Time multiplex ACuc-rf101

V2.1B

Generated by Doxygen 1.8.11

Contents

Chapter 1

ADuc-RF-101 Time multiplex

1.1 Introduction

Program is using time multiplex for transmiting data from optionaly number of Slaves to Maseter device via R↩ F-channel with checking data integrity using hardware implement CRC and retransmiting losted data packets.

Debuged for ADucRF101MKxZ development kit

- Author: Peter Soltys
- Version: 2.1B
- Hardware: ADucRF101MKxZ
- Date: 07.02.2016
- Project: Time-multiplex-ADuc-RF101

DEV: Keil 5.1 Evaluation

- Note: v2.1B fixed synchronization and added binary mode

Compatible with program UWB - Coordinate Reader Deployment from Peter Mikula `uwb_coordinate_reader`

Program is able to find online on `Github`

1.2 Build

1.2.1 Step 1: Opening project in Keil

etc...

Chapter 2

Bug List

Global BINARY_MODE
not entirely verified (still bugs)

Global LEN_OF_RX_PKT
constant lenght not entirely verified (still bugs)

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

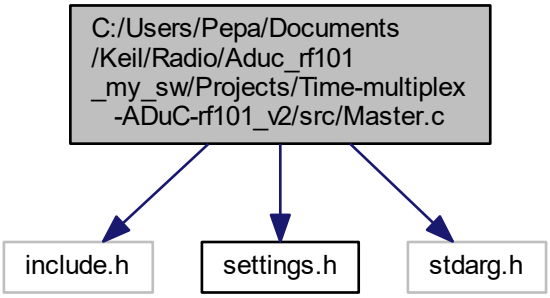
File Documentation

4.1 C:/Users/Pepa/Documents/Keil/Radio/Aduc_rf101_my_sw/Projects/Time-multiplex-ADuC-rf101_v2/src/Master.c File Reference

prigram used for Receiving data in time multiplex data are received via RF Interface and sended to UART working with Slave.c tested on ADucRF101MKxZ

```
#include "include.h"
#include "settings.h"
#include <stdarg.h>
```

Include dependency graph for Master.c:



Macros

- #define LED_OFF DioSet(pADI_GP4,BIT2)
- #define LED_ON DioClr(pADI_GP4,BIT2)
- #define UART_BUFFER_DEEP 50

Functions

- void DMA_UART_TX_Int_Handler (void)

  Interrupt handler managing sending content of pktMemory on UART with DMA.
- void uartInit (void)

  initialize uart port
- void ledInit (void)

  initialize port with led diode
- void setSynnicTimer (void)

  set general purpose timer0 for synchronization intervals
- void radioInit (void)

  initialize Radio interface
- void dmaSend (unsigned char ∗buff, int len)

  send one packet on UART with DMA controller
- int dma_printf (const char ∗format,...)

  this function is equivalent to function printf from library stdio.h
- void radioSend (char ∗buff, unsigned char len)

  send one packet trought radio interface
- unsigned char rf_printf (const char ∗format,...)

  this function is equivalent to function printf from library stdio.h
- char radioRecieve (void)

  function receive one packet with radio interface
- char validPacket (void)

  validate received packt head in global variable (Buffer)
- void copyBufferToMemory (void)

  copy received packet to packet memory
- void ifMissPktGet (void)

  function check packet meory for missing packets, send rquest for retransmit losted packets and save returned packets
- void flushBufferedPackets (void)

  rotate memory and start sending received packets on UART with DMA
- void synchronize (void)

  send synchronization packets in constant time delays
- char zeroPacket (void)

  check Buffer if is zero packet
- char receivePackets (void)

  receive all packets of sended time slot;
- void initializeNewSlot (void)

  Initialize variables for new ID slot.
- void SetInterruptPriority (void)

  Initialize interrupt priority.
- int main (void)

  main function of master program
- void GP_Tmr1_Int_Handler (void)

  Interrupt handler for measuring troughput.
- void GP_Tmr0_Int_Handler (void)

  Interrupt handler for synchronization timing.
- void DMA_UART_RX_Int_Handler ()

  Interrupt handler terminating DMA receiving transaction.
- void UART_Int_Handler ()

  Interrupt handler waiting for SYNC$ word to set synchronization flag.

Variables

- RIE_Responses RIE_Response = RIE_Success
- unsigned char Buffer [240]
- RIE_U8 PktLen
- RIE_S8 RSSI
- char pktMemory [2][NUM_OF_PACKETS_IN_MEMORY][PACKET_MEMORY_DEPTH]

  memory to store all data to send/to receive
- char numOfPkt [2] = {0,0}
- unsigned char lenghtOfPkt [2][NUM_OF_PACKETS_IN_MEMORY] = {0,0}
- unsigned char actualPacket
- signed char actualRxBuffer =0
- signed char actualTxBuffer =1
- char lastRadioTransmitBuffer [PACKET_MEMORY_DEPTH]
- char dmaTxBuffer [255]
- char rxBuffer [UART_BUFFER_DEEP]
- unsigned char slave_ID = 1
- signed char send =0
- signed char TX_flag =0
- signed char RX_flag =0
- signed char flush_flag =0
- signed char sync_flag = 0
- signed char sync_wait = 0
- signed char firstRxPkt = 0
- unsigned char rxUARTcount =0
- signed char dmaTxSlv =0
- signed char dmaTxPkt =0
- signed char dmaTx_flag =0
- char * dmaTxPtr

4.1.1 Detailed Description

prigram used for Receiving data in time multiplex data are received via RF Interface and sended to UART working with Slave.c tested on ADucRF101MKxZ

Version

V2.1B

Author

Peter Soltys

Date

febtuary 2016

Revision History:

- V1.1, July 2015 : initial version.
- V1.2, august 2015 : fully functional.
- V1.3, september 2015 : faster version with higher throughput
- V1.4, january 2016 : added synchronization
- V2.0, febtuary 2016 : new time multiplex conception
- V2.1, febtuary 2016 : fixed synchronization
- V2.1B, febtuary 2016 : Binary data packets (instead of strings with STRING_TERMINATOR)

Note

: in radioeng.c was changed intial value from
static RIE_BOOL bPacketTx = RIE_FALSE;
static RIE_BOOL bPacketRx = RIE_FALSE;
to
static RIE_BOOL bPacketTx = RIE_TRUE;
static RIE_BOOL bPacketRx = RIE_TRUE;

4.1.2 Macro Definition Documentation

4.1.2.1 #define LED_OFF DioSet(pADI_GP4,BIT2)

Definition at line 37 of file Master.c.

4.1.2.2 #define LED_ON DioClr(pADI_GP4,BIT2)

Definition at line 38 of file Master.c.

4.1.2.3 #define UART_BUFFER_DEEP 50

Definition at line 77 of file Master.c.

4.1.3 Function Documentation

4.1.3.1 void copyBufferToMemory (void)

copy received packet to packet memory

Precondition

radioInit() must be called before this function is called.
radioReceive() with returned 1
validPacket() with returned 1

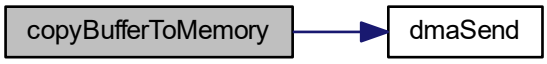
```
1 radioInit();
2 if (radioReceive())
3   if (validPacket()){
4     printf("packet was sucsesfully received with correct packet head");
5     copyBufferToMemory();
6     }
```

Note

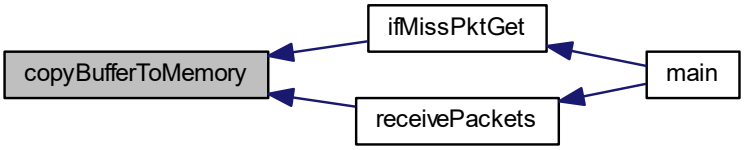
function save packet at place defined in packet head

Definition at line 426 of file Master.c.

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.3.2 int dma_printf (const char * *format*, ...)

this function is equivalent to function printf from library stdio.h

Parameters

format	: {} pointer at string to formating output string
...	:{} additive parameters for formating string

Precondition

uartInit() must be called before this can be called.

```
1 uartInit();
2 int num = 10;
3 dma_printf();
```

Note

output stream is managed with DMA controller after end of transmision is called DMA_UART_TX_Int_Handler

See also

DMA_UART_TX_Int_Handler

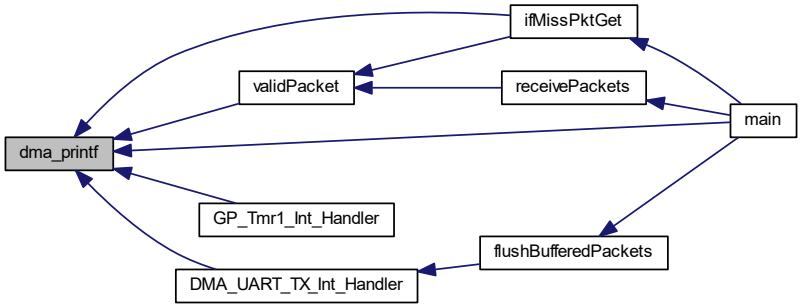
Returns

int number of sending chars (if == 0 error)

Definition at line 216 of file Master.c.

Here is the call graph for this function:

Here is the caller graph for this function:



4.1.3.3 void DMA_UART_RX_Int_Handler ( )

Interrupt handler terminating DMA receiving transaction.

Interrupt handler terminating DMA receiving transaction if constant lenght of packet is set.

Note

not used

See also

LEN_OF_RX_PKT

Definition at line 766 of file Master.c.

4.1.3.4 void DMA_UART_TX_Int_Handler (void)

Interrupt handler managing sending content of pktMemory on UART with DMA.

terminating DMA transaction of function dma_send()

Note

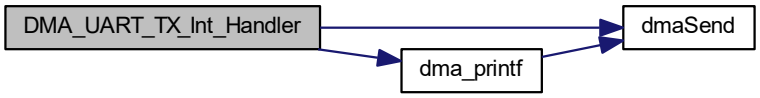
also is terminating DMA transaction of function dma_send()

See also

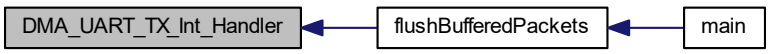
dma_send()
flushPackets()
dma_send()

Definition at line 716 of file Master.c.

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.3.5 void dmaSend (unsigned char * *buff*, int *len*)

send one packet on UART with DMA controller

Parameters

buff	:{} pointer to data to send
len	:{int range} number of bytes to send

Precondition

uartInit() must be called before this can be called.

```
1 uartInit();
2 len =vsprintf(dmaTxBuffer, format,args);//vlozenie formatovaneho retazca do buff
3 dmaSend(dmaTxBuffer,len);
```

Note

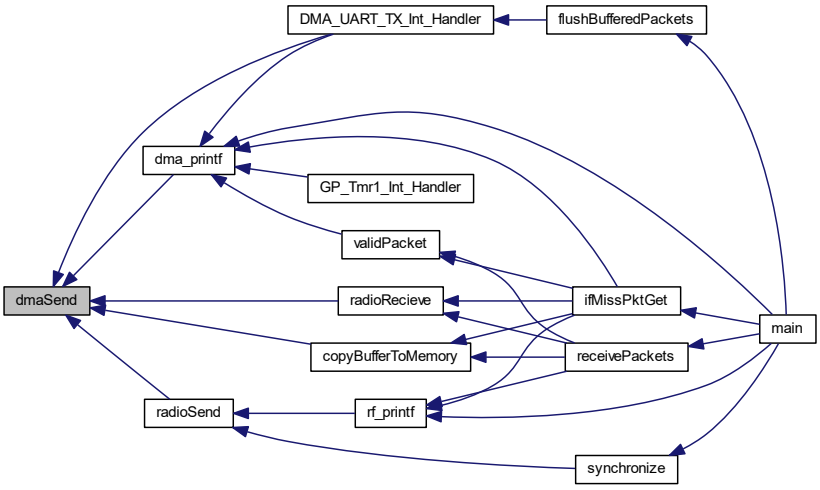
after end of transmision is called DMA_UART_TX_Int_Handler (void)

See also

DMA_UART_TX_Int_Handler

Definition at line 193 of file Master.c.

Here is the caller graph for this function:



4.1.3.6 void flushBufferedPackets (void)

rotate memory and start sending received packets on UART with DMA

See also

pktMemory

Precondition

uartInit() must be called before this function is called.

Note

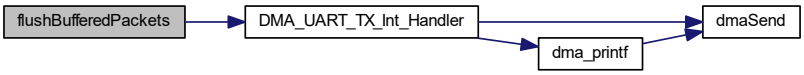
all managment about sending packets is in

See also

DMA_UART_TX_Int_Handler()

Definition at line 503 of file Master.c.

Here is the call graph for this function:



Here is the caller graph for this function:

4.1.3.7 void GP_Tmr0_Int_Handler (void)

Interrupt handler for synchronization timing.

See also

synchronize()

Note

synchronization at falling edge handler manage also rising edge (hughe jitter)

See also

setTimeToSync()

Definition at line 696 of file Master.c.

4.1.3.8 void GP_Tmr1_Int_Handler (void)

Interrupt handler for measuring troughput.

Note

only if THROUGHPUT_MEASURE is set in settings.h

See also

settings.h

Definition at line 673 of file Master.c.

Here is the call graph for this function:

4.1.3.9 void ifMissPktGet (void)

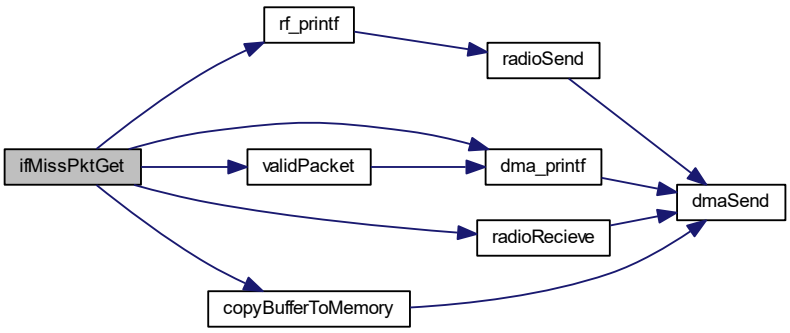
function check packet meory for missing packets, send rquest for retransmit losted packets and save returned packets

Precondition

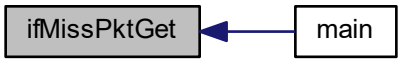
radioInit() must be called before this function is called.

Definition at line 447 of file Master.c.

Here is the call graph for this function:



Here is the caller graph for this function:

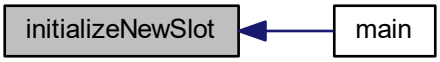


4.1.3.10 void initializeNewSlot (void)

Initialize variables for new ID slot.

Definition at line 606 of file Master.c.

Here is the caller graph for this function:



4.1.3.11 void ledInit (void)

initialize port with led diode

Definition at line 128 of file Master.c.

Here is the caller graph for this function:

4.1.3.12 int main (void)

main function of master program

main function of slave program

Note

Program is using time multiplex to receive data from 4 slave devices

Returns

int

Note

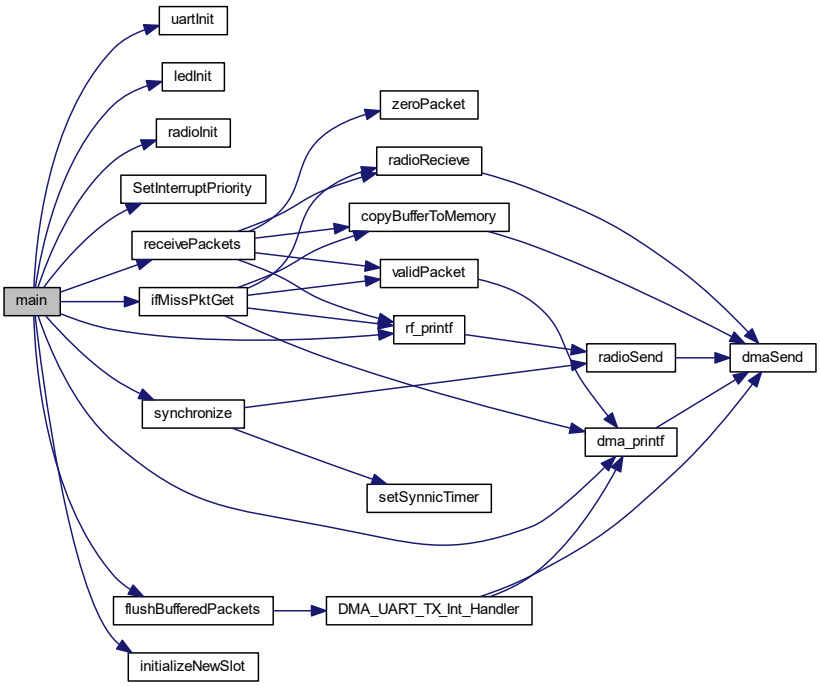
Program is using time multiplex to send data to master device

Returns

int

Definition at line 632 of file Master.c.

Here is the call graph for this function:



4.1.3.13 void radioInit (void)

initialize Radio interface

```
1 radioInit();
```

See also

settings.h

Note

see settings.h for radio configuration

Definition at line 158 of file Master.c.

Here is the caller graph for this function:

4.1.3.14 char radioRecieve (void)

function receive one packet with radio interface

Precondition

radioInit() must be called before this function is called.

```
1 radioInit();
2 if (radioReceive())
3   printf("packet was received and read from radio interface");
4   printf(Buffer);//Buffer is global bufer for radio interface
5 else
6   printf("packet was not received correctly before timeout");
```

Note

function is also reding packet form radio interface to unsigned char Buffer[240]

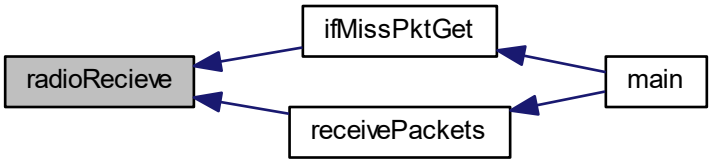
Returns

char 1 == packet received, 0 == packet was not received correctly before timout

Definition at line 316 of file Master.c.

Here is the call graph for this function:

Here is the caller graph for this function:



4.1.3.15 void radioSend (char * *buff*, unsigned char *len*)

send one packet trought radio interface

Parameters

buff	:{1-240} pointer at memory to be sended
len	:{0-240} number of bytes to be sended

Precondition

radioInit() must be called before this function is called.

```
1 len=vsprintf(buff, format,args);//@see rf_printf();
2 if(len<240){//check max lenght
3   radioSend(buff,len+1);//send formated packet
```

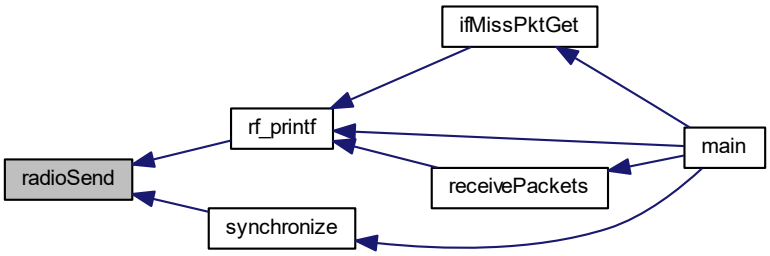
Note

output stream is trought radio interface function is waiting until whole packet is trnsmited

Definition at line 244 of file Master.c.

Here is the call graph for this function:

Here is the caller graph for this function:



4.1.3.16 char receivePackets (void)

receive all packets of sended time slot;

Precondition

radioInit() must be called before this function is called.
rf_printf("1slot") or equivalent should be called first.

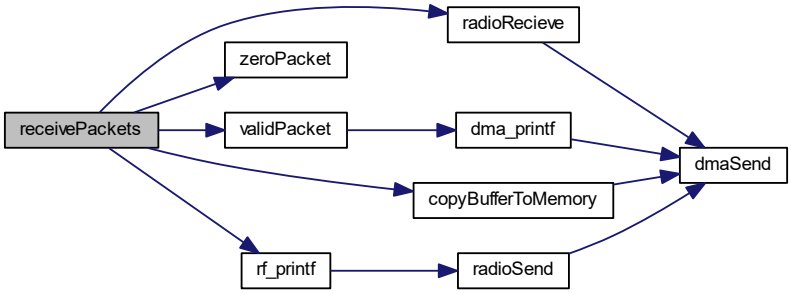
```
1 radioInit();
2 rf_printf("1slot");//slot identificator
3 if (receivePackets())
4   flushPacktes();//send all received packtes on UART
```

Note

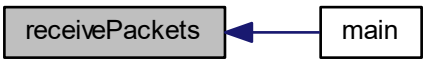
function si also retransmitin slot ID packtes if no response

Definition at line 567 of file Master.c.

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.3.17 int rf_printf (const char * *format*, ...)

this function is equivalent to function printf from library stdio.h

Parameters

format	: {} pointer at string to formating output string
...	:{} additive parameters for formating string

Precondition

radioInoit() must be called before this can be called.

```
1 radioInit();
2 int num = 10;
3 rf_printf();
```

Note

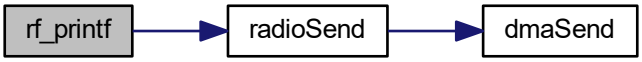
output stream is trought radio interface function is waiting until whole packet is transmited any one formated string (call) is sended in one packet

Returns

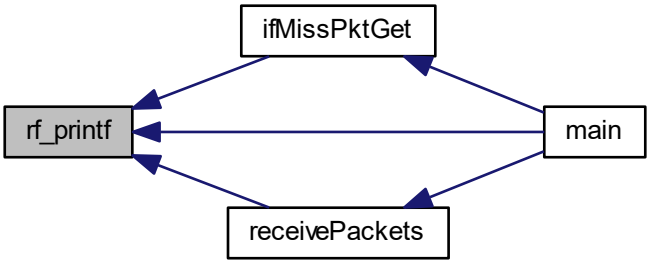
int number of sending chars (if == 0 error)

Definition at line 284 of file Master.c.

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.3.18 void SetInterruptPriority (void)

Initialize interrupt priority.

Definition at line 619 of file Master.c.

Here is the caller graph for this function:

4.1.3.19 void setSynnicTimer (void)

set general purpose timer0 for synchronization intervals

See also

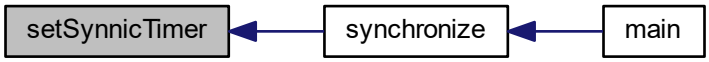
void GP_Tmr1_Int_Handler ()

Note

timer predivider factor = 256, processor clock, periodic mode

Definition at line 140 of file Master.c.

Here is the caller graph for this function:



4.1.3.20 void synchronize (void)

send synchronization packets in constant time delays

Precondition

radioInit() must be called before this function is called.

Note

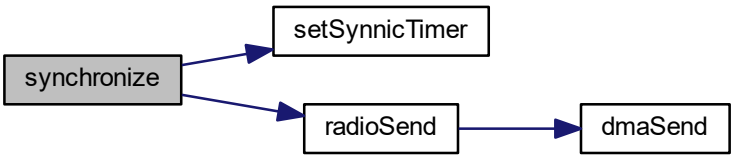
function flags controlled by

See also

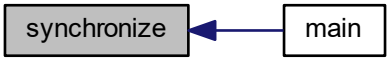
GP_Tmr0_Int_Handler and
UART_Int_Handler

Definition at line 525 of file Master.c.

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.3.21 void UART_Int_Handler (void)

Interrupt handler waiting for SYNC$ word to set synchronization flag.

Interrupt handler managing store received data trought UART.

Note

function is taken from example UARTLoopback.c and modified

See also

synchronize()

- string mode = appedning chars untill STRING_TERMINATOR is received
- binary mode = appending untill buffer is full == 240 chars

  See also

  STRING_TERMINATOR

Definition at line 781 of file Master.c.

4.1.3.22 void uartInit (void)

initialize uart port

```
1 uartInit();
```

Note

speed UART_BAUD_RATE_MASTER baud 8 bits one stop bit output port P1.0/P1.1

Definition at line 113 of file Master.c.

Here is the caller graph for this function:

4.1.3.23 char validPacket (void)

validate received packt head in global variable (Buffer)

Precondition

radioInit() must be called before this function is called.
radioReceive() with returned 1

```
1 radioInit();
2 if (radioReceive())
3   if (validPacket())
4     printf("packet was sucsesfully received with correct packet head");
```

Note

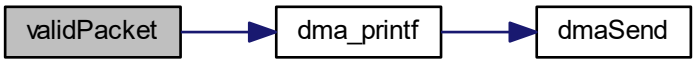
function send messages about incorrect packet on UART

Returns

char 1 == valid packet head, 0 == invalid packet head

Definition at line 371 of file Master.c.

Here is the call graph for this function:



Here is the caller graph for this function:

4.1.3.24 char zeroPacket (void)

check Buffer if is zero packet

Note

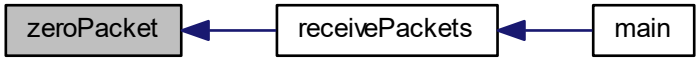
zero packet mean that slave have no buffered packets

Returns

1 == zero packet, 0 == no zero packet

Definition at line 547 of file Master.c.

Here is the caller graph for this function:



4.1.4 Variable Documentation

4.1.4.1 unsigned char actualPacket

Definition at line 71 of file Master.c.

4.1.4.2 signed char actualRxBuffer =0

Definition at line 73 of file Master.c.

4.1.4.3 signed char actualTxBuffer =1

Definition at line 73 of file Master.c.

4.1.4.4 unsigned char Buffer[240]

Definition at line 43 of file Master.c.

4.1.4.5 signed char dmaTx_flag =0

Definition at line 91 of file Master.c.

4.1.4.6 char dmaTxBuffer[255]

Definition at line 76 of file Master.c.

4.1.4.7 signed char dmaTxPkt =0

Definition at line 91 of file Master.c.

4.1.4.8 char* dmaTxPtr

Definition at line 92 of file Master.c.

4.1.4.9 signed char dmaTxSlv =0

Definition at line 91 of file Master.c.

4.1.4.10 signed char firstRxPkt = 0

Definition at line 87 of file Master.c.

4.1.4.11 signed char flush_flag =0

Definition at line 84 of file Master.c.

4.1.4.12 char lastRadioTransmitBuffer[PACKET_MEMORY_DEPTH]

Definition at line 75 of file Master.c.

4.1.4.13 unsigned char lenghtOfPkt[2][NUM_OF_PACKETS_IN_MEMORY] = {0,0}

Definition at line 69 of file Master.c.

4.1.4.14 char numOfPkt[2] = {0,0}

Definition at line 67 of file Master.c.

4.1.4.15 RIE_U8 PktLen

Definition at line 44 of file Master.c.

4.1.4.16 char pktMemory[2][NUM_OF_PACKETS_IN_MEMORY][PACKET_MEMORY_DEPTH]

memory to store all data to send/to receive

Parameters

-	level • packet num • packet data

Note

pktMemory is 2 levels deep puspose is changing in circle 0 actual receiving buffer 1 actual sending buffer for pointing are used flags : actualRxBuffer, actualTxBuffer
size of mermory is restricted because ADuc rf101 have only 16KBytes SRAM
UART nust be much faster(tested on 128000 baud rate) than RF-link (to release memory)

See also

actualRxBuffer
actualTxBuffer

Definition at line 64 of file Master.c.

4.1.4.17 RIE_Responses RIE_Response = RIE_Success

Definition at line 42 of file Master.c.

4.1.4.18 RIE_S8 RSSI

Definition at line 45 of file Master.c.

4.1.4.19 signed char RX_flag =0

Definition at line 84 of file Master.c.

4.1.4.20 char rxBuffer[UART_BUFFER_DEEP]

Definition at line 78 of file Master.c.

4.1.4.21 unsigned char rxUARTcount =0

Definition at line 88 of file Master.c.

4.1.4.22 signed char send =0

Definition at line 81 of file Master.c.

4.1.4.23 unsigned char slave_ID = 1

Definition at line 80 of file Master.c.

4.1.4.24 signed char sync_flag = 0

Definition at line 85 of file Master.c.

4.1.4.25 signed char sync_wait = 0

Definition at line 86 of file Master.c.

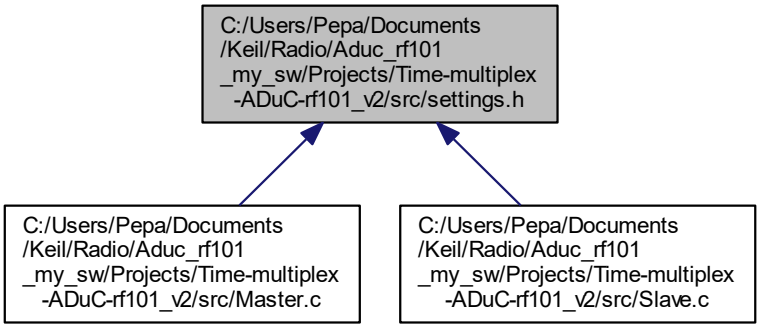
4.1.4.26 signed char TX_flag =0

Definition at line 84 of file Master.c.

4.2 C:/Users/Pepa/Documents/Keil/Radio/Aduc_rf101_my_sw/Projects/Time-multiplex-↩ ADuC-rf101_v2/src/settings.h File Reference

configurating file defining base settings

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



Macros

- #define RADIO_CFG DR_300_0kbps_Dev75_0kHz
- #define RADIO_MODULATION GFSK_Modulation

  Radio Transmitter Modulation Type.
- #define RADIO_FREQENCY 433920000

  Frequency for radio communications.
- #define PA_TYPE DifferentialPA

  PA Type for Radio Transmission.
- #define RADIO_POWER PowerLevel15

  Power Level for Radio Transmission.
- #define DATA_WHITENING RIE_FALSE

  Enable or Disable Data Whitening of payload data.
- #define RADIO_MANCHASTER RIE_FALSE

  Enable or Disable Manchester Encoding of payload data.
- #define BINARY_MODE 0

  Enable or Disable binary mode.
- #define STRING_TERMINATOR '$'

  char witch terminate all received packets
- #define PACKET_MEMORY_DEPTH 240

  maximal memory (packet) depth
- #define NUM_OF_PACKETS_IN_MEMORY 20

  maximal number of packets to send/receive
- #define NUM_OF_SLAVE 4

  number of expected slave devices
- #define LEN_OF_RX_PKT 0

  lenght of received packets from UART

- #define CHAR_OFFSET '0'

  offset in numbers of head
- #define SYNC_INTERVAL 200

  time interval to interrupt for synchronization
- #define T_TIMEOUT 50000

  max time(number of increments) to response of requested devide
- #define UART_BAUD_RATE_MASTER 128000

  UART baudrate with is using master.
- #define TIME_SLOT_ID_MASTER "%dslot",slave_ID

  string defining format of "ID slot" packet
- #define RETRANSMISION 3

  number of retransmission trying until slave is marked as not responding
- #define UART_BAUD_RATE_SLAVE 128000

  UART baudrate with is using master.
- #define TIME_SLOT_ID_SLAVE "2slot"

  format of slot identificator
- #define ZERO_PACKET "200"

  format of zero packet
- #define RETRANSMISION_ID "2RE"

  format of retransmision packet
- #define SLAVE_ID 2

  number of actual slave
- #define HEAD_LENGHT 3

  lenght of head in bytes
- #define HEAD_FORMAT "%d%c%c",SLAVE_ID,txPkt,numOfPackets[actualTxBuffer]-1
- #define T_PROCESSING 0

  appended time(number of increments) after transmition to procesing on master
- #define SYNC_PIN_HIGH DioSet(pADI_GP4,BIT2)
- #define SYNC_PIN_LOW DioClr(pADI_GP4,BIT2)
- #define SYNC_PIN_READ DioRd(pADI_GP4)&0x04
- #define SIMULATE_RETX 0

  sending retransmitin message to test
- #define DEBUG_MESAGES 0

  stream of mesages to UART
- #define RX_STREAM 0

  stream of redeived data to UART
- #define TX_STREAM 0

  stream of transmited data to UART
- #define SEND_HEAD 0

  send also heads of packets on UART
- #define THROUGHPUT_MEASURE 0

  measured data troughput

4.2.1 Detailed Description

configurating file defining base settings

Version

V2.1B

Author

Peter Soltys

Date

february 2016

Note

: in radioeng.c was changed intial value from
static RIE_BOOL bPacketTx = RIE_FALSE;
static RIE_BOOL bPacketRx = RIE_FALSE;
to
static RIE_BOOL bPacketTx = RIE_TRUE;
static RIE_BOOL bPacketRx = RIE_TRUE;

4.2.2 Disclaimer

THIS SOFTWARE IS PROVIDED BY BC PETER SOLTYS. “AS IS" AND ANY EXPRESS OR IMPLIED WARR↩ ANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT, ARE DISCLAIMED. IN NO EVENT SHALL BC PE↩ TER SOLTYS. BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSE↩ QUENTIAL DAMAGES ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

YOU ASSUME ANY AND ALL RISK FROM THE USE OF THIS CODE OR SUPPORT FILE.

IT IS THE RESPONSIBILITY OF THE PERSON INTEGRATING THIS CODE INTO AN APPLICATION TO ENSURE THAT THE RESULTING APPLICATION PERFORMS AS REQUIRED AND IS SAFE.

4.2.3 Macro Definition Documentation

4.2.3.1 #define BINARY_MODE 0

Enable or Disable binary mode.

Binary mode is transmitting data packets in maximal lenght 240 All received data in slave are stored together

In (normal) string mode packets are terminated with STRING_TERMINATOR

Bug not entirely verified (still bugs)

Parameters

bool	:{1 , 0} • 1 if Binary mode • 0 if String mode.

Definition at line 137 of file settings.h.

4.2.3.2 #define CHAR_OFFSET '0'

offset in numbers of head

Note

offset in numbers of head, because 0x00 is defined like end of string => working with packets as with strings

Parameters

char	:{'0'}

Definition at line 183 of file settings.h.

4.2.3.3 #define DATA_WHITENING RIE_FALSE

Enable or Disable Data Whitening of payload data.

Data whitening can be employed to avoid long runs of 1s or 0s in the transmitted data stream.

This ensures sufficient bit transitions in the packet, which aids in receiver clock and data recovery because the encoding breaks up long runs of 1s or 0s in the transmit packet.

The data, excluding the preamble and sync word, is automatically whitened before transmission by XORing the data with an 8-bit pseudorandom sequence.

At the receiver, the data is XORed with the same pseudorandom sequence, thereby reversing the whitening.

The linear feedback shift register polynomial used is x7 + x1 + 1.

Parameters

bEnable	:{RIE_FALSE, RIE_TRUE} • RIE_TRUE if Manchester Encoding is to be enabled. • RIE_FALSE if disabled.

Definition at line 104 of file settings.h.

4.2.3.4 #define DEBUG_MESAGES 0

stream of mesages to UART

Definition at line 270 of file settings.h.

4.2.3.5 #define HEAD_FORMAT "%d%c%c",SLAVE_ID,txPkt,numOfPackets[actualTxBuffer]-1

Definition at line 255 of file settings.h.

4.2.3.6 #define HEAD_LENGHT 3

lenght of head in bytes

Definition at line 253 of file settings.h.

4.2.3.7 #define LEN_OF_RX_PKT 0

lenght of received packets from UART

Note

macro set greatness of packet memory

Bug constant lenght not entirely verified (still bugs)

Parameters

bool	:{0 , 240} • 0 if variable lenght of packets • 1-240 constant lenght of packets

Definition at line 175 of file settings.h.

4.2.3.8 #define NUM_OF_PACKETS_IN_MEMORY 20

maximal number of packets to send/receive

Note

macro set greatness of packet memory

Parameters

number	of packets :{1 , 20}

Definition at line 157 of file settings.h.

4.2.3.9 #define NUM_OF_SLAVE 4

number of expected slave devices

Note

number is restricted by size of memory macro set greatness of packet memory

Parameters

number	of slave devices :{1 , 10} 4

Definition at line 165 of file settings.h.

4.2.3.10 #define PA_TYPE DifferentialPA

PA Type for Radio Transmission.

Parameters

PAType	:{DifferentialPA, SingleEndedPA} Select Single Ended or Differential PA Type

Definition at line 70 of file settings.h.

4.2.3.11 #define PACKET_MEMORY_DEPTH 240

maximal memory (packet) depth

Note

macro set greatness of packet memory

Parameters

memory	deepth :{0 , 240}

Definition at line 150 of file settings.h.

4.2.3.12 #define RADIO_CFG DR_300_0kbps_Dev75_0kHz

Parameters

BaseConfig	:{DR_1_0kbps_Dev10_0kHz , DR_38_4kbps_Dev20kHz ,DR_300_0kbps_Dev75_0kHz } • DR_1_0kbps_Dev10_0kHz Base configuration of 1 kbps datarate, 10.0 kHz frequency deviation. • DR_38_4kbps_Dev20kHz Base configuration of 38.4 kbps datarate, 20 kHz frequency deviation. • DR_300_0kbps_Dev75_0kHz Base configuration of 300 kbps datarate, 75 kHz frequency deviation.

Definition at line 46 of file settings.h.

4.2.3.13 #define RADIO_FREQENCY 433920000

Frequency for radio communications.

Parameters

Frequency	:{431000000-928000000} • This must be within the available bands of the radio: – 431000000Hz to 464000000Hz and – 862000000Hz to 928000000Hz.

Note

433.92 Mhz (EU) free frequency

Definition at line 64 of file settings.h.

4.2.3.14 #define RADIO_MANCHASTER RIE_FALSE

Enable or Disable Manchester Encoding of payload data.

Manchester encoding can be used to ensure a dc-free (zero mean) transmission.

A Binary 0 is mapped to 10, and a Binary 1 is mapped to 01.

Manchester encoding and decoding are applied to the payload data and the CRC.

Parameters

bEnable	:{RIE_FALSE,RIE_TRUE} • RIE_TRUE if Manchester Encoding is to be enabled. • RIE_FALSE if disabled.

Definition at line 120 of file settings.h.

4.2.3.15 #define RADIO_MODULATION GFSK_Modulation

Radio Transmitter Modulation Type.

Parameters

ModulationType	:{FSK_Modulation , GFSK_Modulation } • FSK_Modulation Frequency shift keying modulatino • GFSK_Modulation Gausian frequency shift keying modulatino

Definition at line 54 of file settings.h.

4.2.3.16 #define RADIO_POWER PowerLevel15

Power Level for Radio Transmission.

Parameters

Power	:{PowerLevel0 ,PowerLevel1 ,PowerLevel2 ,PowerLevel3, PowerLevel4 ,PowerLevel5 ,PowerLevel6 ,PowerLevel7, PowerLevel8 ,PowerLevel9 ,PowerLevel10,PowerLevel11, PowerLevel12,PowerLevel13,PowerLevel14,PowerLevel15}

Definition at line 79 of file settings.h.

4.2.3.17 #define RETRANSMISION 3

number of retransmission trying until slave is marked as not responding

Parameters

retransmission	attempts :{3}

Definition at line 219 of file settings.h.

4.2.3.18 #define RETRANSMISION_ID "2RE"

format of retransmision packet

Parameters

slave	number{1 - NUM_OF_SLAVE}

Definition at line 244 of file settings.h.

4.2.3.19 #define RX_STREAM 0

stream of redeived data to UART

Definition at line 271 of file settings.h.

4.2.3.20 #define SEND_HEAD 0

send also heads of packets on UART

Definition at line 273 of file settings.h.

4.2.3.21 #define SIMULATE_RETX 0

sending retransmitin message to test

Definition at line 269 of file settings.h.

4.2.3.22 #define SLAVE_ID 2

number of actual slave

Parameters

slave	number{1 - NUM_OF_SLAVE}

Definition at line 248 of file settings.h.

4.2.3.23 #define STRING_TERMINATOR '$'

char witch terminate all received packets

Parameters

char	:{'$'}

Definition at line 143 of file settings.h.

4.2.3.24 #define SYNC_INTERVAL 200

time interval to interrupt for synchronization

Note

time to interrupt = (1/40 000 000) ∗ 256 ∗ SYNC_INTERVAL [s] 200 = 1.28 ms //up to 255 (unsigned char)

Parameters

count	number :{200}

Definition at line 191 of file settings.h.

4.2.3.25 #define SYNC_PIN_HIGH DioSet(pADI_GP4,BIT2)

Definition at line 262 of file settings.h.

4.2.3.26 #define SYNC_PIN_LOW DioClr(pADI_GP4,BIT2)

Definition at line 263 of file settings.h.

4.2.3.27 #define SYNC_PIN_READ DioRd(pADI_GP4)&0x04

Definition at line 264 of file settings.h.

4.2.3.28 #define T_PROCESSING 0

appended time(number of increments) after transmition to procesing on master

Definition at line 259 of file settings.h.

4.2.3.29 #define T_TIMEOUT 50000

max time(number of increments) to response of requested devide

Note

interval witch is counted until packet is received

See also

radioRecieve()

Parameters

time	:{50000}

Definition at line 199 of file settings.h.

4.2.3.30 #define THROUGHPUT_MEASURE 0

measured data troughput

Parameters

{0-3}	• if ==0 no measuring • if ==1 measure throughput of received data from UART • if ==2 measure maximum throughput with shyntetic data

Note

measured are all data included packet heads (aproximetlz 5000 Bytes/s by slave)

Definition at line 283 of file settings.h.

4.2.3.31 #define TIME_SLOT_ID_MASTER "%dslot",slave_ID

string defining format of "ID slot" packet

Definition at line 213 of file settings.h.

4.2.3.32 #define TIME_SLOT_ID_SLAVE "2slot"

format of slot identificator

Parameters

slave	number{1 - NUM_OF_SLAVE}

Definition at line 236 of file settings.h.

4.2.3.33 #define TX_STREAM 0

stream of transmited data to UART

Definition at line 272 of file settings.h.

4.2.3.34 #define UART_BAUD_RATE_MASTER 128000

UART baudrate with is using master.

Definition at line 208 of file settings.h.

4.2.3.35 #define UART_BAUD_RATE_SLAVE 128000

UART baudrate with is using master.

Note

Baudrate is ste to 9600 because of compatibility with "UWB - Coordinate Reader Deployment" from Peter Mikula

Definition at line 230 of file settings.h.

4.2.3.36 #define ZERO_PACKET "200"

format of zero packet

Parameters

slave	number{1 - NUM_OF_SLAVE} first number

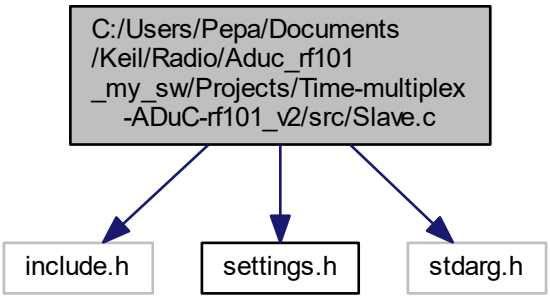
Definition at line 240 of file settings.h.

4.3 C:/Users/Pepa/Documents/Keil/Radio/Aduc_rf101_my_sw/Projects/Time-multiplex-ADuC-rf101_v2/src/Slave.c File Reference

prigram used for transmiting data in time multiplex data are received via UART and sended trouhgt radio interface during associated time slot working with Master.c tested on ADucRF101MKxZ

```
#include "include.h"
#include "settings.h"
#include <stdarg.h>
```

Include dependency graph for Slave.c:



Macros

- #define LED_OFF DioSet(pADI_GP4,BIT2)
- #define LED_ON DioClr(pADI_GP4,BIT2)

Functions

- void UART_Int_Handler (void)
- void uart_init (void)
- void led_init (void)

  initialize port with led diode
- void radioInit (void)
- void dmaSend (unsigned char ∗buff, int len)
- int dma_printf (const char ∗format,...)
- void radioSend (char ∗buff, unsigned char len)
- unsigned char rf_printf (const char ∗format,...)
- char radioRecieve (void)
- void setTimeToSync (unsigned int time)

  set general purpose timer0 for synchronization timeout
- char transmit (void)

  function is transmiting all prepared packets throught radio link
- char retransmit (void)

  function retransmit missing packets if requested
- void SetInterruptPriority (void)
- int main (void)
- void GP_Tmr0_Int_Handler (void)
- void GP_Tmr1_Int_Handler (void)
- void DMA_UART_TX_Int_Handler ()
- void DMA_UART_RX_Int_Handler ()
- void HardFault_Handler (void)

Variables

- RIE_Responses RIE_Response = RIE_Success
- unsigned char Buffer [255]
- RIE_U8 PktLen
- RIE_S8 RSSI
- char pktMemory [2][NUM_OF_PACKETS_IN_MEMORY][PACKET_MEMORY_DEPTH]

  memory to store all data to send/to receive
- char numOfPkt [2] = {0,0}
- unsigned char lenghtOfPkt [2][NUM_OF_PACKETS_IN_MEMORY] = {0,0}
- char actualRxBuffer =0
- char actualTxBuffer =1
- char rxUARTbuffer [255]
- char * rxPktPtr
- int rxUARTcount = 0
- char TX_flag =0
- char RX_flag =0
- char terminate_flag =0
- char buffer_change_flag =0
- char my_slot = 0
- int i =0
- int j =0
- int debugTimer =0

4.3.1 Detailed Description

prigram used for transmiting data in time multiplex data are received via UART and sended trouhgt radio interface during associated time slot working with Master.c tested on ADucRF101MKxZ

Version

V2.1B

Author

Peter Soltys

Date

febtuary 2016

Revision History:

- V1.0, July 2015 : initial version.
- V1.1, august 2015 : fully functional. (only shyntetic data transmitted)
- V1.2, august 2015 : fully functional. (transmiting received data via UART (only fixed lenght packet))
- V1.3, september 2015 : faster version with higher throughput and transmiting variable lenght packets working with "UWB - Coordinate Reader Deployment" from Peter Mikula
- V1.4, january 2015 : added synchronization
- V2.0, febtuary 2016 : new time multiplex conception
- V2.1, febtuary 2016 : fixed synchronization
- V2.1B, febtuary 2016 : Binary data packets (instead of strings with STRING_TERMINATOR)

Note

: in radioeng.c was changed intial value from
static RIE_BOOL bPacketTx = RIE_FALSE;
static RIE_BOOL bPacketRx = RIE_FALSE;
to
static RIE_BOOL bPacketTx = RIE_TRUE;
static RIE_BOOL bPacketRx = RIE_TRUE;

4.3.2 Macro Definition Documentation

4.3.2.1 #define LED_OFF DioSet(pADI_GP4,BIT2)

Definition at line 40 of file Slave.c.

4.3.2.2 #define LED_ON DioClr(pADI_GP4,BIT2)

Definition at line 41 of file Slave.c.

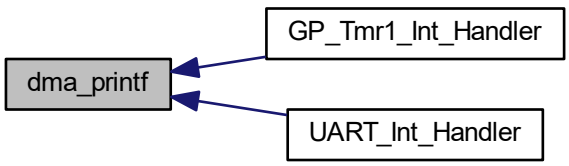
4.3.3 Function Documentation

4.3.3.1 int dma_printf (const char * *format,* ...)

Definition at line 215 of file Slave.c.

Here is the call graph for this function:

Here is the caller graph for this function:



4.3.3.2 void DMA_UART_RX_Int_Handler ( )

Definition at line 612 of file Slave.c.

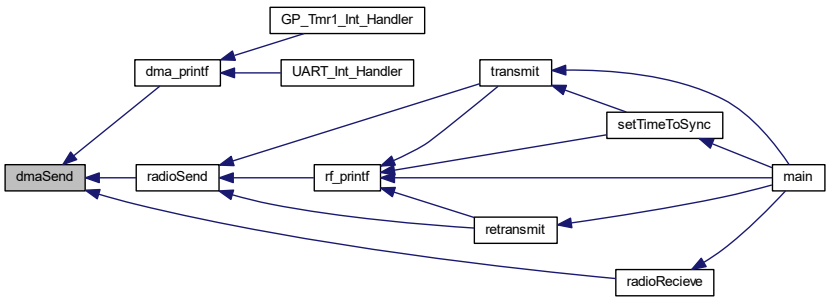
4.3.3.3 void DMA_UART_TX_Int_Handler (void)

Definition at line 597 of file Slave.c.

4.3.3.4 void dmaSend (unsigned char * *buff,* int *len*)

Definition at line 192 of file Slave.c.

Here is the caller graph for this function:



4.3.3.5 void GP_Tmr0_Int_Handler (void)

Definition at line 552 of file Slave.c.

4.3.3.6 void GP_Tmr1_Int_Handler (void)

Definition at line 577 of file Slave.c.

Here is the call graph for this function:

4.3.3.7 void HardFault_Handler (void)

Definition at line 637 of file Slave.c.

4.3.3.8 void led_init (void)

initialize port with led diode

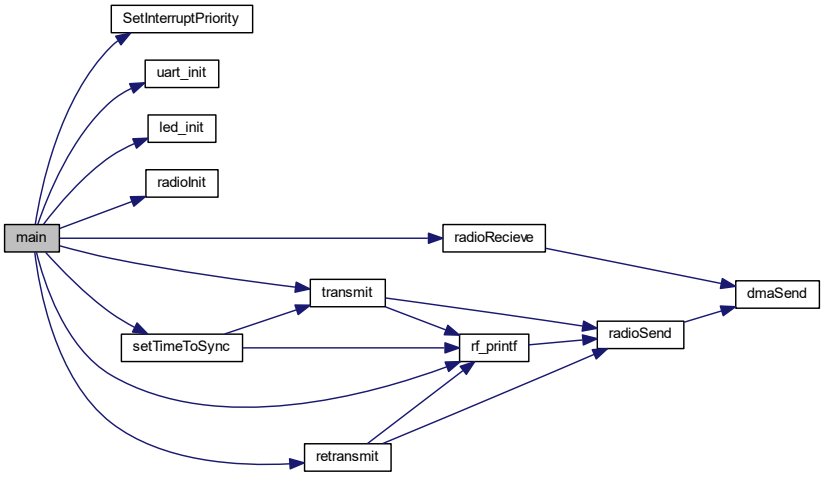
Definition at line 142 of file Slave.c.

Here is the caller graph for this function:

4.3.3.9 int main (void)

Definition at line 506 of file Slave.c.

Here is the call graph for this function:



4.3.3.10 void radioInit (void)

Definition at line 157 of file Slave.c.

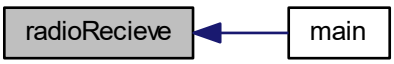
Here is the caller graph for this function:

4.3.3.11 char radioRecieve (void)

Definition at line 323 of file Slave.c.

Here is the call graph for this function:

Here is the caller graph for this function:

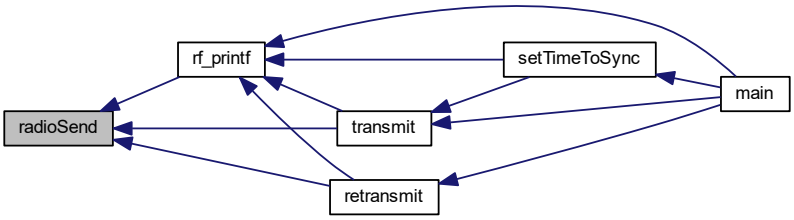


4.3.3.12 void radioSend (char * *buff*, unsigned char *len*)

Definition at line 244 of file Slave.c.

Here is the call graph for this function:

Here is the caller graph for this function:



4.3.3.13 char retransmit (void)

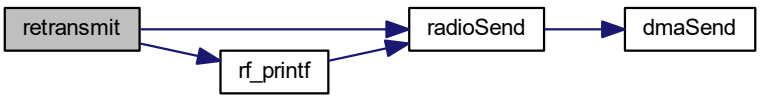
function retransmit missing packets if requested

Returns

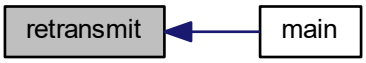
char - number of retransmited packets

Definition at line 470 of file Slave.c.

Here is the call graph for this function:



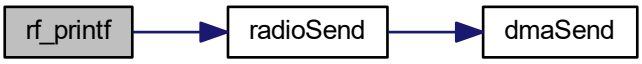
Here is the caller graph for this function:



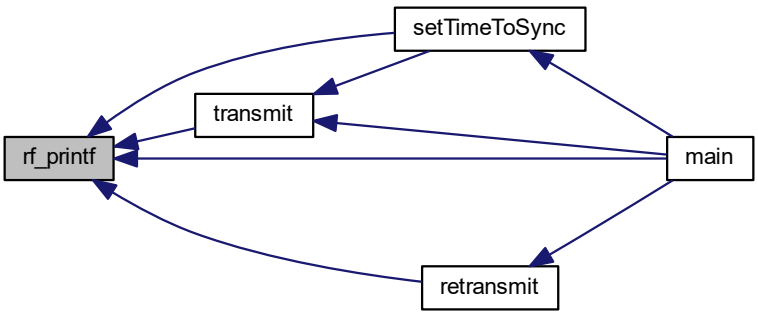
4.3.3.14 unsigned char rf_printf (const char * *format*, ...)

Definition at line 290 of file Slave.c.

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.3.15 void SetInterruptPriority (void)

Definition at line 491 of file Slave.c.

Here is the caller graph for this function:

4.3.3.16 void setTimeToSync (unsigned int *time*)

set general purpose timer0 for synchronization timeout

Parameters

time	{0- uint range} should be (N ∗ SYNC_INTERVAL) where N=1-3

See also

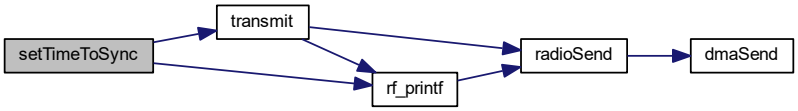
void GP_Tmr0_Int_Handler ()
SYNC_INTERVAL

Note

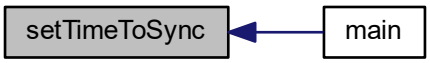
timer predivider factor = 256, processor clock, periodic mode

Definition at line 373 of file Slave.c.

Here is the call graph for this function:



Here is the caller graph for this function:



4.3.3.17 char transmit (void)

function is transmiting all prepared packets throught radio link

Note

rotate packet memory all data packets are received via UART

See also

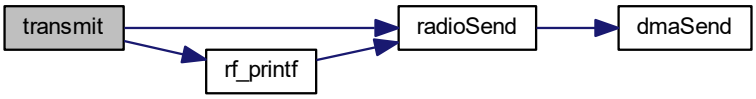
pktMemory

Returns

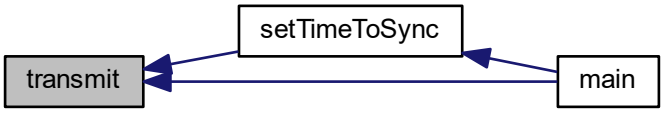
char - number of transmitted packets

Definition at line 425 of file Slave.c.

Here is the call graph for this function:



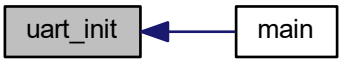
Here is the caller graph for this function:



4.3.3.18 void uart_init (void)

Definition at line 111 of file Slave.c.

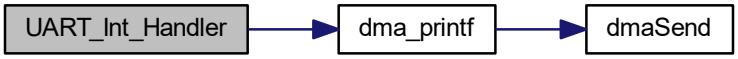
Here is the caller graph for this function:



4.3.3.19 void UART_Int_Handler (void)

Definition at line 652 of file Slave.c.

Here is the call graph for this function:



4.3.4 Variable Documentation

4.3.4.1 char actualRxBuffer =0

Definition at line 80 of file Slave.c.

4.3.4.2 char actualTxBuffer =1

Definition at line 81 of file Slave.c.

4.3.4.3 unsigned char Buffer[255]

Definition at line 45 of file Slave.c.

4.3.4.4 char buffer_change_flag =0

Definition at line 87 of file Slave.c.

4.3.4.5 int debugTimer =0

Definition at line 93 of file Slave.c.

4.3.4.6 int i =0

Definition at line 92 of file Slave.c.

4.3.4.7 int j =0

Definition at line 92 of file Slave.c.

4.3.4.8 unsigned char lenghtOfPkt[2][NUM_OF_PACKETS_IN_MEMORY] = {0,0}

Definition at line 77 of file Slave.c.

4.3.4.9 char my_slot = 0

Definition at line 88 of file Slave.c.

4.3.4.10 char numOfPkt[2] = {0,0}

Definition at line 75 of file Slave.c.

4.3.4.11 RIE_U8 PktLen

Definition at line 46 of file Slave.c.

4.3.4.12 char pktMemory[2][NUM_OF_PACKETS_IN_MEMORY][PACKET_MEMORY_DEPTH]

memory to store all data to send/to receive

Parameters

-	level • packet num • packet data

Note

pktMemory is 2 levels deep puspose is changing in circle 0 actual receiving buffer 1 actual sending buffer for pointing are used flags : actualRxBuffer, actualTxBuffer
size of mermory is restricted because ADuc rf101 have only 16KBytes SRAM
UART nust be much faster(tested on 128000 baud rate) than RF-link (to release memory)

See also

actualRxBuffer
actualTxBuffer

Definition at line 66 of file Slave.c.

4.3.4.13 RIE_Responses RIE_Response = RIE_Success

Definition at line 44 of file Slave.c.

4.3.4.14 RIE_S8 RSSI

Definition at line 47 of file Slave.c.

4.3.4.15 char RX_flag =0

Definition at line 87 of file Slave.c.

4.3.4.16 char* rxPktPtr

Definition at line 84 of file Slave.c.

4.3.4.17 char rxUARTbuffer[255]

Definition at line 83 of file Slave.c.

4.3.4.18 int rxUARTcount = 0

Definition at line 85 of file Slave.c.

4.3.4.19 char terminate_flag =0

Definition at line 87 of file Slave.c.

4.3.4.20 char TX_flag =0

Definition at line 87 of file Slave.c.

Index