CANDrv

Generated by Doxygen 1.8.12

Contents

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 CANPAGEHandler Class Reference

Class for safe handling of the CANPAGE register.

```
#include <CANDrv.h>
```

Public Member Functions

- CANPAGEHandler (uint8_t mob_nr)

  set CANPAGE to the supplied NR
- ∼CANPAGEHandler ()

  recovers the previously saved CANPAGE

3.1.1 Detailed Description

Class for safe handling of the CANPAGE register.

This class is initialized with the mob_nr to switch to. The current CANPAGE is saved an will be restored after destruction of the context

3.1.2 Constructor & Destructor Documentation

3.1.2.1 CANPAGEHandler()

```
CANPAGEHandler::CANPAGEHandler (
            uint8_t mob_nr )
```

set CANPAGE to the supplied NR

Parameters

mob←↩ _nr	the Number of the MOB to switch to

3.1.2.2 ∼CANPAGEHandler()

```
CANPAGEHandler::∼CANPAGEHandler ( )
```

recovers the previously saved CANPAGE

The documentation for this class was generated from the following files:

- CANDrv.h
- CANDrv.cpp

3.2 mob_settings Struct Reference

Settings for a MOB.

```
#include <CANDrv.h>
```

Public Attributes

- uint16_t can_id

    CAN ID.
- uint16_t can_msk

    CAN Mask.
- uint8_t ide

    Extended Message Format.
- uint8_t dlc

    Data Length Coding.
- uint8_t ∗ data

    Pointer to the Data array.

3.2.1 Detailed Description

Settings for a MOB.

3.2.2 Member Data Documentation

3.2.2.1 can_id

```
uint16_t mob_settings::can_id
```

CAN ID.

Input for Transmisson.
Input or Output for Receiving. Depending on the CAN Mask.

3.2.2.2 can_msk

```
uint16_t mob_settings::can_msk
```

CAN Mask.

Mask of the CAN ID for RX.
Input for Receiving, Irrelevant for Transmitting<br>

The Can Message is filtered by a Acceptance filter according to this Mask.
This Element is bit coded. 0 means this bit in the CAN ID is not relevant for Acceptance. 1 means it is relevant.

for example:

Configured CAN ID	Configured CAN MSK	Actual CAN ID	ACCEPTANCE
7E0	3FF	7E0	YES
7E0	3FF	7E8	NO
7E0	3F7	7E8	YES
7E0	000	ANY	YES

3.2.2.3 data

```
uint8_t* mob_settings::data
```

Pointer to the Data array.

The Data arry has to be already allocated.

3.2.2.4 dlc

```
uint8_t mob_settings::dlc
```

Data Length Coding.

Length of the Data. Output for Receiving. Input for Transmission

3.2.2.5 ide

```
uint8_t mob_settings::ide
```

Extended Message Format.

this element is set to 1 if the extended Message Format is used. Input for Transmitting and Receiving

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

- CANDrv.h

3.3 MobConfigElement Struct Reference

configuration object for the FRMMan Settings

```
#include <CANDrv.h>
```

Public Attributes

- mob_purpose op
- mob_settings ms
- void(∗ f)(uint8_t)
- uint32_t timestamp
- void ∗ additionalData

3.3.1 Detailed Description

configuration object for the FRMMan Settings

3.3.2 Member Data Documentation

3.3.2.1 additionalData

```
void* MobConfigElement::additionalData
```

additional Data for usage in an custom ISR

3.3.2.2 f

```
void(* MobConfigElement::f) (uint8_t)
```

ISR routine

3.3.2.3 ms

```
mob_settings MobConfigElement::ms
```

settings struct

3.3.2.4 op

```
mob_purpose MobConfigElement::op
```

purpose of the MOB

3.3.2.5 timestamp

```
uint32_t MobConfigElement::timestamp
```

OUT: timestamp of the Last Interrupt

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

- CANDrv.h

Chapter 4

File Documentation

4.1 CANDrv.cpp File Reference

CANDrv cpp file.

```
#include <avr/io.h>
#include "CANDrv.h"
#include "Arduino.h"
```

Functions

- uint8_t CANDrv_Set_bt (CanBaudrate baudrate)

  set the Timing Parameters for the CAN.
- void CLEAR_RXOK ()

  Clears the interrupt FLAG.
- void CANDrv_ClearAll_MOB (void)

  advanced Function not needed for normal operation
- uint8_t CANDrv_Init (CanBaudrate baudrate)

  Initialize the CAN Driver.
- void getMOBsetup (mob_settings ∗ms)

  advanced Function not needed for normal operation
- void setupMOB (mob_settings ∗ms)

  advanced Function not needed for normal operation
- void receiveData_generic (uint8_t mob_NR)

  generic Receive funtion for an Interrupt Driven MOB.
- void receiveData_generic_restart (uint8_t mob_NR)

  generic Receive funtion for an Interrupt Driven MOB.
- uint8_t CANDrv_FRMMan_Init (MobConfigElement ∗CAN_Config)

  Initialize the FRMMan.
- uint8_t CANDrv_FRMMan_Send_Msg (uint8_t index)

  sends a configured Message
- uint8_t CANDrv_FRMMan_Get_Msg (uint8_t index, mob_settings ∗ms, uint8_t iteration=0)

  get a received Can Message.
- uint8_t CANDrv_FRMMan_Get_Msg (uint8_t index, mob_settings ∗ms)

  get a received Can Message.
- mob_status CANDrv_FRMMan_Get_MSG_State (uint8_t index)
- ISR (CAN_INT_vect)

Variables

- MobConfigElement ∗ __internal_CAN_Config

4.1.1 Detailed Description

CANDrv cpp file.

This File contains the Implementation of the CANDrv and the FRMMan

4.1.2 Function Documentation

4.1.2.1 CANDrv_ClearAll_MOB()

```
CANDrv_ClearAll_MOB (
            void  )
```

advanced Function not needed for normal operation

This function clear all MOB registers. This is used in INI.

4.1.2.2 CANDrv_FRMMan_Get_Msg() [1/2]

```
CANDrv_FRMMan_Get_Msg (
            uint8_t index,
            mob_settings * ms,
            uint8_t iteration = 0 )
```

get a received Can Message.

This function reads a received Can Message from the internal Buffer.
It also retries to read if there were any Changes, because of an Interrupt, receiving a new Message.
In this case the read operation is retried.

!ONLY USED INTERNALLY!

Parameters

index	the index of the mob to read
ms	OUTPUT Pointer to a message settings object, where the data is written.
iteration	Helper argument to limit the number of tries

Returns

1 on success. 0 in case of an error.

4.1.2.3 CANDrv_FRMMan_Get_Msg() [2/2]

```
CANDrv_FRMMan_Get_Msg (
              uint8_t index,
              mob_settings * ms )
```

get a received Can Message.

This function reads a received Can Message from the internal Buffer.
It also retries to read if there were any Changes, because of an Interrupt, receiving a new Message.
In this case the read operation is retried.

Parameters

index	the index of the mob to read
ms	OUTPUT Pointer to a message settings object, where the data is written.

Returns

1 on success. 0 in case of an error.

4.1.2.4 CANDrv_FRMMan_Get_MSG_State()

```
mob_status CANDrv_FRMMan_Get_MSG_State (
              uint8_t index )
```

4.1.2.5 CANDrv_FRMMan_Init()

```
CANDrv_FRMMan_Init (
              MobConfigElement * CAN_Config )
```

Initialize the FRMMan.

This function initializes the FRMMan with the given CAN_Config.

Example configuration:

```
  //Make sure all these variables are global an not in a scoped context, like a funtion.
  uint8_t data_130[8];
  uint8_t data_7e8[8];
  uint8_t data_7e0[8];
  MobConfigElement CAN_Config[] =
  {
    {TX_DATA_SW_DRIVEN,{0x7e0,0x000,0,8,(uint8_t*)&data_7e0}},//This Messsage can be sent
       with CANDrv_FRMMan_Send_Msg(0);
    {RX_DATA_INTERRUPT_DRIVEN,{0x130,0x3FF,0,8,(uint8_t*)&data_130},&
      receiveData_generic_restart,0,0},//This Message will be received with an
       Interrupt. It will be directly enabled again
    {RX_DATA_INTERRUPT_DRIVEN,{0x7e8,0x3FF,0,8,(uint8_t*)&data_7e8},&
      receiveData_generic,0,0},//This Message will be received with an Interrupt. It will be
       only received once!
    {UNUSED},//unused MOBs
    {UNUSED},
    {UNUSED}
  };

  void setup() {
  data_7e0[0] = 0x02; //Initialize the Message to send.
  data_7e0[1] = 0x01;
  data_7e0[2] = 0x05;
  data_7e0[3] = 0x33;
  data_7e0[4] = 0x44;
  data_7e0[5] = 0x55;
  data_7e0[6] = 0x66;
  data_7e0[7] = 0x77;
  CANDrv_Init(CAN_500k); //init CANDrv
  CANDrv_FRMMan_Init(CAN_Config);//init FRMMan
  sei();//enable interrupts. This is neccessary for the Interrupt driven receives
}
```

Parameters

CAN_Config	The CAN_Config to use. For examples see funtion description.

Returns

1 on success. 0 in case of an error.

4.1.2.6 CANDrv_FRMMan_Send_Msg()

```
CANDrv_FRMMan_Send_Msg (
             uint8_t index )
```

sends a configured Message

This function sends a predefined Message.

Parameters

index	the index of the Message to send

Returns

1 on success. 0 in case of an error.

4.1.2.7 CANDrv_Init()

```
uint8_t CANDrv_Init (
             CanBaudrate baudrate )
```

Initialize the CAN Driver.

Parameters

baudrate	the baudrate to set

Returns

1 on success. 0 in case of an error.

4.1.2.8 CANDrv_Set_bt()

```
uint8_t CANDrv_Set_bt (
             CanBaudrate baudrate )  [inline]
```

set the Timing Parameters for the CAN.

the Values of the Registers CANBT1-3 are also defined in this file.

!ONLY USED INTERNALLY!

Parameters

baudrate	the baudrate to set.

Returns

1 on success. 0 in case of an error.

4.1.2.9 CLEAR_RXOK()

```
void CLEAR_RXOK ( )  [inline]
```

Clears the interrupt FLAG.

4.1.2.10 getMOBsetup()

```
void getMOBsetup (
             mob_settings * ms )
```

advanced Function not needed for normal operation

This function reads the Settings from the currently selected MOB. To select a MOB see CANPAGEHandler

See also

CANPAGEHandler

Parameters

ms	pointer to the MOB settings

4.1.2.11 ISR()

```
ISR (
             CAN_INT_vect  )
```

4.1.2.12 receiveData_generic()

```
receiveData_generic (
             uint8_t mob_NR )
```

generic Receive funtion for an Interrupt Driven MOB.

This function only receives this Message once!

Parameters

mob_NR	the MOB Number given by the ISR

4.1.2.13 receiveData_generic_restart()

```
receiveData_generic_restart (
              uint8_t mob_NR )
```

generic Receive funtion for an Interrupt Driven MOB.

This function will activate the Interrupt again.

Parameters

mob_NR	the MOB Number given by the ISR

4.1.2.14 setupMOB()

```
setupMOB (
              mob_settings * ms )
```

advanced Function not needed for normal operation

This function writes the given Settings to the currently selected MOB. To select a MOB see CANPAGEHandler

See also

CANPAGEHandler

Parameters

ms	pointer to the MOB settings

4.1.3 Variable Documentation

4.1.3.1 __internal_CAN_Config

```
MobConfigElement* __internal_CAN_Config
```

4.2 CANDrv.h File Reference

CANDrv header File.

Classes

- struct mob_settings

  Settings for a MOB.
- struct MobConfigElement

  configuration object for the FRMMan Settings
- class CANPAGEHandler

  Class for safe handling of the CANPAGE register.

Enumerations

- enum CanBaudrate { CAN_500k, CAN_800k }

  Baudrate Enumerator for the Baudrate.
- enum mob_operation { TX_DATA = 0x01, RX_DATA = 0x02, DISABLED = 0x00 }

  MOB Operation Enumerator for the operation while manually using a MOB.
- enum mob_purpose {
  TX_DATA_SW_DRIVEN = 0x10, RX_DATA_SW_DRIVEN = 0x20, MULTIPURPOSE = 0x30, RX_DATA_↩
  INTERRUPT_DRIVEN = 0x62,
  RX_DATA_INTERRUPT_DRIVEN_INACTIVE = 0x60, TX_DATA_INTERRUPT_ACTIVE = 0x50, UNUSED
  = 0x00 }

  MOB Purpose Enumerator for the Purpose of the MOB when configuring the FRMMan.
- enum mob_status {
  TX_PENDING = 0x03, TX_OK = 0x05, RX_PENDING = 0x02, RX_OK = 0x04,
  RX_ERROR = 0xFE, TX_ERROR = 0xFF, IDLE = 0x00 }

  MOB Status Enumerator for the Status of the MOB.

Functions

- uint8_t CANDrv_Init (CanBaudrate baudrate)

  Initialize the CAN Driver.
- uint8_t CANDrv_FRMMan_Get_Msg (uint8_t index, mob_settings ∗ms)

  get a received Can Message.
- mob_status CANDrv_FRMMan_Get_MSG_State (uint8_t index)
- uint8_t CANDrv_FRMMan_Send_Msg (uint8_t index)

  sends a configured Message
- uint8_t CANDrv_FRMMan_Init (MobConfigElement ∗CAN_Config)

  Initialize the FRMMan.
- void receiveData_generic (uint8_t mob_NR)

  generic Receive funtion for an Interrupt Driven MOB.
- void receiveData_generic_restart (uint8_t mob_NR)

  generic Receive funtion for an Interrupt Driven MOB.
- void CANDrv_ClearAll_MOB (void)

  advanced Function not needed for normal operation
- void getMOBsetup (mob_settings ∗ms)

  advanced Function not needed for normal operation
- void setMOB_Operation (mob_operation mo)

  set the Operation of the current MOB

4.2.1 Detailed Description

CANDrv header File.

This File describes the Interface of the CANDrv and the FRMMan

4.2.2 Enumeration Type Documentation

4.2.2.1 CanBaudrate

```
enum CanBaudrate
```

Baudrate Enumerator for the Baudrate.

Enumerator

CAN_500k	500 Kbaud
CAN_800k	800 Kbaud

4.2.2.2 mob_operation

```
enum mob_operation
```

MOB Operation Enumerator for the operation while manually using a MOB.

Enumerator

TX_DATA	Transmit DATA
RX_DATA	Receive DATA
DISABLED	DISABLED

4.2.2.3 mob_purpose

```
enum mob_purpose
```

MOB Purpose Enumerator for the Purpose of the MOB when configuring the FRMMan.

Enumerator

TX_DATA_SW_DRIVEN	TX triggered by SW
RX_DATA_SW_DRIVEN	RX triggered by SW
MULTIPURPOSE	RX and TX triggered by SW
RX_DATA_INTERRUPT_DRIVEN	RX triggered by Interrupt. Automatically activated at Startup
RX_DATA_INTERRUPT_DRIVEN_INACTIVE	RX triggered by Interrupt. Automatically deactivated at Startup
TX_DATA_INTERRUPT_ACTIVE	TX triggered by SW. Will cause an Interrupt on completion
UNUSED	MOB is unused

4.2.2.4 mob_status

```
enum mob_status
```

MOB Status Enumerator for the Status of the MOB.

Enumerator

TX_PENDING	TX PENDING
TX_OK	TX SUCCESSFULL
RX_PENDING	RX PENDING
RX_OK	RX SUCCESSFULL
RX_ERROR	ERROR
TX_ERROR	ERROR
IDLE	MOB is unused

4.2.3 Function Documentation

4.2.3.1 CANDrv_ClearAll_MOB()

```
void CANDrv_ClearAll_MOB (
             void )
```

advanced Function not needed for normal operation

This function clear all MOB registers. This is used in INI.

4.2.3.2 CANDrv_FRMMan_Get_Msg()

```
uint8_t CANDrv_FRMMan_Get_Msg (
             uint8_t index,
             mob_settings * ms )
```

get a received Can Message.

This function reads a received Can Message from the internal Buffer.
It also retries to read if there were any Changes, because of an Interrupt, receiving a new Message.
In this case the read operation is retried.

Parameters

index	the index of the mob to read
ms	OUTPUT Pointer to a message settings object, where the data is written.

Returns

1 on success. 0 in case of an error.

4.2.3.3 CANDrv_FRMMan_Get_MSG_State()

```
mob_status CANDrv_FRMMan_Get_MSG_State (
              uint8_t index )
```

4.2.3.4 CANDrv_FRMMan_Init()

```
uint8_t CANDrv_FRMMan_Init (
              MobConfigElement * CAN_Config )
```

Initialize the FRMMan.

This function initializes the FRMMan with the given CAN_Config.

Example configuration:

```
  //Make sure all these variables are global an not in a scoped context, like a funtion.
  uint8_t data_130[8];
  uint8_t data_7e8[8];
  uint8_t data_7e0[8];
  MobConfigElement CAN_Config[] =
  {
    {TX_DATA_SW_DRIVEN,{0x7e0,0x000,0,8,(uint8_t*)&data_7e0}},//This Messsage can be sent
       with CANDrv_FRMMan_Send_Msg(0);
    {RX_DATA_INTERRUPT_DRIVEN,{0x130,0x3FF,0,8,(uint8_t*)&data_130},&
      receiveData_generic_restart,0,0},//This Message will be received with an
       Interrupt. It will be directly enabled again
    {RX_DATA_INTERRUPT_DRIVEN,{0x7e8,0x3FF,0,8,(uint8_t*)&data_7e8},&
      receiveData_generic,0,0},//This Message will be received with an Interrupt. It will be
       only received once!
    {UNUSED},//unused MOBs
    {UNUSED},
    {UNUSED}
  };

  void setup() {
  data_7e0[0] = 0x02; //Initialize the Message to send.
  data_7e0[1] = 0x01;
  data_7e0[2] = 0x05;
  data_7e0[3] = 0x33;
  data_7e0[4] = 0x44;
  data_7e0[5] = 0x55;
  data_7e0[6] = 0x66;
  data_7e0[7] = 0x77;
  CANDrv_Init(CAN_500k); //init CANDrv
  CANDrv_FRMMan_Init(CAN_Config);//init FRMMan
  sei();//enable interrupts. This is neccessary for the Interrupt driven receives
}
```

Parameters

CAN_Config	The CAN_Config to use. For examples see funtion description.

Returns

1 on success. 0 in case of an error.

4.2.3.5 CANDrv_FRMMan_Send_Msg()

```
uint8_t CANDrv_FRMMan_Send_Msg (
              uint8_t index )
```

sends a configured Message

This function sends a predefined Message.

Parameters

index	the index of the Message to send

Returns

1 on success. 0 in case of an error.

4.2.3.6 CANDrv_Init()

```
uint8_t CANDrv_Init (
             CanBaudrate baudrate )
```

Initialize the CAN Driver.

Parameters

baudrate	the baudrate to set

Returns

1 on success. 0 in case of an error.

4.2.3.7 getMOBsetup()

```
void getMOBsetup (
             mob_settings * ms )
```

advanced Function not needed for normal operation

This function reads the Settings from the currently selected MOB. To select a MOB see CANPAGEHandler

See also

CANPAGEHandler

Parameters

ms	pointer to the MOB settings

4.2.3.8 receiveData_generic()

```
void receiveData_generic (
             uint8_t mob_NR )
```

generic Receive funtion for an Interrupt Driven MOB.

This function only receives this Message once!

Parameters

mob_NR	the MOB Number given by the ISR

4.2.3.9 receiveData_generic_restart()

```
void receiveData_generic_restart (
            uint8_t mob_NR )
```

generic Receive funtion for an Interrupt Driven MOB.

This function will activate the Interrupt again.

Parameters

mob_NR	the MOB Number given by the ISR

4.2.3.10 setMOB_Operation()

```
setMOB_Operation (
            mob_operation mo )  [inline]
```

set the Operation of the current MOB

This function sets the Operation of the currently selected MOB To select a MOB see CANPAGEHandler

See also

CANPAGEHandler

Parameters

mo	the operation mode to set the MOB to

Index