive Auskommentieren bei Simulation oder Brennen mit Normal-Brenner!!!***

Data 0                              'Leerbyte
Data 1                              'Anzahl belegter FM-Senderspeicher


'10 LW-Speicherfrequenzen
Data 153%                           '1.LW-Senderfrequenz
Data 162%
Data 177%
Data 183%
Data 198%
Data 207%
Data 216%
Data 234%
Data 252%
Data 279%

'10 MW-Speicherfrequenzen
Data 549%                           '1.MW-Senderfrequenz
Data 621%
'Data 675%
Data 693%
Data 720%
Data 747%
```

Data 756%

'Data 801%

Data 810%

'Data 828%

Data 864%

'Data 1008%

'data 1215%

Data 1422%

Data 1440%

'Data 1593%

'10 KW-Speicerhfrequenzen, Bandbereich 1

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

'10 KW-Speicerhfrequenzen, Bandbereich 2

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

'10 KW-Speicerhfrequenzen, Bandbereich 3

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

'50 FM-Speicherfrequenzen

Data 10080% '1.FM-Frequenz

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

Data 0%

'10 LW Speichernamen

Data "DLF " '1.LW-Sendernamen

Data "France Int"

Data "DLR Kultur"

Data "Europe One"

Data "BBC4/BBCWS"

Data "DLF "

Data "MonteCarlo"

Data "RTL-Luxemb"

Data "Chaine 3 "

Data "Belarus 1 "

'10 MW Speichernamen

```
Data "DLF       "                          '2. MW-Sendernamen
Data "RTBF/Bel  "
'Data "RadioMaria"
Data "BBC5/RUSSL"
Data "WDR2/VERA "
Data "Radio5 NL "
Data "DLF       "
'Data "BR Plus   "
Data "BBC Scotld"
'Data "NDR Info  "                                  'oder "Radio10 NL"
Data "FranceBleu"
'Data "Gr.Nieuws "
'Data "V.o.Russia"
Data "DLF       "
Data "RTL/China "
'Data "WDR2      "

'10 KW Speichernamen, KW-Bandbereich 1
Data "          "                          '1.KW-Sendername; Tropenbänder
Data "          "
Data "          "
Data "          "
Data "          "
Data "          "
Data "          "
Data "          "
Data "          "
Data "          "
```

'10 KW Speichernamen, KW-Bandbereich 2

Data " "

Data " "

Data " "

Data " "

Data " "

Data " "

Data " "

Data " "

Data " "

Data " "

Data " "

'10 KW Speichernamen, KW-Bandbereich 3

Data " "

Data " "

Data " "

Data " "

Data " "

Data " "

Data " "

Data " "

Data " "

Data " "

Data " "

'Frei-Bytes 752-994

Data 52 , 53 , 54 , 55 , 56 , 57 , 58 , 59 , 60 , 61 , 62 , 63 , 64 , 65 , 66

Data 67 , 68 , 69 , 70 , 71 , 72 , 73 , 74 , 75 , 76 , 77 , 78 , 79 , 80 , 81 , 82 , 83 , 84 , 85 , 86 , 87 , 88 , 89 , 90 , 91 , 92 , 93 , 94 , 95 , 96 , 97 , 98 , 99

Data 0 , 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

Data 33 , 34 , 35 , 36 , 37 , 38 , 39 , 40 , 41 , 42 , 43 , 44 , 45 , 46 , 47 , 48 , 49 , 50 , 51 , 52 , 53 , 54 , 55 , 56 , 57 , 58 , 59 , 60 , 61 , 62 , 63 , 64 , 65 , 66

Data 67 , 68 , 69 , 70 , 71 , 72 , 73 , 74 , 75 , 76 , 77 , 78 , 79 , 80 , 81 , 82 , 83 , 84 , 85 , 86 , 87 , 88 , 89 , 90 , 91 , 92 , 93 , 94 , 95 , 96 , 97 , 98 , 99

Data 0 , 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

Data 33 , 34 , 35 , 36 , 37 , 38 , 39 , 40 , 41 , 42 , 43 , 44 , 45 , 46 , 47 , 48 , 49 , 50 , 51 , 52 , 53 , 54 , 55 , 56 , 57 , 58 , 59 , 60 , 61 , 62 , 63 , 64 , 65 , 66

Data 67 , 68 , 69 , 70 , 71 , 72 , 73 , 74 , 75 , 76 , 77 , 78 , 79 , 80 , 81 , 82 , 83 , 84 , 85 , 86 , 87 , 88 , 89 , 90 , 91 , 92 , 93 , 94

'Adresse 995:

Data 10 'Lw_memorymax = Position des höchsten LW-Speichers

Data 20 'Mw_memorymax = Position des höchsten MW-Speichers

Data 20 'Kw1_memorymax = Position des höchsten KW1-Speichers

Data 30 'Kw2_memorymax = Position des höchsten KW2-Speichers

Data 40 'Kw3_memorymax = Position des höchsten KW2-Speichers

'Adresse 1000

Data 0 'letzter Modus: AM/FM

Data 10080% 'letzte FM-Frequenz

Data 648% 'letzte AM-Frequenz

Data 2 'letztes AM-Band

'Adresse 1006

Data &H01 , &H31 , &H1E , &H40 , &H10 , &H04 , &H03 , &H14 '16 Property-Default-Werte ab 1006

```
Data &H00 , &H03 , &H40 , &H02 , &H10 , &H0A , &H05 , &H19


'Adresse 1022       = Flag für FM-PWM-Ausgabe
Data 0

'Adresse 1023       = Flag für manuelle Schrittweite
Data 0




$data


'Band-Bezeichnungen
Band_names:
Data " "
Data "Langwelle"
Data "Mittelwelle"
Data "120m Tropenband"                  'KW-Bereich 1; AM-Speicher 21-30
Data "90m Tropenband"
Data "75m Tropenband"
Data "60m Tropenband"
Data "49m Europaband"                   'KW-Bereich 2; AM-Speicher 31-40
Data "41m KW-Band"
Data "31m KW-Band"
Data "25m KW-Band"
Data "21m KW-Band"                      'KW-Bereich 3: AM-Speicher 41-50
Data "19m KW-Band"
```

```
Data "16m KW-Band"

Data "15m KW-Band"

Data "13m KW-Band"

Data "11m KW-Band"


Band_data:

Data 153% , 279%                        'LW-Bandgrenzen

Data 522% , 1710%                       'MW-Bandgrenzen

Data 2300% , 2495%                      'KW-Band1 Bandgrenzen

Data 3200 % , 3400%

Data 3900% , 4000%

Data 4750% , 5060%

Data 5900% , 6200%

Data 7100% , 7450%

Data 9400% , 9900%

Data 11600% , 12100%

Data 13570% , 13870%

Data 15100% , 15800%

Data 17480% , 17900%

Data 18900% , 19020%

Data 21450% , 21850%

Data 25600% , 26100%

Data 8750% , 10800%                     'FM-Bandgrenzen

'Band_start(17) = 6400                          'unterste FM-Grenze

'Call Si4735_set_property(&H1400 , 64 , 00)             'auf 64 MHz setzen



'Property-Adressen N=1...16; max./min.-Werte; Default-Werte
```

```
Property_data:
Data &H1100% , 1 , 2 , &H01                 'FM-Deemphasis
Data &H1105% , 0 , 127 , &H31                'FM-Stereo-Schwelle
Data &H1106% , 0 , 127 , &H1E                'FM-Mono-Schwelle
Data &H1300% , 1 , 255 , &H40                'FM Mute-Geschwindigkeit
Data &H1302% , 0 , 31 , &H10                'FM Soft_Mute, max. Absenkung
Data &H1303% , 0 , 15 , &H04                'FM Soft_Mute, SNR-Schwelle
Data &H1403% , 0 , 127 , &H03                'FM Suchlauf, SNR-Schwelle
Data &H1404% , 0 , 127 , &H14                'FM Suchlauf, Signalstärke-Schwelle
Data &H3100% , 0 , 1 , &H00                 'AM-Deemphasis
Data &H3102% , 0 , 4 , &H03                 'AM-Bandbreite
Data &H3300% , 11 , 255 , &H40                'AM Soft-Mute Geschwindigkeit
Data &H3301% , 1 , 5 , &H02                 'AM Soft-Mute Steilheit
Data &H3302% , 0 , 63 , &H10                'AM Soft_Mute, max. Absenkung
Data &H3303% , 0 , 63 , &H0A                'AM Soft_Mute, SNR-Schwelle
Data &H3403% , 0 , 63 , &H05                'AM Suchlauf, SNR-Schwelle
Data &H3404% , 0 , 63 , &H19                'AM Suchlauf, Signalstärke-Schwelle


'Property-Bezeichnungen (8 Zeichen f. LCD)
Property_names:
Data " "
Data "Deemphas"
Data "ST-Blend"
Data "MO-Blend"
Data "MuteRate"
Data "MuteAttn"
Data "Mute SNR"
Data "Seek SNR"
```

Data "SeekRSSI"

Data "Deemphas"

Data "Bndwidth"

Data "MuteRate"

Data "MuteSlop"

Data "MuteAttn"

Data "Mute SNR"

Data "Seek SNR"

Data "SeekRSSI"

Pty_codes:

'(

'RDS-PTY-Klartext-Bezeichnung (16-Zeichen-Code)

Data "None" '8-Character Anzeige " None "

Data "News" '8-Character Anzeige " News "

Data "Current Affairs" '8-Character Anzeige "Affairs "

Data "Information" '8-Character Anzeige " Info "

Data "Sport" '8-Character Anzeige " Sport "

Data "Education" '8-Character Anzeige "Educate "

Data "Drama" '8-Character Anzeige " Drama "

Data "Culture" '8-Character Anzeige "Culture "

Data "Science" '8-Character Anzeige "Science "

Data "Varied Speech" '8-Character Anzeige " Varied "

Data "Pop Music" '8-Character Anzeige " Pop M "

Data "Rock Music" '8-Character Anzeige " Rock M "

Data "Easy Listening" '8-Character Anzeige " Easy M "

Data "Light Classics M" '8-Character Anzeige "Light M "

Data "Serious Classics" '8-Character Anzeige "Classics"

```
Data "Other Music"                          '8-Character Anzeige "Other M "
Data "Weather & Metr"                       '8-Character Anzeige "Weather "
Data "Finance"                              '8-Character Anzeige "Finance "
Data "Children's Progs"                     '8-Character Anzeige "Children"
Data "Social Affairs"                       '8-Character Anzeige " Social "
Data "Religion"                             '8-Character Anzeige "Religion"
Data "Phone In"                             '8-Character Anzeige "Phone In"
Data "Travel & Touring"                     '8-Character Anzeige " Travel "
Data "Leisure & Hobby"                      '8-Character Anzeige "Leisure "
Data "Jazz Music"                           '8-Character Anzeige "  Jazz  "
Data "Country Music"                        '8-Character Anzeige "Country "
Data "National Music"                       '8-Character Anzeige "Nation M"
Data "Oldies Music"                         '8-Character Anzeige " Oldies "
Data "Folk Music"                           '8-Character Anzeige " Folk M "
Data "Documentary"                          '8-Character Anzeige "Document"
Data "Alarm Test"                           '8-Character Anzeige "  TEST  "
Data "ALARM!"                               '8-Character Anzeige " ALARM! "
')

'RDS-PTY-Klartext-Bezeichnung (8-Zeichen-Code)
Data "None"
Data "News"
Data "Affairs"
Data "Info"
Data "Sport"
Data "Educate"
Data "Drama"
Data "Culture"
Data "Science"
```

Data "Varied"

Data "Pop M"

Data "Rock M"

Data "Easy M"

Data "Light M"

Data "Classics"

Data "Other M"

Data "Weather"

Data "Finance"

Data "Children"

Data "Social"

Data "Religion"

Data "Phone In"

Data "Travel"

Data "Leisure"

Data "Jazz"

Data "Country"

Data "Nation M"

Data "Oldies"

Data "Folk M"

Data "Document"

Data "TEST"

Data "ALARM!"

'RDS-Ländercodes

'(

Country_codes:

Data "DE" , "DZ" , "AD" , "IL" , "IT" , "BE" , "RU" , "PS" , "AL" , "AT" , "HU" , "MT" , "DE" , "nn" , "EU"

Data "GR" , "CY" , "SM" , "CH" , "JO" , "FI" , "LU" , "BG" , "DK" , "GI" , "IQ" , "GB" , "LY" , "RO" , "FR"

Data "MA" , "CZ" , "PL" , "VA" , "SK" , "SY" , "TN" , "nn" , "LI" , "IS" , "MC" , "LT" , "YU" , "ES" , "NO"

Data "IE" , "TR" , "MK" , "nn" , "nn" , "nn" , "NL" , "LV" , "LB" , "nn" , "HR" , "nn" , "SE" , "BY" , "nn"

Data "MD" , "EE" , "nn" , "nn" , "nn" , "UA" , "nn" , "PT" , "SI" , "nn" , "nn" , "nn" , "nn" , "nn" , "BA"

')

'(

'RDS-Spachen-Kodierung

Language_codes:

Data "unknown" , "Albanian" , "Breton" , "Catalan" , "Croatian" , "Welsh" , "Czech" , "Danish" , "German" , "English" , "Spanish" , "Esperanto" , "Estonian" , "Basque" , "French"

Data "Frisian" , "Irish" , "Gaelic" , "Galician" , "Icelandic" , "Italian" , "Lappish" , "Latin" , "Latvian" , "Luxembourgian" , "Lithuanian" , "Hungarian" , "Maltese" , "Dutch" , "Norwegian" , "Occitan"

Data "Polish" , "Portugese" , "Romanian" , "Romansh" , "Serbian" , "Slovak" , "Slovene" , "Finnish" , "Swedish" , "Turkish" , "Flemish" , "Walloon" , " " , " " , " " , " "

')

Day_names:

Data "Monday" , "Tuesday" , "Wednesday" , "Thursday" , "Friday" , "Saturday" , "Sunday"

'RDS-Sonderzeichen ab ASCII 128

Special_characters:

Data &HE0 , &H85 , &H82 , &H8A , &HE1 , &H8D , &HE2 , &H95 , &HE3 , &H97 , &H9C , &H80 , &H20 , &H00 , &HE9 , &H20

Data &H83 , &H84 , &H88 , &H89 , &H86 , &H8B , &H93 , &H94 , &H96 , &H81 , &H9B , &H87 , &H20 , &H20 , &H20 , &H20

Data &H9D , &H1F , &H03 , &H04 , &H20 , &H20 , &H20 , &H99 , &H19 , &HE4 , &HE5 , &H24 , &H08 , &H05 , &H06 , &H07

Data &HF2 , &H20 , &H20 , &H02 , &H20 , &HE9 , &H6E , &H9A , &H75 , &H9F , &HF8 , &HDF , &HF6 , &HF5 , &H20 , &H12

Data &H41 , &H41 , &H90 , &H45 , &H49 , &H49 , &H4F , &H4F , &H55 , &H55 , &H52 , &H43 , &H53 , &H5A , &H44 , &H4C

Data &H41 , &H8E , &H45 , &H45 , &H49 , &H8B , &H93 , &H99 , &H95 , &H9A , &H72 , &H63 , &H73 , &H7A , &H64 , &H6C

Data &HEC , &H8F , &H92 , &H99 , &H98 , &H59 , &HEC , &HEE , &H4E , &H6E , &H52 , &H43 , &H53 , &H5A , &H20 , &H20

Data &HED , &H83 , &H91 , &H94 , &H77 , &H79 , &HED , &HEF , &H6E , &H9A , &H72 , &H63 , &H73 , &H7A , &H20 , &H20