ATmega8GPSLoggerBasismodul

20131018

Generated by Doxygen 1.8.7

Fri Jun 20 2014 14:56:36

Contents

Chapter 1

Todo List

File atmega8_gps_logger_basismodul.c

Code schöner schreiben, Kommtentare einfügen, Doxygen Doku verbessern

- FAT, UART und MMC Libraries in eigenem Ordner speichern
- In neuer Version auf FAT verzichten und Daten ohne Dateisystem auf SD Karte schreiben (siehe g↩ LoggerMini) -> weniger Speicher für Code, schneller?
- Wartezeit für GPS Datenempfang einstellbar machen (eventuell nur Positionsdaten auslesen?)
- GPS Maus mit anderen Einstellungen initialisieren (höhere Datenrate, andere NMEA Sätze?)
- Start / Stop aus dem File entfernen -> Überprüfen ob Datei vorhanden und dann neue anlegen (Dateinamen bei jedem starten ändern und damit neue Datei anlegen -> String append ?)
- Akkuspannung messen und bei zu niedriger Spannung -> sleep mode (Datei abschließen), GPS und Zusatzmodul abschalten und warnen
- GPS nur einschalten wenn Aufzeichnung aktiviert, sonst ausschalten
- GPS Daten auf Display ausgeben, zusätzliche Sensoren, Jeti Kommunikation -> größerer uC benötigt? -> ATmega32?
- Berechnungen für Oldtimerrallye Training machen und am Display ausgeben
- Erkennung von Zusatzmodulen und Code für diese
- Bei Verwendung mit LCD Zusatzmodul das Logging abschaltbar machen -> nur Anzeige der Daten

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Chapter 3

Data Structure Index

3.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Module Documentation

5.1 UART Library

Interrupt UART library using the built-in UART with transmit and receive circular buffers.

Macros

- #define UART_BAUD_SELECT(baudRate, xtalCpu) ((xtalCpu)/((baudRate)∗16l)-1)

  UART Baudrate Expression.
- #define UART_BAUD_SELECT_DOUBLE_SPEED(baudRate, xtalCpu) (((xtalCpu)/((baudRate)∗8l)-1)|0x8000)

  UART Baudrate Expression for ATmega double speed mode.
- #define UART_RX_BUFFER_SIZE 64
- #define UART_TX_BUFFER_SIZE 64
- #define UART_FRAME_ERROR 0x0800 /∗ Framing Error by UART ∗/
- #define UART_OVERRUN_ERROR 0x0400 /∗ Overrun condition by UART ∗/
- #define UART_BUFFER_OVERFLOW 0x0200 /∗ receive ringbuffer overflow ∗/
- #define UART_NO_DATA 0x0100 /∗ no receive data available ∗/
- #define uart_puts_P(__s) uart_puts_p(PSTR(__s))

  Macro to automatically put a string constant into program memory.
- #define uart1_puts_P(__s) uart1_puts_p(PSTR(__s))

  Macro to automatically put a string constant into program memory.

Functions

- void uart_init (unsigned int baudrate)

  Initialize UART and set baudrate.
- unsigned int uart_getc (void)

  Get received byte from ringbuffer.
- void uart_putc (unsigned char data)

  Put byte to ringbuffer for transmitting via UART.
- void uart_puts (const char ∗s)

  Put string to ringbuffer for transmitting via UART.
- void uart_puts_p (const char ∗progmem_s)

  Put string from program memory to ringbuffer for transmitting via UART.
- void uart1_init (unsigned int baudrate)

  Initialize USART1 (only available on selected ATmegas)

- unsigned int uart1_getc (void)

  Get received byte of USART1 from ringbuffer. (only available on selected ATmega)
- void uart1_putc (unsigned char data)

  Put byte to ringbuffer for transmitting via USART1 (only available on selected ATmega)
- void uart1_puts (const char ∗s)

  Put string to ringbuffer for transmitting via USART1 (only available on selected ATmega)
- void uart1_puts_p (const char ∗s)

  Put string from program memory to ringbuffer for transmitting via USART1 (only available on selected ATmega)

5.1.1 Detailed Description

Interrupt UART library using the built-in UART with transmit and receive circular buffers.

```
#include <uart.h>
```

This library can be used to transmit and receive data through the built in UART.

An interrupt is generated when the UART has finished transmitting or receiving a byte. The interrupt handling routines use circular buffers for buffering received and transmitted data.

The UART_RX_BUFFER_SIZE and UART_TX_BUFFER_SIZE constants define the size of the circular buffers in bytes. Note that these constants must be a power of 2. You may need to adapt this constants to your target and your application by adding CDEFS += -DUART_RX_BUFFER_SIZE=nn -DUART_RX_BUFFER_SIZE=nn to your Makefile.

Note

Based on Atmel Application Note AVR306

Author

Peter Fleury pfleury@gmx.ch http://jump.to/fleury

5.1.2 Macro Definition Documentation

5.1.2.1 #define uart1_puts_P(__s) uart1_puts_p(PSTR(__s))

Macro to automatically put a string constant into program memory.

Definition at line 186 of file uart.h.

5.1.2.2 #define UART_BAUD_SELECT(baudRate, xtalCpu) ((xtalCpu)/((baudRate)∗16l)-1)

UART Baudrate Expression.

constants and macros

Parameters

xtalCpu	system clock in Mhz, e.g. 4000000L for 4Mhz
baudRate	baudrate in bps, e.g. 1200, 2400, 9600

Definition at line 67 of file uart.h.

Referenced by gps_init().

5.1.2.3 #define UART_BAUD_SELECT_DOUBLE_SPEED(baudRate, xtalCpu) (((xtalCpu)/((baudRate)∗8l)-1)|0x8000)

UART Baudrate Expression for ATmega double speed mode.

Parameters

xtalCpu	system clock in Mhz, e.g. 4000000L for 4Mhz
baudRate	baudrate in bps, e.g. 1200, 2400, 9600

Definition at line 73 of file uart.h.

5.1.2.4 #define UART_BUFFER_OVERFLOW 0x0200 /* receive ringbuffer overflow */

Definition at line 95 of file uart.h.

Referenced by gps_get_char(), and SIGNAL().

5.1.2.5 #define UART_FRAME_ERROR 0x0800 /* Framing Error by UART */

test if the size of the circular buffers fits into SRAM

high byte error return code of uart_getc()

Definition at line 93 of file uart.h.

5.1.2.6 #define UART_NO_DATA 0x0100 /* no receive data available */

Definition at line 96 of file uart.h.

Referenced by gps_get_char(), and uart_getc().

5.1.2.7 #define UART_OVERRUN_ERROR 0x0400 /* Overrun condition by UART */

Definition at line 94 of file uart.h.

5.1.2.8 #define uart_puts_P(__s) uart_puts_p(PSTR(__s))

Macro to automatically put a string constant into program memory.

Definition at line 171 of file uart.h.

5.1.2.9 #define UART_RX_BUFFER_SIZE 64

Size of the circular receive buffer, must be power of 2

Definition at line 78 of file uart.h.

5.1.2.10 #define UART_TX_BUFFER_SIZE 64

Size of the circular transmit buffer, must be power of 2

Definition at line 82 of file uart.h.

5.1.3 Function Documentation

5.1.3.1 unsigned int uart1_getc (void)

Get received byte of USART1 from ringbuffer. (only available on selected ATmega)

See also

uart_getc

5.1.3.2 void uart1_init (unsigned int *baudrate*)

Initialize USART1 (only available on selected ATmegas)

See also

uart_init

5.1.3.3 void uart1_putc (unsigned char *data*)

Put byte to ringbuffer for transmitting via USART1 (only available on selected ATmega)

See also

uart_putc

5.1.3.4 void uart1_puts (const char * *s*)

Put string to ringbuffer for transmitting via USART1 (only available on selected ATmega)

See also

uart_puts

5.1.3.5 void uart1_puts_p (const char * *s*)

Put string from program memory to ringbuffer for transmitting via USART1 (only available on selected ATmega)

See also

uart_puts_p

5.1.3.6 unsigned int uart_getc (void)

Get received byte from ringbuffer.

Returns in the lower byte the received character and in the higher byte the last receive error. UART_NO_DATA is returned when no data is available.

Returns

lower byte: received byte from ringbuffer
higher byte: last receive status

- **0** successfully received data from UART
- **UART_NO_DATA**
  no receive data available
- **UART_BUFFER_OVERFLOW**
  Receive ringbuffer overflow. We are not reading the receive buffer fast enough, one or more received character have been dropped

- **UART_OVERRUN_ERROR**
  Overrun condition by UART. A character already present in the UART UDR register was not read by the interrupt handler before the next character arrived, one or more received characters have been dropped.
- **UART_FRAME_ERROR**
  Framing Error by UART

uart_getc() return byte from ringbuffer Returns: lower byte: received byte from ringbuffer higher byte: last receive error < no data available

calculate / store buffer index

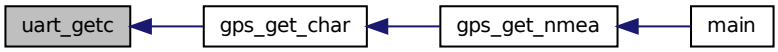
get data from receive buffer

Definition at line 390 of file uart.c.

References UART_NO_DATA, and UART_RX_BUFFER_MASK.

Referenced by gps_get_char().

Here is the caller graph for this function:



5.1.3.7 void uart_init (unsigned int *baudrate*)

Initialize UART and set baudrate.

function prototypes

Parameters

baudrate	Specify baudrate using macro UART_BAUD_SELECT()

uart_init() initialize UART and set baudrate

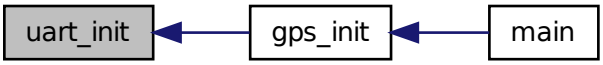
Parameters

in	*baudrate*	using macro UART_BAUD_SELECT()

Definition at line 313 of file uart.c.

Referenced by gps_init().

Here is the caller graph for this function:



5.1.3.8 void uart_putc (unsigned char *data*)

Put byte to ringbuffer for transmitting via UART.

Parameters

data	byte to be transmitted

uart_putc() write byte to ringbuffer for transmitting via UART

Parameters

in	*data*	byte to be transmitted

wait for free space in buffer

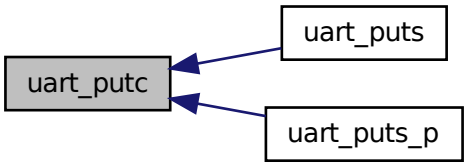
enable UDRE interrupt

Definition at line 416 of file uart.c.

References UART_TX_BUFFER_MASK.

Referenced by uart_puts(), and uart_puts_p().

Here is the caller graph for this function:



5.1.3.9 void uart_puts (const char * *s*)

Put string to ringbuffer for transmitting via UART.

The string is buffered by the uart library in a circular buffer and one character at a time is transmitted to the UART using interrupts. Blocks if it can not write the whole string into the circular buffer.

Parameters

s	string to be transmitted

uart_puts() transmit string to UART

Parameters

in	*s*	string to be transmitted

Definition at line 440 of file uart.c.

References uart_putc().

Here is the call graph for this function:

5.1.3.10 void uart_puts_p (const char * *progmem_s*)

Put string from program memory to ringbuffer for transmitting via UART.

The string is buffered by the uart library in a circular buffer and one character at a time is transmitted to the UART using interrupts. Blocks if it can not write the whole string into the circular buffer.

Parameters

progmem_s	program memory string to be transmitted

See also

uart_puts_P

uart_puts_p() transmit string from program memory to UART

Parameters

in	*progmem_s*	program memory string to be transmitted

Definition at line 452 of file uart.c.

References uart_putc().

Here is the call graph for this function:

Chapter 6

Data Structure Documentation

6.1 afile Struct Reference

`#include <fat16.h>`

Data Fields

- unsigned long start_cluster
- unsigned long cluster_pointer
- unsigned char sector_index
- unsigned int byte_index
- unsigned char mode
- unsigned long filesize
- unsigned long fileposition
- unsigned long sector_in_buffer
- unsigned long directory_sector
- unsigned char directory_index
- unsigned char attribute

6.1.1 Detailed Description

Definition at line 10 of file fat16.h.

6.1.2 Field Documentation

6.1.2.1 unsigned char afile::attribute

Definition at line 22 of file fat16.h.

Referenced by fclose_(), fopen_(), ScanSubDirectories(), and SeekDirectoryEntry().

6.1.2.2 unsigned int afile::byte_index

Definition at line 15 of file fat16.h.

Referenced by fclose_(), fflush_(), fgetchar_(), fopen_(), fputchar_(), and fseek_().

6.1.2.3 unsigned long afile::cluster_pointer

Definition at line 13 of file fat16.h.

Referenced by AppendCluster(), fclose_(), fflush_(), fgetchar_(), fopen_(), fputchar_(), fseek_(), GetFatClusterOffset(), GetNextCluster(), ScanSubDirectories(), and SeekDirectoryEntry().

6.1.2.4 unsigned char afile::directory_index

Definition at line 21 of file fat16.h.

Referenced by CreateDirectoryEntry(), fclose_(), fflush_(), fopen_(), and SeekDirectoryEntry().

6.1.2.5 unsigned long afile::directory_sector

Definition at line 20 of file fat16.h.

Referenced by CreateDirectoryEntry(), fclose_(), fflush_(), fopen_(), and SeekDirectoryEntry().

6.1.2.6 unsigned long afile::fileposition

Definition at line 18 of file fat16.h.

Referenced by fclose_(), fopen_(), fputchar_(), and fseek_().

6.1.2.7 unsigned long afile::filesize

Definition at line 17 of file fat16.h.

Referenced by fclose_(), fflush_(), fgetchar_(), fopen_(), fputchar_(), fseek_(), and SeekDirectoryEntry().

6.1.2.8 unsigned char afile::mode

Definition at line 16 of file fat16.h.

Referenced by fclose_(), fflush_(), and fopen_().

6.1.2.9 unsigned long afile::sector_in_buffer

Definition at line 19 of file fat16.h.

Referenced by fclose_(), fgetchar_(), FindNextFreeCluster(), fopen_(), and GetNextCluster().

6.1.2.10 unsigned char afile::sector_index

Definition at line 14 of file fat16.h.

Referenced by fclose_(), fflush_(), fgetchar_(), fopen_(), fputchar_(), and fseek_().

6.1.2.11 unsigned long afile::start_cluster

Definition at line 12 of file fat16.h.

Referenced by fclose_(), fopen_(), fseek_(), and SeekDirectoryEntry().

The documentation for this struct was generated from the following file:

- fat16.h

6.2 DirEntry Struct Reference

`#include <fat16.h>`

Data Fields

- unsigned char name [8]
- unsigned char extension [3]
- unsigned char attribute
- unsigned char reserved [10]
- unsigned int time
- unsigned int date
- unsigned int startcluster
- unsigned long size

6.2.1 Detailed Description

Definition at line 31 of file fat16.h.

6.2.2 Field Documentation

6.2.2.1 unsigned char DirEntry::attribute

Definition at line 35 of file fat16.h.

Referenced by CreateDirectoryEntry(), and SeekDirectoryEntry().

6.2.2.2 unsigned int DirEntry::date

Definition at line 38 of file fat16.h.

Referenced by fflush_().

6.2.2.3 unsigned char DirEntry::extension[3]

Definition at line 34 of file fat16.h.

6.2.2.4 unsigned char DirEntry::name[8]

Definition at line 33 of file fat16.h.

Referenced by CreateDirectoryEntry(), and SeekDirectoryEntry().

6.2.2.5 unsigned char DirEntry::reserved[10]

Definition at line 36 of file fat16.h.

6.2.2.6 unsigned long DirEntry::size

Definition at line 40 of file fat16.h.

Referenced by CreateDirectoryEntry(), fflush_(), fread_(), fwrite_(), and SeekDirectoryEntry().

6.2.2.7 unsigned int DirEntry::startcluster

Definition at line 39 of file fat16.h.

Referenced by CreateDirectoryEntry(), and SeekDirectoryEntry().

6.2.2.8 unsigned int DirEntry::time

Definition at line 37 of file fat16.h.

Referenced by fflush_().

The documentation for this struct was generated from the following file:

- fat16.h

6.3 FatEntry Struct Reference

`#include <fat16.h>`

Data Fields

- unsigned int next_cluster

6.3.1 Detailed Description

Definition at line 49 of file fat16.h.

6.3.2 Field Documentation

6.3.2.1 unsigned int FatEntry::next_cluster

Definition at line 51 of file fat16.h.

Referenced by AppendCluster(), and FindNextFreeCluster().

The documentation for this struct was generated from the following file:

- fat16.h

Chapter 7

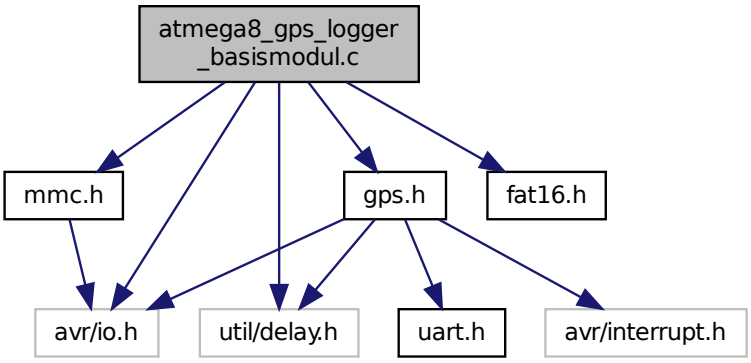
File Documentation

7.1 atmega8_gps_logger_basismodul.c File Reference

Hauptfile des Projekts ATmega8 GPS Logger.

```
#include <avr/io.h>
#include <util/delay.h>
#include "gps.h"
#include "mmc.h"
#include "fat16.h"
```

Include dependency graph for atmega8_gps_logger_basismodul.c:



Macros

- #define F_CPU 16000000L
- #define LED_PORT PORTD
- #define LED_STAT PD6
- #define LED_WARN PD7
- #define LEDCODE_OFF LED_PORT &= ~((1 << LED_WARN) | (1 << LED_STAT))
- #define LEDCODE_OK LED_PORT = (LED_PORT & ~(1 << LED_WARN)) | (1 << LED_STAT)
- #define LEDCODE_WARNING LED_PORT = (LED_PORT & ~(1 << LED_STAT)) | (1 << LED_WARN)
- #define LEDCODE_PROCESSING LED_PORT |= (1 << LED_WARN) | (1 << LED_STAT)
- #define LEDCODE_BLINK LED_PORT ^= (1 << LED_STAT)

Functions

- int main (void)

  Hauptfunktion.

Variables

- struct {
  unsigned record:1
  unsigned keydown:1
  } status

7.1.1 Detailed Description

Hauptfile des Projekts ATmega8 GPS Logger.

Author

Martin Matysiak (mail@k621.de)
V. Pippan (webmaster@vpippan.at)

Date

2013-10-18

Version

20131018

Todo

- Code schöner schreiben, Kommtentare einfügen, Doxygen Doku verbessern
- FAT, UART und MMC Libraries in eigenem Ordner speichern
- In neuer Version auf FAT verzichten und Daten ohne Dateisystem auf SD Karte schreiben (siehe g↩ LoggerMini) -> weniger Speicher für Code, schneller?
- Wartezeit für GPS Datenempfang einstellbar machen (eventuell nur Positionsdaten auslesen?)
- GPS Maus mit anderen Einstellungen initialisieren (höhere Datenrate, andere NMEA Sätze?)

- Start / Stop aus dem File entfernen -> Überprüfen ob Datei vorhanden und dann neue anlegen (Dateinamen bei jedem starten ändern und damit neue Datei anlegen -> String append ?)
- Akkuspannung messen und bei zu niedriger Spannung -> sleep mode (Datei abschließen), GPS und Zusatzmodul abschalten und warnen
- GPS nur einschalten wenn Aufzeichnung aktiviert, sonst ausschalten
- GPS Daten auf Display ausgeben, zusätzliche Sensoren, Jeti Kommunikation -> größerer uC benötigt? -> ATmega32?
- Berechnungen für Oldtimerrallye Training machen und am Display ausgeben
- Erkennung von Zusatzmodulen und Code für diese
- Bei Verwendung mit LCD Zusatzmodul das Logging abschaltbar machen -> nur Anzeige der Daten

Definition in file atmega8_gps_logger_basismodul.c.

7.1.2 Macro Definition Documentation

7.1.2.1 #define F_CPU 16000000L

Definition at line 51 of file atmega8_gps_logger_basismodul.c.

Referenced by gps_init().

7.1.2.2 #define LED_PORT PORTD

Definition at line 53 of file atmega8_gps_logger_basismodul.c.

7.1.2.3 #define LED_STAT PD6

Definition at line 54 of file atmega8_gps_logger_basismodul.c.

7.1.2.4 #define LED_WARN PD7

Definition at line 55 of file atmega8_gps_logger_basismodul.c.

7.1.2.5 #define LEDCODE_BLINK LED_PORT ^= (1 << LED_STAT)

Definition at line 61 of file atmega8_gps_logger_basismodul.c.

Referenced by main().

7.1.2.6 #define LEDCODE_OFF LED_PORT &= ~((1 << LED_WARN) | (1 << LED_STAT))

Definition at line 57 of file atmega8_gps_logger_basismodul.c.

Referenced by main().

7.1.2.7 #define LEDCODE_OK LED_PORT = (LED_PORT & ~(1 << LED_WARN)) | (1 << LED_STAT)

Definition at line 58 of file atmega8_gps_logger_basismodul.c.

Referenced by main().

7.1.2.8 #define LEDCODE_PROCESSING LED_PORT |= (1 << LED_WARN) | (1 << LED_STAT)

Definition at line 60 of file atmega8_gps_logger_basismodul.c.

Referenced by main().

7.1.2.9 #define LEDCODE_WARNING LED_PORT = (LED_PORT & ~(1 << LED_STAT)) | (1 << LED_WARN)

Definition at line 59 of file atmega8_gps_logger_basismodul.c.

Referenced by main().

7.1.3 Function Documentation

7.1.3.1 int main (void)

Hauptfunktion.

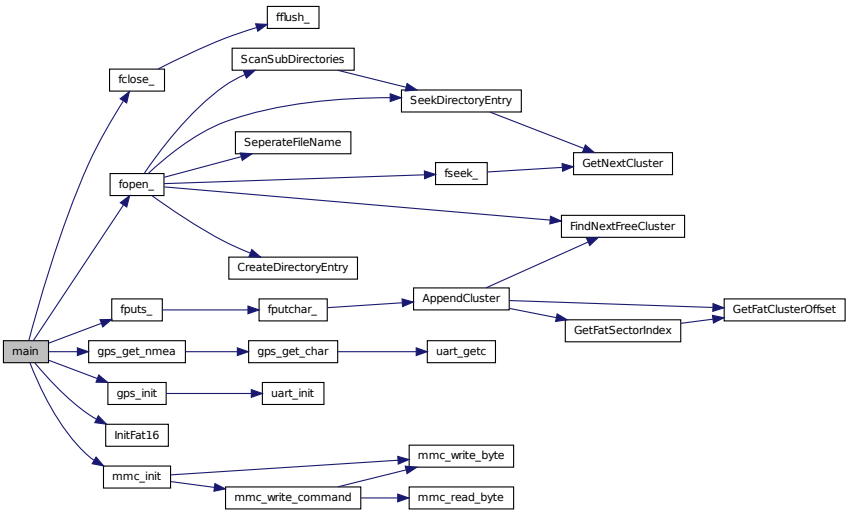
Die Hauptfunktion enthält die Initialisierungen beim Programmstart und die Hauptschleife des Programms. < definiert ein character feld für nmea daten mit 128 zeichen länge

Hauptschleife

Definition at line 74 of file atmega8_gps_logger_basismodul.c.

References fclose_(), fopen_(), fputs_(), gps_get_nmea(), gps_init(), InitFat16(), LEDCODE_BLINK, LEDCODE_↩ OFF, LEDCODE_OK, LEDCODE_PROCESSING, LEDCODE_WARNING, mmc_init(), and status.

Here is the call graph for this function:



7.1.4 Variable Documentation

7.1.4.1 unsigned keydown

Definition at line 66 of file atmega8_gps_logger_basismodul.c.

7.1.4.2 unsigned record

Definition at line 65 of file atmega8_gps_logger_basismodul.c.

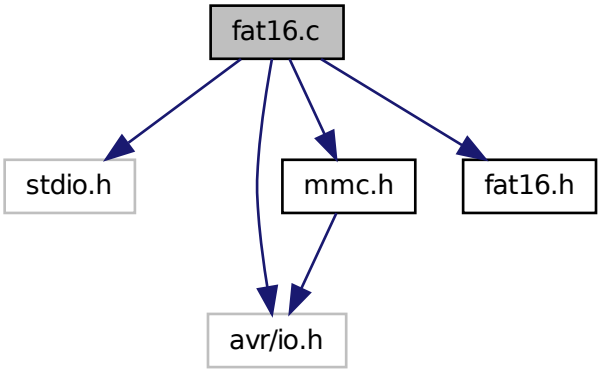
7.1.4.3 struct { ... } status

Referenced by main().

7.2 fat16.c File Reference

```
#include <stdio.h>
#include <avr/io.h>
#include "mmc.h"
#include "fat16.h"
```

Include dependency graph for fat16.c:



Functions

- unsigned char InitFat16 (void)
- unsigned char fopen_ (unsigned char *fname, char mode, File *file)
- int fflush_ (File *file)
- void fclose_ (File *file)
- unsigned long fread_ (void *buffer, unsigned long size, unsigned long count, File *file)
- unsigned long fwrite_ (void *buffer, unsigned long size, unsigned long count, File *file)
- int fseek_ (File *file, long offset, int origin)
- int fgetchar_ (File *file)
- unsigned char fputchar_ (File *file, char c)
- unsigned char fputs_ (File *file, char *string)
- char * fgets_ (char *string, int count, File *file)
- unsigned char fexist_ (unsigned char *fname, File *file)
- unsigned int GetNextCluster (File *file)
- unsigned int FindNextFreeCluster (File *file)
- unsigned char AppendCluster (File *file)
- unsigned int GetFatClusterOffset (File *file)

- unsigned int GetFatSectorIndex (File *file)
- unsigned char CreateDirectoryEntry (unsigned char *fname, unsigned int cluster, File *file, unsigned char attrib)
- unsigned char SeekDirectoryEntry (unsigned char *fname, File *file)
- unsigned char ScanSubDirectories (unsigned char *fname, File *file)
- void SeperateFileName (unsigned char *fname, unsigned char *name)

Variables

- unsigned char SectorsPerCluster = 0
- unsigned char FatCopies = 0
- unsigned int PossibleRootEntries = 0
- unsigned int SectorsPerFat = 0
- unsigned long ReservedSectors = 0
- unsigned long FirstPartitionSector = 0
- unsigned long FileAllocationTable = 0
- unsigned long RootDirectory = 0
- unsigned long FirstDataCluster = 0
- unsigned char FileBuffer [512]
- struct DirEntry * DirectoryEntry
- struct FatEntry * Fat

7.2.1 Function Documentation

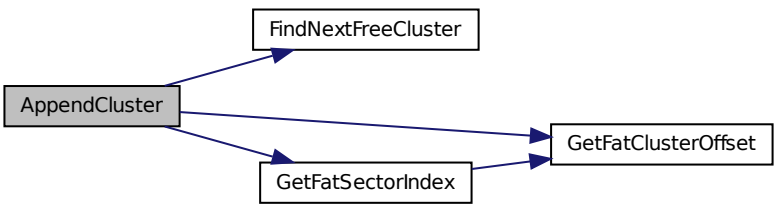
7.2.1.1 unsigned char AppendCluster (File * *file*)

Definition at line 614 of file fat16.c.

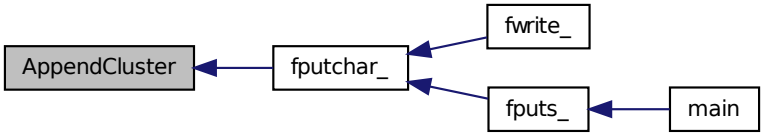
References afile::cluster_pointer, FileAllocationTable, FileBuffer, FindNextFreeCluster(), FirstDataCluster, GetFatClusterOffset(), GetFatSectorIndex(), FatEntry::next_cluster, SDC_GetSector, SDC_PutSector, and SectorsPerCluster.

Referenced by fputchar_().

Here is the call graph for this function:



Here is the caller graph for this function:



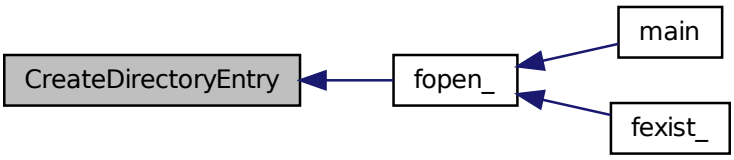
7.2.1.2 unsigned char CreateDirectoryEntry (unsigned char * *fname,* unsigned int *cluster,* File * *file,* unsigned char *attrib*)

Definition at line 687 of file fat16.c.

References DirEntry::attribute, afile::directory_index, afile::directory_sector, FileBuffer, DirEntry::name, PossibleRootEntries, RootDirectory, SDC_GetSector, SDC_PutSector, DirEntry::size, and DirEntry::startcluster.

Referenced by fopen_().

Here is the caller graph for this function:



7.2.1.3 void fclose_ (File * *file*)

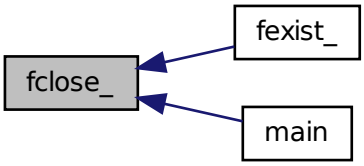
Definition at line 195 of file fat16.c.

References afile::attribute, afile::byte_index, afile::cluster_pointer, afile::directory_index, afile::directory_sector, fflush_(), afile::fileposition, afile::filesize, afile::mode, afile::sector_in_buffer, afile::sector_index, and afile::start_cluster.

Referenced by fexist_(), and main().

Here is the call graph for this function:

Here is the caller graph for this function:

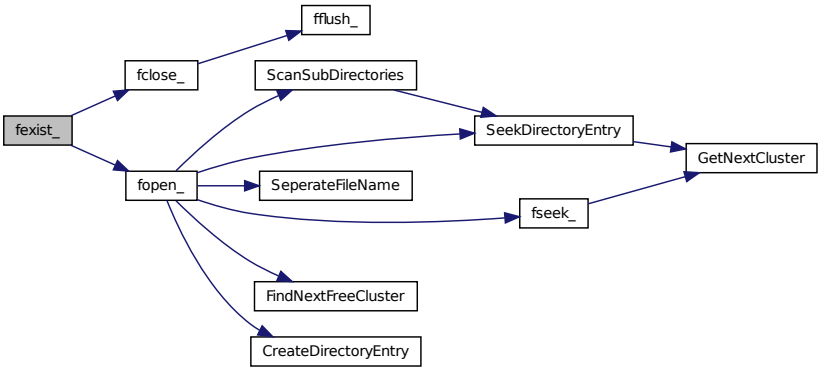


7.2.1.4 unsigned char fexist_ (unsigned char * *fname,* File * *file*)

Definition at line 505 of file fat16.c.

References fclose_(), and fopen_().

Here is the call graph for this function:



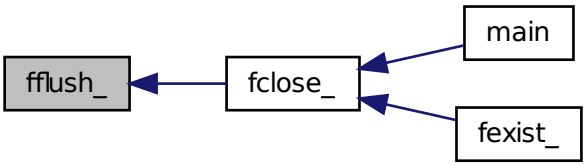
7.2.1.5 int fflush_ (File * *file*)

Definition at line 159 of file fat16.c.

References afile::byte_index, afile::cluster_pointer, DirEntry::date, afile::directory_index, afile::directory_sector, FileBuffer, afile::filesize, afile::mode, SDC_GetSector, SDC_PutSector, afile::sector_index, DirEntry::size, and Dir↩ Entry::time.

Referenced by fclose_().

Here is the caller graph for this function:



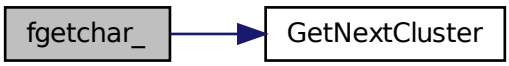
7.2.1.6 int fgetchar_ (File * *file*)

Definition at line 356 of file fat16.c.

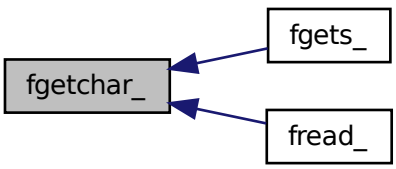
References afile::byte_index, afile::cluster_pointer, FileBuffer, afile::filesize, GetNextCluster(), SDC_GetSector, afile::sector_in_buffer, afile::sector_index, and SectorsPerCluster.

Referenced by fgets_(), and fread_().

Here is the call graph for this function:



Here is the caller graph for this function:

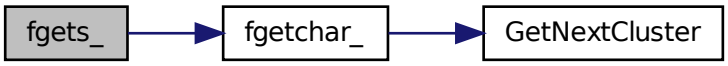


7.2.1.7 char* fgets_ (char * *string,* int *count,* File * *file*)

Definition at line 463 of file fat16.c.

References fgetchar_().

Here is the call graph for this function:



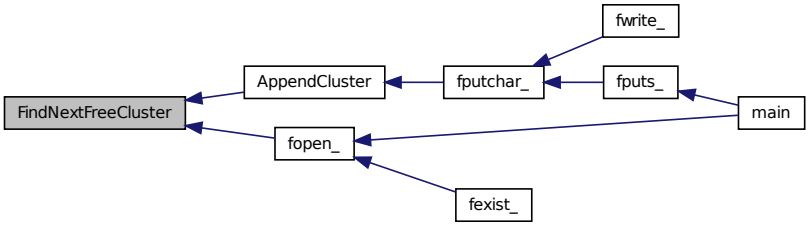
7.2.1.8 unsigned int FindNextFreeCluster (File * *file*)

Definition at line 574 of file fat16.c.

References FileAllocationTable, FileBuffer, FatEntry::next_cluster, SDC_GetSector, SDC_PutSector, afile::sector_in_buffer, and SectorsPerFat.

Referenced by AppendCluster(), and fopen_().

Here is the caller graph for this function:



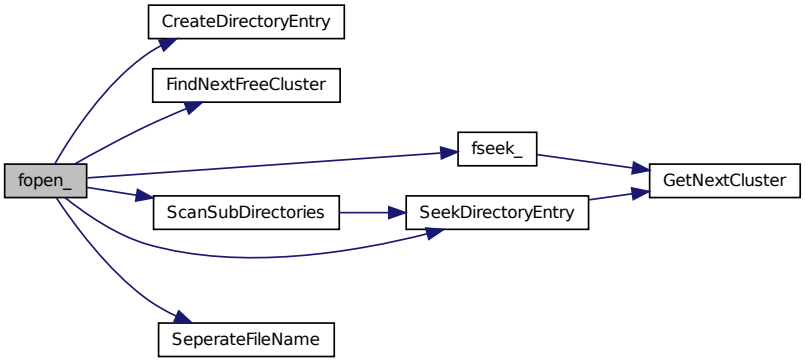
7.2.1.9 unsigned char fopen_ (unsigned char * *fname,* char *mode,* File * *file*)

Definition at line 97 of file fat16.c.

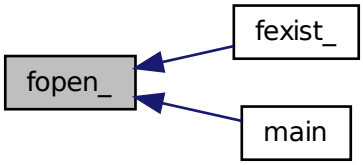
References _FILE, afile::attribute, afile::byte_index, afile::cluster_pointer, CreateDirectoryEntry(), afile::directory_index, afile::directory_sector, afile::fileposition, afile::filesize, FindNextFreeCluster(), FirstDataCluster, fseek_(), afile::mode, ScanSubDirectories(), afile::sector_in_buffer, afile::sector_index, SectorsPerCluster, SeekDirectoryEntry(), SeperateFileName(), and afile::start_cluster.

Referenced by fexist_(), and main().

Here is the call graph for this function:



Here is the caller graph for this function:



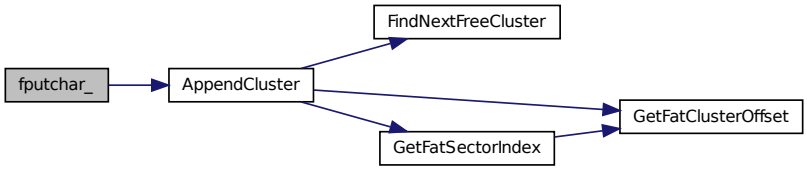
7.2.1.10 unsigned char fputchar_ (File * *file,* char *c*)

Definition at line 400 of file fat16.c.

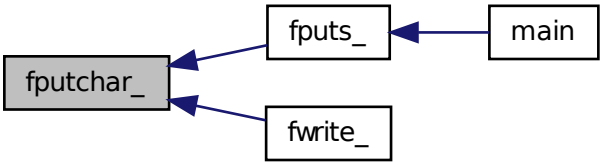
References AppendCluster(), afile::byte_index, afile::cluster_pointer, FileBuffer, afile::fileposition, afile::filesize, SDC_PutSector, afile::sector_index, and SectorsPerCluster.

Referenced by fputs_(), and fwrite_().

Here is the call graph for this function:



Here is the caller graph for this function:



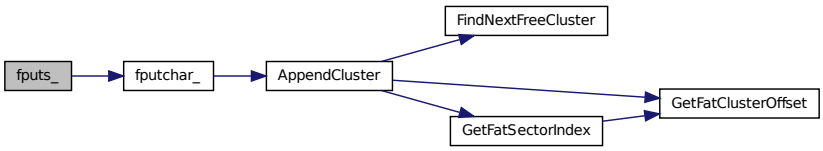
7.2.1.11 unsigned char fputs_ (File * *file,* char * *string*)

Definition at line 443 of file fat16.c.

References fputchar_().

Referenced by main().

Here is the call graph for this function:



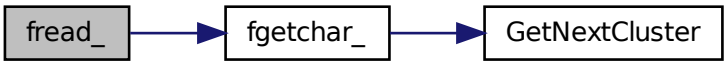
Here is the caller graph for this function:

7.2.1.12 unsigned long fread_ (void * *buffer,* unsigned long *size,* unsigned long *count,* File * *file*)

Definition at line 221 of file fat16.c.

References fgetchar_(), and DirEntry::size.

Here is the call graph for this function:



7.2.1.13 int fseek_ (File * *file,* long *offset,* int *origin*)

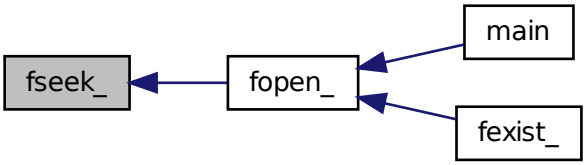
Definition at line 286 of file fat16.c.

References afile::byte_index, afile::cluster_pointer, FileBuffer, afile::fileposition, afile::filesize, GetNextCluster(), SDC_GetSector, afile::sector_index, SectorsPerCluster, and afile::start_cluster.

Referenced by fopen_().

Here is the call graph for this function:

Here is the caller graph for this function:

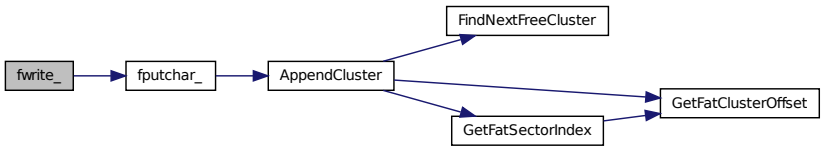


7.2.1.14 unsigned long fwrite_ (void * *buffer,* unsigned long *size,* unsigned long *count,* File * *file*)

Definition at line 254 of file fat16.c.

References fputchar_(), and DirEntry::size.

Here is the call graph for this function:



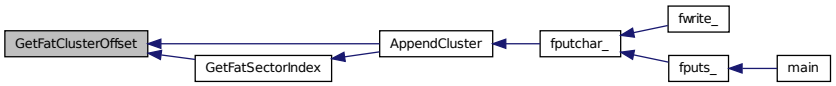
7.2.1.15 unsigned int GetFatClusterOffset (File * *file*)

Definition at line 646 of file fat16.c.

References afile::cluster_pointer, FirstDataCluster, and SectorsPerCluster.

Referenced by AppendCluster(), and GetFatSectorIndex().

Here is the caller graph for this function:



7.2.1.16 unsigned int GetFatSectorIndex (File * *file*)

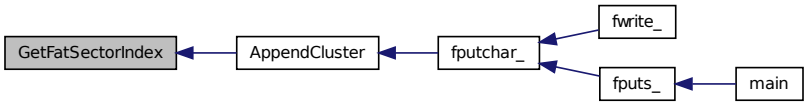
Definition at line 664 of file fat16.c.

References GetFatClusterOffset().

Referenced by AppendCluster().

Here is the call graph for this function:

Here is the caller graph for this function:



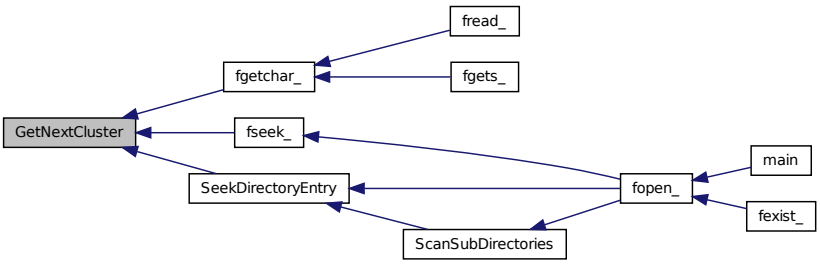
7.2.1.17 unsigned int GetNextCluster (File * *file*)

Definition at line 526 of file fat16.c.

References afile::cluster_pointer, FileAllocationTable, FileBuffer, FirstDataCluster, RootDirectory, SDC_GetSector, afile::sector_in_buffer, and SectorsPerCluster.

Referenced by fgetchar_(), fseek_(), and SeekDirectoryEntry().

Here is the caller graph for this function:



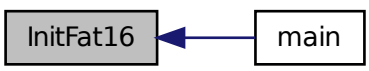
7.2.1.18 unsigned char InitFat16 (void)

Definition at line 71 of file fat16.c.

References FatCopies, FileAllocationTable, FileBuffer, FirstDataCluster, FirstPartitionSector, MBR_SECTOR, PossibleRootEntries, ReservedSectors, RootDirectory, SDC_GetSector, SectorsPerCluster, and SectorsPerFat.

Referenced by main().

Here is the caller graph for this function:



7.2.1.19 unsigned char ScanSubDirectories (unsigned char * *fname,* File * *file*)

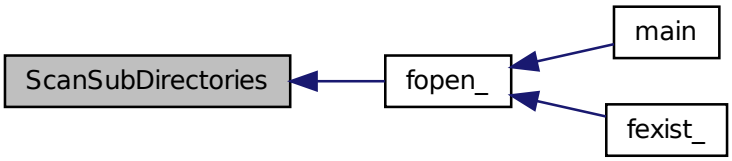
Definition at line 790 of file fat16.c.

References _DIRECTORY, _FILE, afile::attribute, afile::cluster_pointer, RootDirectory, and SeekDirectoryEntry().

Referenced by fopen_().

Here is the call graph for this function:

Here is the caller graph for this function:



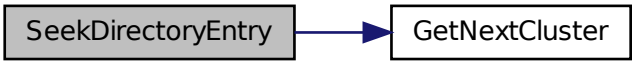
7.2.1.20 unsigned char SeekDirectoryEntry (unsigned char * *fname,* File * *file*)

Definition at line 736 of file fat16.c.

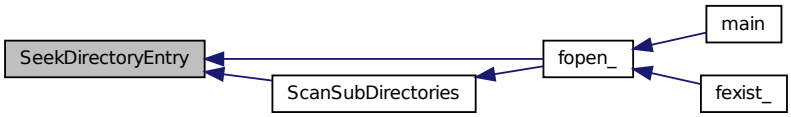
References afile::attribute, DirEntry::attribute, afile::cluster_pointer, afile::directory_index, afile::directory_sector, FileBuffer, afile::filesize, FirstDataCluster, GetNextCluster(), DirEntry::name, SDC_GetSector, SectorsPerCluster, DirEntry::size, afile::start_cluster, and DirEntry::startcluster.

Referenced by fopen_(), and ScanSubDirectories().

Here is the call graph for this function:



Here is the caller graph for this function:

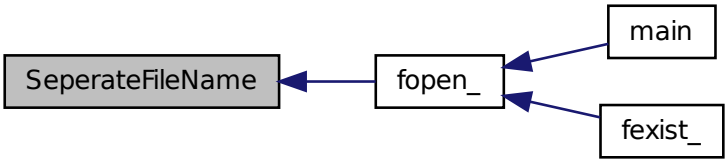


7.2.1.21 void SeperateFileName (unsigned char * *fname,* unsigned char * *name*)

Definition at line 854 of file fat16.c.

Referenced by fopen_().

Here is the caller graph for this function:



7.2.2 Variable Documentation

7.2.2.1 struct DirEntry* DirectoryEntry

Definition at line 56 of file fat16.c.

7.2.2.2 struct FatEntry* Fat

Definition at line 57 of file fat16.c.

7.2.2.3 unsigned char FatCopies = 0

Definition at line 46 of file fat16.c.

Referenced by InitFat16().

7.2.2.4 unsigned long FileAllocationTable = 0

Definition at line 51 of file fat16.c.

Referenced by AppendCluster(), FindNextFreeCluster(), GetNextCluster(), and InitFat16().

7.2.2.5 unsigned char FileBuffer[512]

Definition at line 55 of file fat16.c.

Referenced by AppendCluster(), CreateDirectoryEntry(), fflush_(), fgetchar_(), FindNextFreeCluster(), fputchar_(), fseek_(), GetNextCluster(), InitFat16(), and SeekDirectoryEntry().

7.2.2.6 unsigned long FirstDataCluster = 0

Definition at line 53 of file fat16.c.

Referenced by AppendCluster(), fopen_(), GetFatClusterOffset(), GetNextCluster(), InitFat16(), and Seek↩ DirectoryEntry().

7.2.2.7 unsigned long FirstPartitionSector = 0

Definition at line 50 of file fat16.c.

Referenced by InitFat16().

7.2.2.8 unsigned int PossibleRootEntries = 0

Definition at line 47 of file fat16.c.

Referenced by CreateDirectoryEntry(), and InitFat16().

7.2.2.9 unsigned long ReservedSectors = 0

Definition at line 49 of file fat16.c.

Referenced by InitFat16().

7.2.2.10 unsigned long RootDirectory = 0

Definition at line 52 of file fat16.c.

Referenced by CreateDirectoryEntry(), GetNextCluster(), InitFat16(), and ScanSubDirectories().

7.2.2.11 unsigned char SectorsPerCluster = 0

Definition at line 45 of file fat16.c.

Referenced by AppendCluster(), fgetchar_(), fopen_(), fputchar_(), fseek_(), GetFatClusterOffset(), GetNext↩ Cluster(), InitFat16(), and SeekDirectoryEntry().

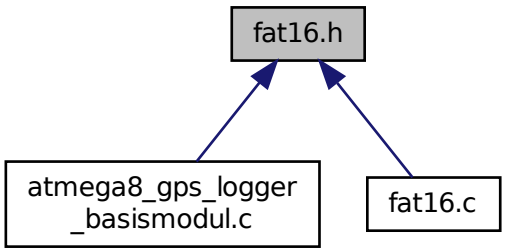
7.2.2.12 unsigned int SectorsPerFat = 0

Definition at line 48 of file fat16.c.

Referenced by FindNextFreeCluster(), and InitFat16().

7.3 fat16.h File Reference

This graph shows which files directly or indirectly include this file:



Data Structures

- struct afile
- struct DirEntry
- struct FatEntry

Macros

- #define MBR_SECTOR 0
- #define _UNUSED 1
- #define _ARCHIVE 2
- #define _READ_ONLY 4
- #define _SYSTEM 8
- #define _DIRECTORY 16
- #define _FILE 32

Typedefs

- typedef struct afile File

Functions

- unsigned char InitFat16 (void)
- unsigned char fopen_ (unsigned char ∗fname, char mode, File ∗file)
- int fflush_ (File ∗file)
- void fclose_ (File ∗file)
- unsigned long fread_ (void ∗buffer, unsigned long size, unsigned long count, File ∗file)
- unsigned long fwrite_ (void ∗buffer, unsigned long size, unsigned long count, File ∗file)
- int fseek_ (File ∗file, long offset, int origin)
- int fgetchar_ (File ∗file)
- unsigned char fputchar_ (File ∗file, char c)
- unsigned char fputs_ (File ∗file, char ∗string)
- char ∗ fgets_ (char ∗s, int count, File ∗file)

- int rename (char *oldname, char *newname)
- unsigned char fexist_ (unsigned char *fname, File *file)
- unsigned char CreateDirectoryEntry (unsigned char *fname, unsigned int cluster, File *file, unsigned char attrib)
- unsigned int FindNextFreeCluster (File *file)
- unsigned char SeekDirectoryEntry (unsigned char *fname, File *file)
- void SeperateFileName (unsigned char *fname, unsigned char *name)
- unsigned char ScanSubDirectories (unsigned char *fname, File *file)
- unsigned int GetNextCluster (File *file)
- unsigned char AppendCluster (File *file)
- unsigned int GetFatClusterOffset (File *file)
- unsigned int GetFatSectorIndex (File *file)

Variables

- unsigned char SectorsPerCluster
- unsigned char FileBuffer [512]
- File myfile

7.3.1 Macro Definition Documentation

7.3.1.1 #define _ARCHIVE 2

Definition at line 99 of file fat16.h.

7.3.1.2 #define _DIRECTORY 16

Definition at line 102 of file fat16.h.

Referenced by ScanSubDirectories().

7.3.1.3 #define _FILE 32

Definition at line 103 of file fat16.h.

Referenced by fopen_(), and ScanSubDirectories().

7.3.1.4 #define _READ_ONLY 4

Definition at line 100 of file fat16.h.

7.3.1.5 #define _SYSTEM 8

Definition at line 101 of file fat16.h.

7.3.1.6 #define _UNUSED 1

Definition at line 98 of file fat16.h.

7.3.1.7 #define MBR_SECTOR 0

Definition at line 97 of file fat16.h.

Referenced by InitFat16().

7.3.2 Typedef Documentation

7.3.2.1 typedef struct afile File

7.3.3 Function Documentation

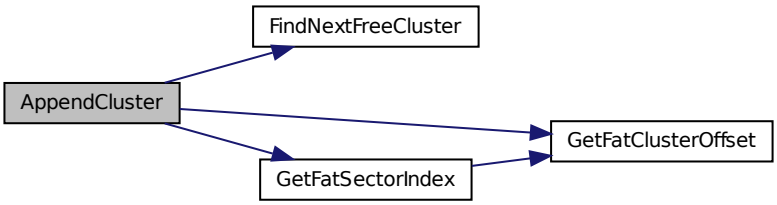
7.3.3.1 unsigned char AppendCluster (File * *file*)

Definition at line 614 of file fat16.c.

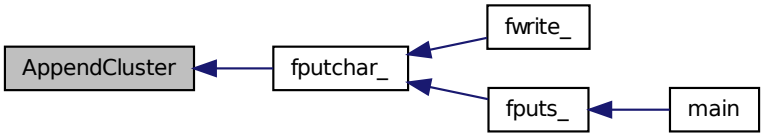
References afile::cluster_pointer, FileAllocationTable, FileBuffer, FindNextFreeCluster(), FirstDataCluster, GetFatClusterOffset(), GetFatSectorIndex(), FatEntry::next_cluster, SDC_GetSector, SDC_PutSector, and SectorsPerCluster.

Referenced by fputchar_().

Here is the call graph for this function:



Here is the caller graph for this function:



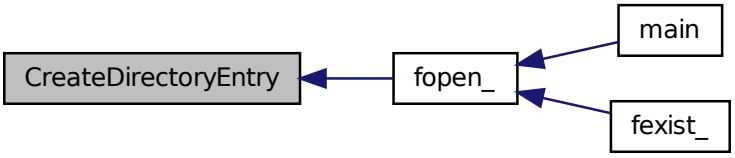
7.3.3.2 unsigned char CreateDirectoryEntry (unsigned char * *fname,* unsigned int *cluster,* File * *file,* unsigned char *attrib*)

Definition at line 687 of file fat16.c.

References DirEntry::attribute, afile::directory_index, afile::directory_sector, FileBuffer, DirEntry::name, PossibleRootEntries, RootDirectory, SDC_GetSector, SDC_PutSector, DirEntry::size, and DirEntry::startcluster.

Referenced by fopen_().

Here is the caller graph for this function:



7.3.3.3 void fclose_ (File * *file*)

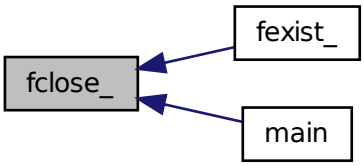
Definition at line 195 of file fat16.c.

References afile::attribute, afile::byte_index, afile::cluster_pointer, afile::directory_index, afile::directory_sector, fflush_(), afile::fileposition, afile::filesize, afile::mode, afile::sector_in_buffer, afile::sector_index, and afile::start_cluster.

Referenced by fexist_(), and main().

Here is the call graph for this function:

Here is the caller graph for this function:

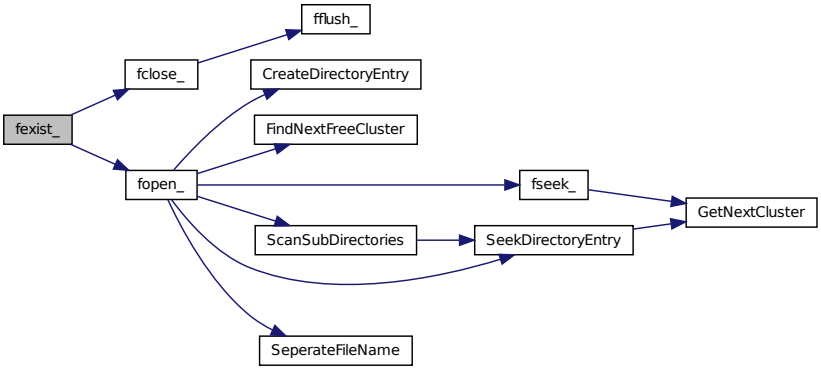


7.3.3.4 unsigned char fexist_ (unsigned char * *fname,* File * *file*)

Definition at line 505 of file fat16.c.

References fclose_(), and fopen_().

Here is the call graph for this function:



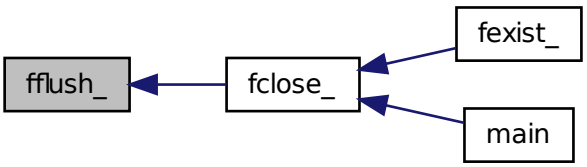
7.3.3.5 int fflush_ (File * *file*)

Definition at line 159 of file fat16.c.

References afile::byte_index, afile::cluster_pointer, DirEntry::date, afile::directory_index, afile::directory_sector, FileBuffer, afile::filesize, afile::mode, SDC_GetSector, SDC_PutSector, afile::sector_index, DirEntry::size, and Dir↩ Entry::time.

Referenced by fclose_().

Here is the caller graph for this function:



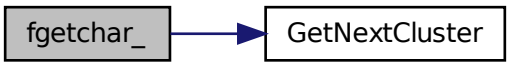
7.3.3.6 int fgetchar_ (File * *file*)

Definition at line 356 of file fat16.c.

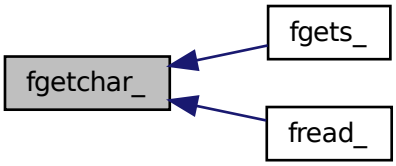
References afile::byte_index, afile::cluster_pointer, FileBuffer, afile::filesize, GetNextCluster(), SDC_GetSector, afile::sector_in_buffer, afile::sector_index, and SectorsPerCluster.

Referenced by fgets_(), and fread_().

Here is the call graph for this function:



Here is the caller graph for this function:

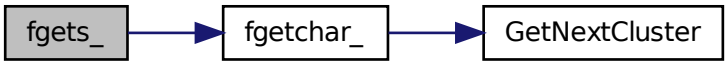


7.3.3.7 char* fgets_ (char * *s*, int *count*, File * *file*)

Definition at line 463 of file fat16.c.

References fgetchar_().

Here is the call graph for this function:



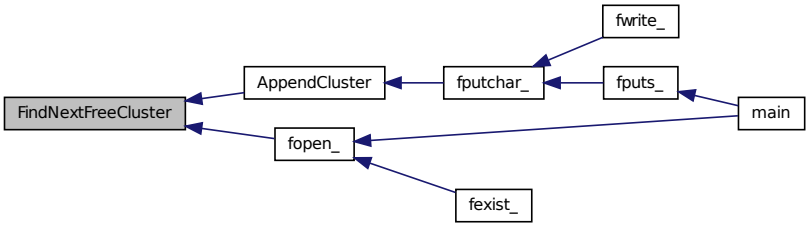
7.3.3.8 unsigned int FindNextFreeCluster (File * *file*)

Definition at line 574 of file fat16.c.

References FileAllocationTable, FileBuffer, FatEntry::next_cluster, SDC_GetSector, SDC_PutSector, afile::sector_in_buffer, and SectorsPerFat.

Referenced by AppendCluster(), and fopen_().

Here is the caller graph for this function:



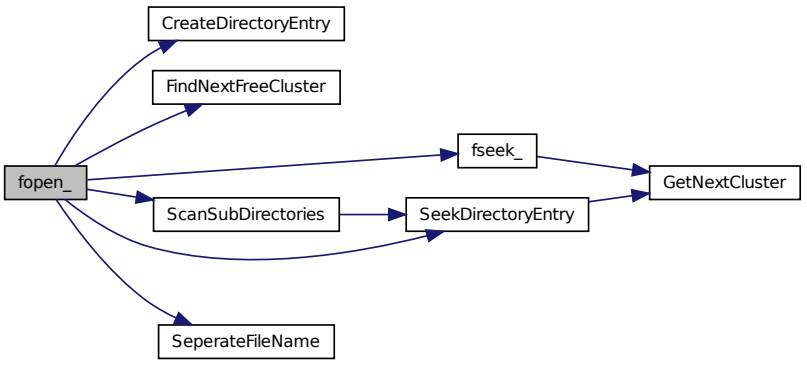
7.3.3.9 unsigned char fopen_ (unsigned char * *fname,* char *mode,* File * *file*)

Definition at line 97 of file fat16.c.

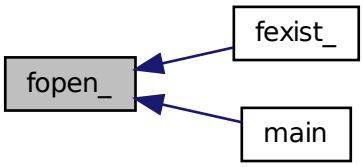
References _FILE, afile::attribute, afile::byte_index, afile::cluster_pointer, CreateDirectoryEntry(), afile::directory_index, afile::directory_sector, afile::fileposition, afile::filesize, FindNextFreeCluster(), FirstDataCluster, fseek_(), afile::mode, ScanSubDirectories(), afile::sector_in_buffer, afile::sector_index, SectorsPerCluster, SeekDirectoryEntry(), SeperateFileName(), and afile::start_cluster.

Referenced by fexist_(), and main().

Here is the call graph for this function:



Here is the caller graph for this function:



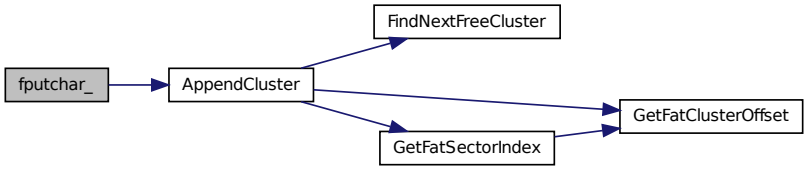
7.3.3.10 unsigned char fputchar_ (File * *file*, char *c*)

Definition at line 400 of file fat16.c.

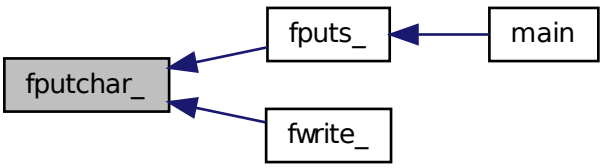
References AppendCluster(), afile::byte_index, afile::cluster_pointer, FileBuffer, afile::fileposition, afile::filesize, SDC_PutSector, afile::sector_index, and SectorsPerCluster.

Referenced by fputs_(), and fwrite_().

Here is the call graph for this function:



Here is the caller graph for this function:



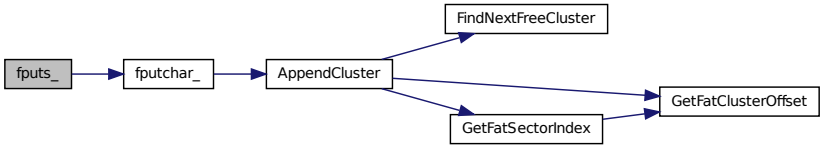
7.3.3.11 unsigned char fputs_ (File * *file*, char * *string*)

Definition at line 443 of file fat16.c.

References fputchar_().

Referenced by main().

Here is the call graph for this function:



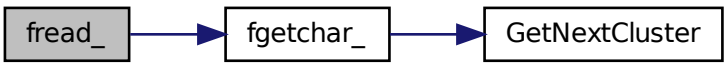
Here is the caller graph for this function:

7.3.3.12 unsigned long fread_ (void * *buffer*, unsigned long *size*, unsigned long *count*, File * *file*)

Definition at line 221 of file fat16.c.

References fgetchar_(), and DirEntry::size.

Here is the call graph for this function:



7.3.3.13 int fseek_ (File * *file*, long *offset*, int *origin*)

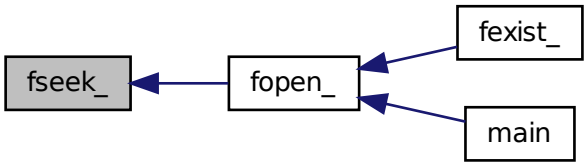
Definition at line 286 of file fat16.c.

References afile::byte_index, afile::cluster_pointer, FileBuffer, afile::fileposition, afile::filesize, GetNextCluster(), SDC_GetSector, afile::sector_index, SectorsPerCluster, and afile::start_cluster.

Referenced by fopen_().

Here is the call graph for this function:

Here is the caller graph for this function:

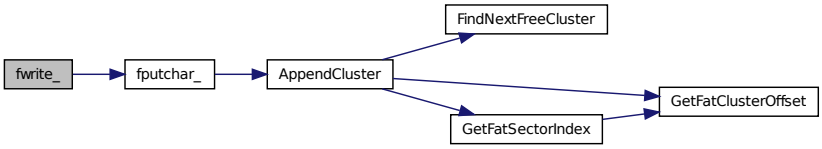


7.3.3.14 unsigned long fwrite_ (void * *buffer,* unsigned long *size,* unsigned long *count,* File * *file*)

Definition at line 254 of file fat16.c.

References fputchar_(), and DirEntry::size.

Here is the call graph for this function:



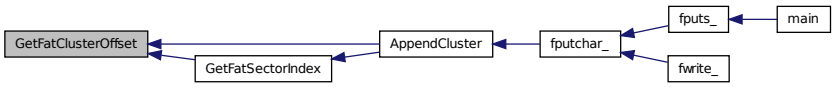
7.3.3.15 unsigned int GetFatClusterOffset (File * *file*)

Definition at line 646 of file fat16.c.

References afile::cluster_pointer, FirstDataCluster, and SectorsPerCluster.

Referenced by AppendCluster(), and GetFatSectorIndex().

Here is the caller graph for this function:



7.3.3.16 unsigned int GetFatSectorIndex (File * *file*)

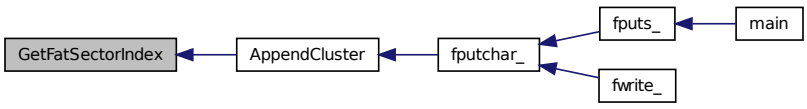
Definition at line 664 of file fat16.c.

References GetFatClusterOffset().

Referenced by AppendCluster().

Here is the call graph for this function:

Here is the caller graph for this function:



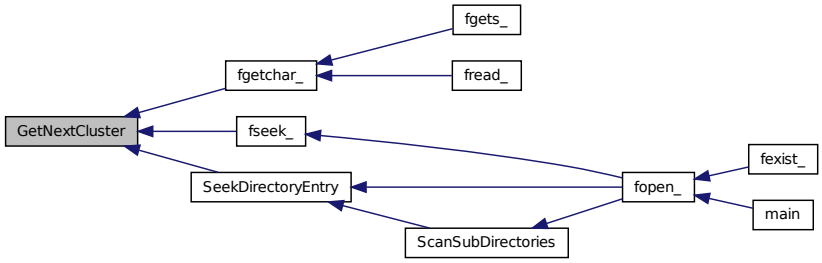
7.3.3.17 unsigned int GetNextCluster (File * *file*)

Definition at line 526 of file fat16.c.

References afile::cluster_pointer, FileAllocationTable, FileBuffer, FirstDataCluster, RootDirectory, SDC_GetSector, afile::sector_in_buffer, and SectorsPerCluster.

Referenced by fgetchar_(), fseek_(), and SeekDirectoryEntry().

Here is the caller graph for this function:



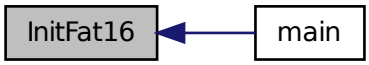
7.3.3.18 unsigned char InitFat16 (void)

Definition at line 71 of file fat16.c.

References FatCopies, FileAllocationTable, FileBuffer, FirstDataCluster, FirstPartitionSector, MBR_SECTOR, PossibleRootEntries, ReservedSectors, RootDirectory, SDC_GetSector, SectorsPerCluster, and SectorsPerFat.

Referenced by main().

Here is the caller graph for this function:



7.3.3.19 int rename (char * *oldname,* char * *newname*)

7.3.3.20 unsigned char ScanSubDirectories (unsigned char * *fname,* File * *file*)

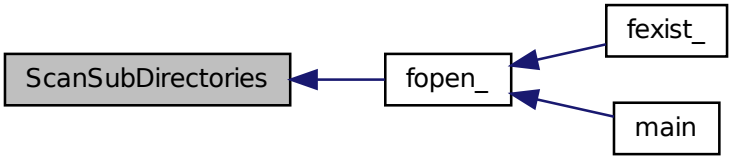
Definition at line 790 of file fat16.c.

References _DIRECTORY, _FILE, afile::attribute, afile::cluster_pointer, RootDirectory, and SeekDirectoryEntry().

Referenced by fopen_().

Here is the call graph for this function:

Here is the caller graph for this function:



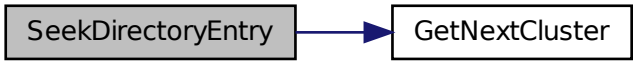
7.3.3.21 unsigned char SeekDirectoryEntry (unsigned char * *fname,* File * *file*)

Definition at line 736 of file fat16.c.

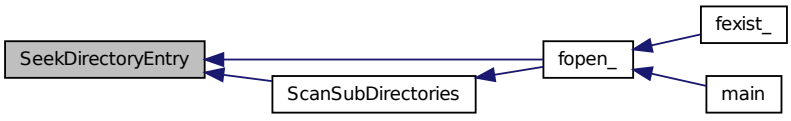
References afile::attribute, DirEntry::attribute, afile::cluster_pointer, afile::directory_index, afile::directory_sector, FileBuffer, afile::filesize, FirstDataCluster, GetNextCluster(), DirEntry::name, SDC_GetSector, SectorsPerCluster, DirEntry::size, afile::start_cluster, and DirEntry::startcluster.

Referenced by fopen_(), and ScanSubDirectories().

Here is the call graph for this function:



Here is the caller graph for this function:

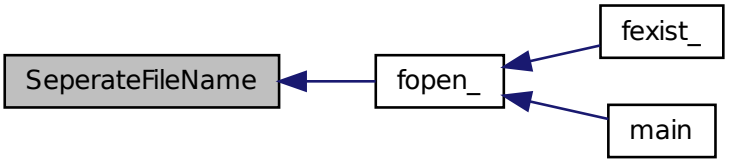


7.3.3.22 void SeperateFileName (unsigned char * *fname,* unsigned char * *name*)

Definition at line 854 of file fat16.c.

Referenced by fopen_().

Here is the caller graph for this function:



7.3.4 Variable Documentation

7.3.4.1 unsigned char FileBuffer[512]

Definition at line 55 of file fat16.c.

Referenced by AppendCluster(), CreateDirectoryEntry(), fflush_(), fgetchar_(), FindNextFreeCluster(), fputchar_(), fseek_(), GetNextCluster(), InitFat16(), and SeekDirectoryEntry().

7.3.4.2 File myfile

7.3.4.3 unsigned char SectorsPerCluster

Definition at line 113 of file fat16.h.

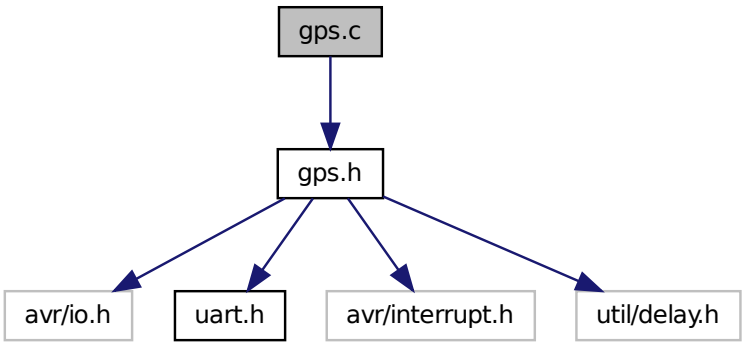
Referenced by AppendCluster(), fgetchar_(), fopen_(), fputchar_(), fseek_(), GetFatClusterOffset(), GetNext↩ Cluster(), InitFat16(), and SeekDirectoryEntry().

7.4 gps.c File Reference

Kommunikation zwischen AVR und Navilock GPS Modul.

```
#include "gps.h"
```

Include dependency graph for gps.c:



Functions

- void gps_init ()

  Routine zur Initialisierung des GPS Moduls.
- char gps_get_char ()

  Gibt sobald verfügbar ein Zeichen vom UART zurück.
- void gps_get_nmea (char *buf, uint8_t bufSize)

  Empfängt einen vollständige NMEA-Befehl und gibt diesen dann zurück.

7.4.1 Detailed Description

Kommunikation zwischen AVR und Navilock GPS Modul.

Author

Martin Matysiak (mail@k621.de)
V. Pippan (webmaster@vpippan.at)

Date

2013-10-18

Version

20131018

Bei Fragen und Verbesserungen wendet euch per EMail an mich.

Datenblätter zum Umgang mit Navilock GPS Modulen:

Datenblatt Navilock U-Blox
NMEA Reference Manual

This file is part of ATmega8 GPS Logger.

ATmega8 GPS Logger is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or any later version.

ATmega8 GPS Logger is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with ATtiny2313 Cessna Lichtsteuerung. If not, see
www.gnu.org/licenses.

Definition in file gps.c.

7.4.2 Function Documentation

7.4.2.1 char gps_get_char (void)

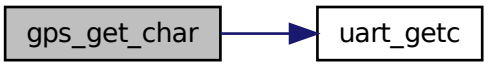
Gibt sobald verfügbar ein Zeichen vom UART zurück.

Definition at line 53 of file gps.c.

References UART_BUFFER_OVERFLOW, uart_getc(), and UART_NO_DATA.

Referenced by gps_get_nmea().

Here is the call graph for this function:



Here is the caller graph for this function:

7.4.2.2 void gps_get_nmea (char * *buf*, uint8_t *bufSize*)

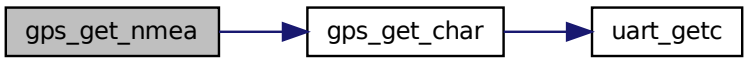
Empfängt einen vollständige NMEA-Befehl und gibt diesen dann zurück.

Definition at line 69 of file gps.c.

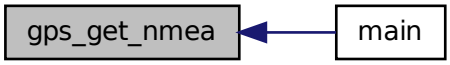
References gps_get_char().

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:



7.4.2.3 void gps_init (void)

Routine zur Initialisierung des GPS Moduls.

Definition at line 41 of file gps.c.

References F_CPU, GPS_BAUD, UART_BAUD_SELECT, and uart_init().

Referenced by main().

Here is the call graph for this function:

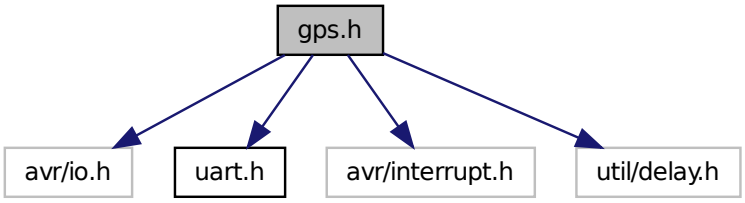
Here is the caller graph for this function:

7.5 gps.h File Reference

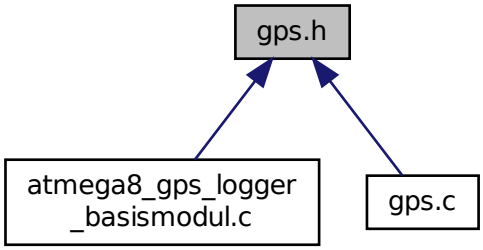
Include File für gps.c.

```
#include <avr/io.h>
#include "uart.h"
#include <avr/interrupt.h>
#include <util/delay.h>
```

Include dependency graph for gps.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define GPS_BAUD 4800

Functions

- void gps_init (void)

  Routine zur Initialisierung des GPS Moduls.
- char gps_get_char (void)

  Gibt sobald verfügbar ein Zeichen vom UART zurück.
- void gps_get_nmea (char *buf, uint8_t bufSize)

  Empfängt einen vollständige NMEA-Befehl und gibt diesen dann zurück.

7.5.1 Detailed Description

Include File für gps.c.

Author

Martin Matysiak (mail@k621.de)
Pippan Vincent (webmaster@vpippan.at)

Date

2013-10-18

Version

20131018

Definition in file gps.h.

7.5.2 Macro Definition Documentation

7.5.2.1 #define GPS_BAUD 4800

Definition at line 18 of file gps.h.

Referenced by gps_init().

7.5.3 Function Documentation

7.5.3.1 char gps_get_char (void)

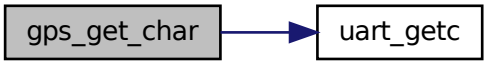
Gibt sobald verfügbar ein Zeichen vom UART zurück.

Definition at line 53 of file gps.c.

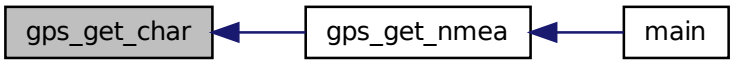
References UART_BUFFER_OVERFLOW, uart_getc(), and UART_NO_DATA.

Referenced by gps_get_nmea().

Here is the call graph for this function:



Here is the caller graph for this function:



7.5.3.2 void gps_get_nmea (char * *buf*, uint8_t *bufSize*)

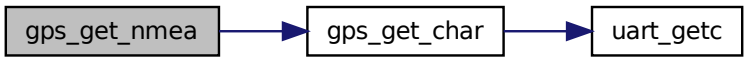
Empfängt einen vollständige NMEA-Befehl und gibt diesen dann zurück.

Definition at line 69 of file gps.c.

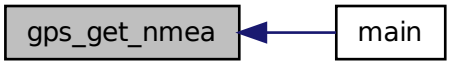
References gps_get_char().

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:



7.5.3.3 void gps_init (void)

Routine zur Initialisierung des GPS Moduls.

Definition at line 41 of file gps.c.

References F_CPU, GPS_BAUD, UART_BAUD_SELECT, and uart_init().

Referenced by main().

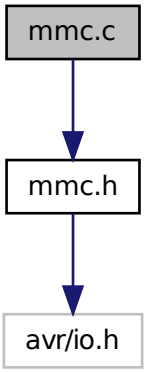
Here is the call graph for this function:

Here is the caller graph for this function:

7.6 mmc.c File Reference

```
#include "mmc.h"
```

Include dependency graph for mmc.c:



Functions

- unsigned char mmc_init ()
- unsigned char mmc_write_command (unsigned char ∗cmd)
- unsigned char mmc_read_byte (void)
- void mmc_write_byte (unsigned char Byte)
- unsigned char mmc_write_sector (unsigned long addr, unsigned char ∗Buffer)
- void mmc_read_block (unsigned char ∗cmd, unsigned char ∗Buffer, unsigned int Bytes)
- unsigned char mmc_read_sector (unsigned long addr, unsigned char ∗Buffer)
- unsigned char mmc_read_cid (unsigned char ∗Buffer)
- unsigned char mmc_read_csd (unsigned char ∗Buffer)

7.6.1 Function Documentation

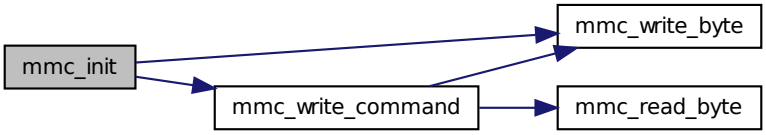
7.6.1.1 unsigned char mmc_init (void)

Definition at line 32 of file mmc.c.

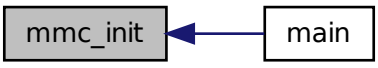
References MMC_Direction_REG, MMC_Disable, MMC_Write, mmc_write_byte(), mmc_write_command(), and nop.

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:



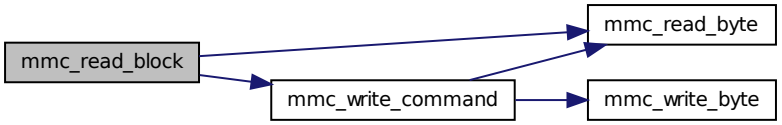
7.6.1.2 void mmc_read_block (unsigned char ∗ *cmd,* unsigned char ∗ *Buffer,* unsigned int *Bytes*)

Definition at line 254 of file mmc.c.

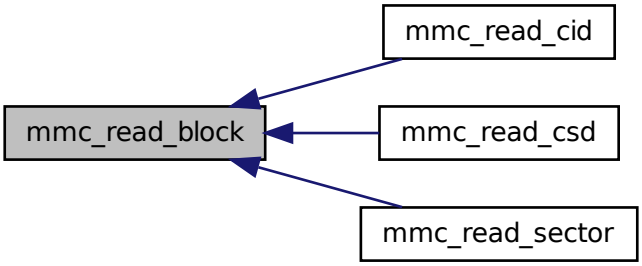
References MMC_Disable, mmc_read_byte(), and mmc_write_command().

Referenced by mmc_read_cid(), mmc_read_csd(), and mmc_read_sector().

Here is the call graph for this function:



Here is the caller graph for this function:



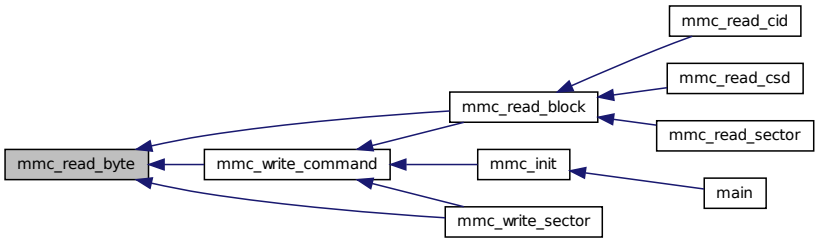
7.6.1.3 unsigned char mmc_read_byte (void)

Definition at line 135 of file mmc.c.

References MMC_Read, and MMC_Write.

Referenced by mmc_read_block(), mmc_write_command(), and mmc_write_sector().

Here is the caller graph for this function:

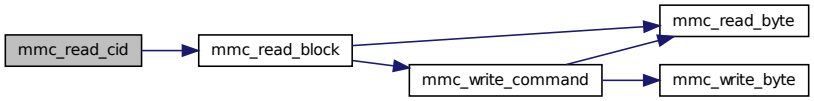


7.6.1.4 unsigned char mmc_read_cid (unsigned char * *Buffer*)

Definition at line 309 of file mmc.c.

References mmc_read_block().

Here is the call graph for this function:

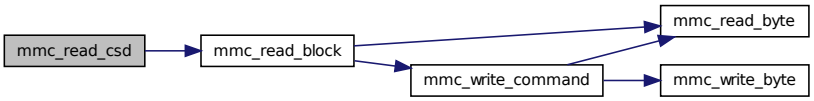


7.6.1.5 unsigned char mmc_read_csd (unsigned char * *Buffer*)

Definition at line 322 of file mmc.c.

References mmc_read_block().

Here is the call graph for this function:

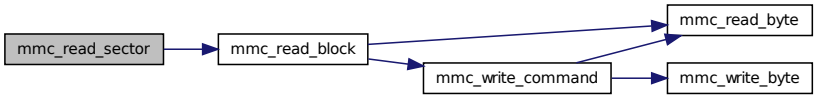


7.6.1.6 unsigned char mmc_read_sector (unsigned long *addr,* unsigned char * *Buffer*)

Definition at line 286 of file mmc.c.

References mmc_read_block().

Here is the call graph for this function:



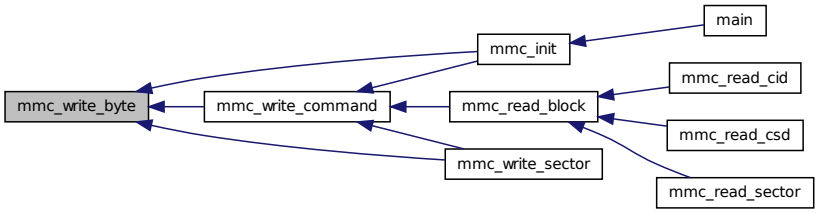
7.6.1.7 void mmc_write_byte (unsigned char *Byte*)

Definition at line 166 of file mmc.c.

References MMC_Write.

Referenced by mmc_init(), mmc_write_command(), and mmc_write_sector().

Here is the caller graph for this function:



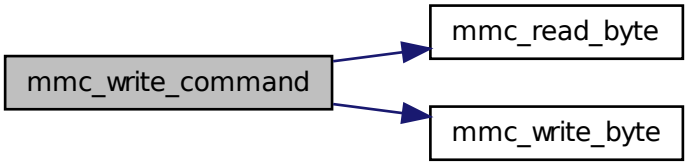
7.6.1.8 unsigned char mmc_write_command (unsigned char * *cmd*)

Definition at line 99 of file mmc.c.

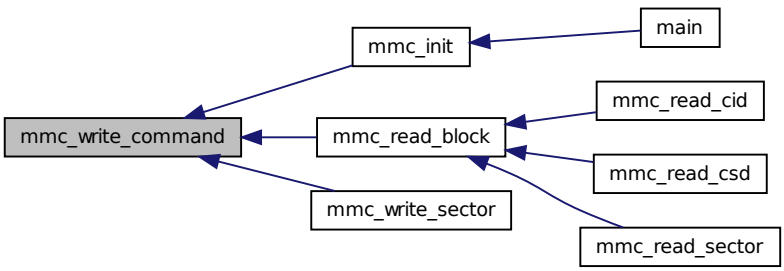
References MMC_Disable, MMC_Enable, mmc_read_byte(), and mmc_write_byte().

Referenced by mmc_init(), mmc_read_block(), and mmc_write_sector().

Here is the call graph for this function:



Here is the caller graph for this function:

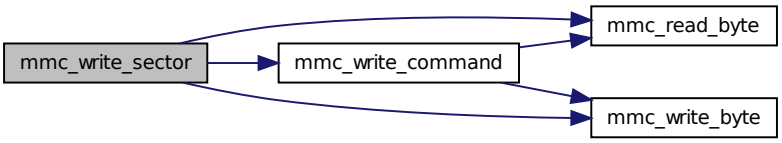


7.6.1.9 unsigned char mmc_write_sector (unsigned long *addr,* unsigned char * *Buffer*)

Definition at line 195 of file mmc.c.

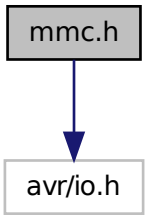
References MMC_Disable, mmc_read_byte(), mmc_write_byte(), and mmc_write_command().

Here is the call graph for this function:

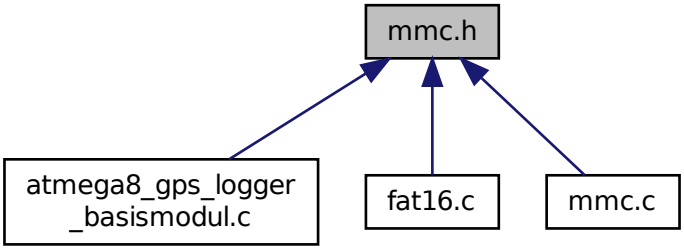


7.7 mmc.h File Reference

`#include <avr/io.h>`
Include dependency graph for mmc.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define SPI_Mode 1
- #define MMC_Write PORTB
- #define MMC_Read PINB
- #define MMC_Direction_REG DDRB
- #define MMC_Disable() MMC_Write|= (1<<MMC_Chip_Select);
- #define MMC_Enable() MMC_Write&=~(1<<MMC_Chip_Select);
- #define SDC_PutSector mmc_write_sector
- #define SDC_GetSector mmc_read_sector
- #define nop() __asm__ __volatile__ ("nop" ::)

Functions

- unsigned char mmc_read_byte (void)
- void mmc_write_byte (unsigned char)
- void mmc_read_block (unsigned char *, unsigned char *, unsigned in)
- unsigned char mmc_init (void)
- unsigned char mmc_read_sector (unsigned long, unsigned char *)
- unsigned char mmc_write_sector (unsigned long, unsigned char *)
- unsigned char mmc_write_command (unsigned char *)
- unsigned char mmc_read_csd (unsigned char *)
- unsigned char mmc_read_cid (unsigned char *)

7.7.1 Macro Definition Documentation

7.7.1.1 #define MMC_Direction_REG DDRB

Definition at line 17 of file mmc.h.

Referenced by mmc_init().

7.7.1.2 #define MMC_Disable( ) MMC_Write|= (1<<MMC_Chip_Select);

Definition at line 63 of file mmc.h.

Referenced by mmc_init(), mmc_read_block(), mmc_write_command(), and mmc_write_sector().

7.7.1.3 #define MMC_Enable( ) MMC_Write&=~(1<<MMC_Chip_Select);

Definition at line 66 of file mmc.h.

Referenced by mmc_write_command().

7.7.1.4 #define MMC_Read PINB

Definition at line 16 of file mmc.h.

Referenced by mmc_read_byte().

7.7.1.5 #define MMC_Write PORTB

Definition at line 15 of file mmc.h.

Referenced by mmc_init(), mmc_read_byte(), and mmc_write_byte().

7.7.1.6 #define nop( ) __asm__ __volatile__ ("nop" ::)

Definition at line 72 of file mmc.h.

Referenced by mmc_init().

7.7.1.7 #define SDC_GetSector mmc_read_sector

Definition at line 70 of file mmc.h.

Referenced by AppendCluster(), CreateDirectoryEntry(), fflush_(), fgetchar_(), FindNextFreeCluster(), fseek_(), GetNextCluster(), InitFat16(), and SeekDirectoryEntry().

7.7.1.8 #define SDC_PutSector mmc_write_sector

Definition at line 68 of file mmc.h.

Referenced by AppendCluster(), CreateDirectoryEntry(), fflush_(), FindNextFreeCluster(), and fputchar_().

7.7.1.9 #define SPI_Mode 1

Definition at line 12 of file mmc.h.

7.7.2 Function Documentation

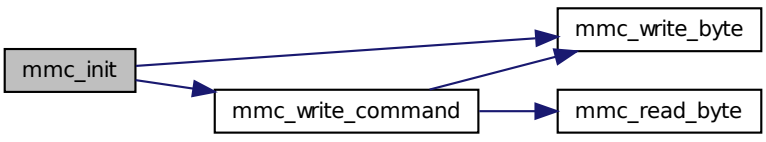
7.7.2.1 unsigned char mmc_init (void)

Definition at line 32 of file mmc.c.

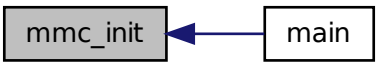
References MMC_Direction_REG, MMC_Disable, MMC_Write, mmc_write_byte(), mmc_write_command(), and nop.

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:



7.7.2.2 void mmc_read_block (unsigned char *, unsigned char *, unsigned *in*)

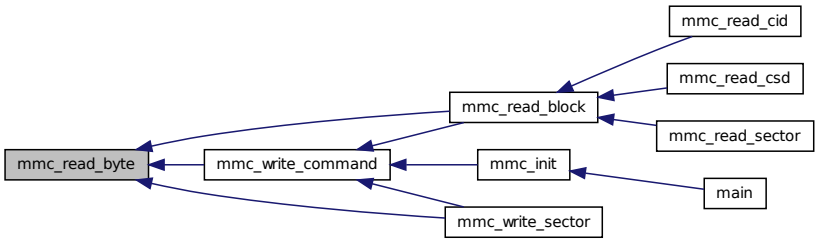
7.7.2.3 unsigned char mmc_read_byte (void)

Definition at line 135 of file mmc.c.

References MMC_Read, and MMC_Write.

Referenced by mmc_read_block(), mmc_write_command(), and mmc_write_sector().

Here is the caller graph for this function:

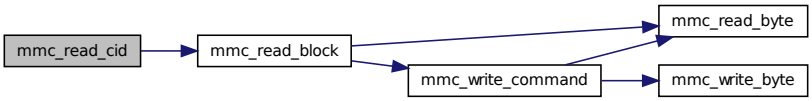


7.7.2.4 unsigned char mmc_read_cid (unsigned char *)

Definition at line 309 of file mmc.c.

References mmc_read_block().

Here is the call graph for this function:

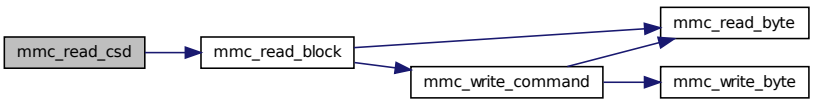


7.7.2.5 unsigned char mmc_read_csd (unsigned char *)

Definition at line 322 of file mmc.c.

References mmc_read_block().

Here is the call graph for this function:

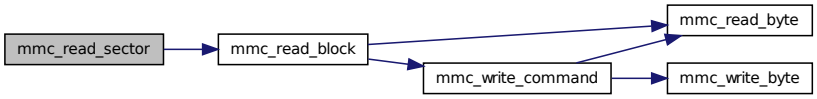


7.7.2.6 unsigned char mmc_read_sector (unsigned *long,* unsigned char *)

Definition at line 286 of file mmc.c.

References mmc_read_block().

Here is the call graph for this function:



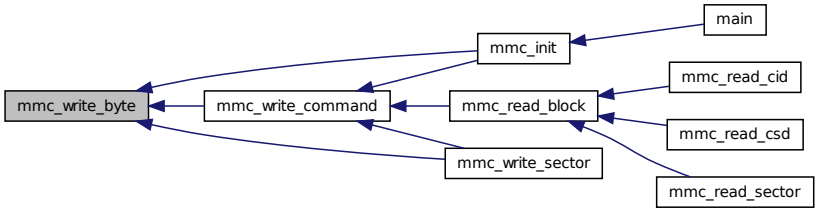
7.7.2.7 void mmc_write_byte (unsigned *char*)

Definition at line 166 of file mmc.c.

References MMC_Write.

Referenced by mmc_init(), mmc_write_command(), and mmc_write_sector().

Here is the caller graph for this function:



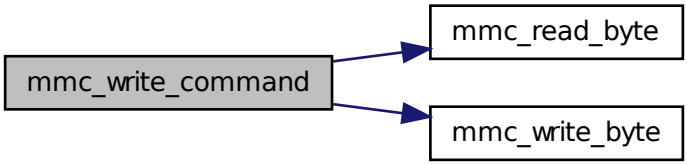
7.7.2.8 unsigned char mmc_write_command (unsigned char *)

Definition at line 99 of file mmc.c.

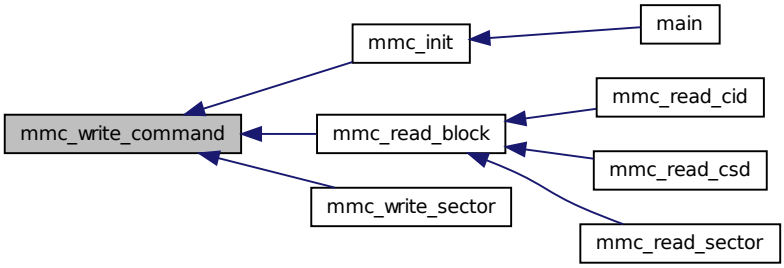
References MMC_Disable, MMC_Enable, mmc_read_byte(), and mmc_write_byte().

Referenced by mmc_init(), mmc_read_block(), and mmc_write_sector().

Here is the call graph for this function:



Here is the caller graph for this function:

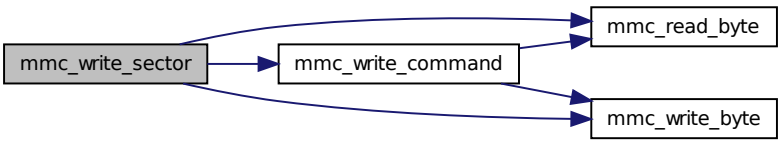


7.7.2.9 unsigned char mmc_write_sector (unsigned *long,* unsigned char *)

Definition at line 195 of file mmc.c.

References MMC_Disable, mmc_read_byte(), mmc_write_byte(), and mmc_write_command().

Here is the call graph for this function:

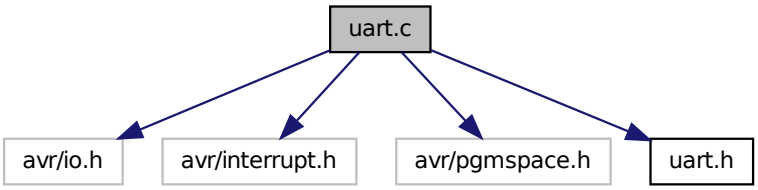


7.8 uart.c File Reference

Interrupt UART library with receive/transmit circular buffers.

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/pgmspace.h>
#include "uart.h"
```

Include dependency graph for uart.c:



Macros

- #define UART_RX_BUFFER_MASK (UART_RX_BUFFER_SIZE - 1)
- #define UART_TX_BUFFER_MASK (UART_TX_BUFFER_SIZE - 1)

Functions

- SIGNAL (UART0_RECEIVE_INTERRUPT)
- SIGNAL (UART0_TRANSMIT_INTERRUPT)
- void uart_init (unsigned int baudrate)

  Initialize UART and set baudrate.
- unsigned int uart_getc (void)

  Get received byte from ringbuffer.
- void uart_putc (unsigned char data)

  Put byte to ringbuffer for transmitting via UART.
- void uart_puts (const char *s)

  Put string to ringbuffer for transmitting via UART.
- void uart_puts_p (const char *progmem_s)

  Put string from program memory to ringbuffer for transmitting via UART.

7.8.1 Detailed Description

Interrupt UART library with receive/transmit circular buffers.

Author

Peter Fleury <pfleury@gmx.ch>
Peter Fleury

Date

2007-07-01

Version

1.6.2.1

Software: AVR-GCC 4.1, AVR Libc 1.4.6 or higher
Hardware: any AVR with built-in UART
License: GNU General Public License

DESCRIPTION:
An interrupt is generated when the UART has finished transmitting or receiving a byte. The interrupt handling routines use circular buffers for buffering received and transmitted data.

The UART_RX_BUFFER_SIZE and UART_TX_BUFFER_SIZE variables define the buffer size in bytes. Note that these variables must be a power of 2.

USAGE:
Refere to the header file uart.h for a description of the routines. See also example test_uart.c.

Note

Based on Atmel Application Note AVR306

Definition in file uart.c.

7.8.2 Macro Definition Documentation

7.8.2.1 #define UART_RX_BUFFER_MASK (UART_RX_BUFFER_SIZE - 1)

constants and macros

size of RX/TX buffers

Definition at line 52 of file uart.c.

Referenced by SIGNAL(), and uart_getc().

7.8.2.2 #define UART_TX_BUFFER_MASK (UART_TX_BUFFER_SIZE - 1)

Definition at line 53 of file uart.c.

Referenced by SIGNAL(), and uart_putc().

7.8.3 Function Documentation

7.8.3.1 SIGNAL (UART0_RECEIVE_INTERRUPT)

UART Receive Complete interrupt called when the UART has received a character read UART status register and UART data register

calculate buffer index

error: receive buffer overflow

store new index

store received data in buffer

Definition at line 248 of file uart.c.

References UART_BUFFER_OVERFLOW, and UART_RX_BUFFER_MASK.

7.8.3.2 SIGNAL (UART0_TRANSMIT_INTERRUPT)

UART Data Register Empty interrupt called when the UART is ready to transmit the next byte calculate and store new buffer index

get one byte from buffer and write it to UART

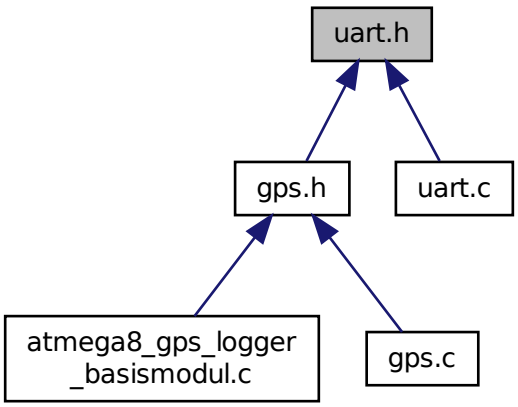
tx buffer empty, disable UDRE interrupt

Definition at line 290 of file uart.c.

References UART_TX_BUFFER_MASK.

7.9 uart.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- #define UART_BAUD_SELECT(baudRate, xtalCpu) ((xtalCpu)/((baudRate)∗16l)-1)

  UART Baudrate Expression.
- #define UART_BAUD_SELECT_DOUBLE_SPEED(baudRate, xtalCpu) (((xtalCpu)/((baudRate)∗8l)-1)|0x8000)

  UART Baudrate Expression for ATmega double speed mode.
- #define UART_RX_BUFFER_SIZE 64
- #define UART_TX_BUFFER_SIZE 64

- #define UART_FRAME_ERROR 0x0800 /* Framing Error by UART */
- #define UART_OVERRUN_ERROR 0x0400 /* Overrun condition by UART */
- #define UART_BUFFER_OVERFLOW 0x0200 /* receive ringbuffer overflow */
- #define UART_NO_DATA 0x0100 /* no receive data available */
- #define uart_puts_P(__s) uart_puts_p(PSTR(__s))

  Macro to automatically put a string constant into program memory.
- #define uart1_puts_P(__s) uart1_puts_p(PSTR(__s))

  Macro to automatically put a string constant into program memory.

Functions

- void uart_init (unsigned int baudrate)

  Initialize UART and set baudrate.
- unsigned int uart_getc (void)

  Get received byte from ringbuffer.
- void uart_putc (unsigned char data)

  Put byte to ringbuffer for transmitting via UART.
- void uart_puts (const char *s)

  Put string to ringbuffer for transmitting via UART.
- void uart_puts_p (const char *progmem_s)

  Put string from program memory to ringbuffer for transmitting via UART.
- void uart1_init (unsigned int baudrate)

  Initialize USART1 (only available on selected ATmegas)
- unsigned int uart1_getc (void)

  Get received byte of USART1 from ringbuffer. (only available on selected ATmega)
- void uart1_putc (unsigned char data)

  Put byte to ringbuffer for transmitting via USART1 (only available on selected ATmega)
- void uart1_puts (const char *s)

  Put string to ringbuffer for transmitting via USART1 (only available on selected ATmega)
- void uart1_puts_p (const char *s)

  Put string from program memory to ringbuffer for transmitting via USART1 (only available on selected ATmega)

Index