

i-easy lib Reference Manual

0.3

V0.3 by optiCompo electronics

Feb 2002

Contents

1	i-easy lib Data Structure Index	1
1.1	i-easy lib Data Structures	1
2	i-easy lib File Index	3
2.1	i-easy lib File List	3
3	i-easy lib Data Structure Documentation	5
3.1	IP.PORT Struct Reference	5
4	i-easy lib File Documentation	7
4.1	DEMO.C File Reference	7
4.2	dial.c File Reference	10
4.3	dial.h File Reference	13
4.4	GLOBAL.H File Reference	15
4.5	ICHIP.C File Reference	18
4.6	ICHIP.H File Reference	23
4.7	SOCKET.C File Reference	40
4.8	SOCKET.H File Reference	43
4.9	TIMER.C File Reference	46
4.10	TIMER.H File Reference	47
4.11	UART.C File Reference	49
4.12	UART.H File Reference	53

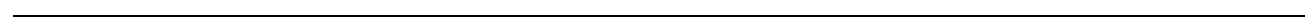
Chapter 1

i-easy lib Data Structure Index

1.1 i-easy lib Data Structures

Here are the data structures with brief descriptions:

[IP_PORT](#) (Structure for holding all connection parameters for a TCP or UDP connection) 5



Chapter 2

i-easy lib File Index

2.1 i-easy lib File List

Here is a list of all files with brief descriptions:

DEMO.C (Sample Application for i-easy board V 1.0 (e-mail and time server test))	7
dial.c (Dialup functions for dialing and PPP login (i-easy library))	10
dial.h (Dialup definitions for dialing and PPP login (i-easy library))	13
GLOBAL.H (Global define and typedefs)	15
ICHIP.C (Low level access functions (i-easy library))	18
ICHIP.H (Low level access definitions (i-easy library))	23
SOCKET.C (Socket and TCP/IP connection functions (i-easy library))	40
SOCKET.H (Socket and TCP/IP connection definitions (i-easy library))	43
TIMER.C (Timer library by Volker Oth (modified))	46
TIMER.H (Timer library by Volker Oth (modified))	47
UART.C (UART library by Volker Oth (modified))	49
UART.H (UART library by Volker Oth (modified))	53

Chapter 3

i-easy lib Data Structure Documentation

3.1 IP_PORT Struct Reference

structure for holding all connection parameters for a TCP or UDP connection.

```
#include <ICHIP.H>
```

Data Fields

- [u08 Addr_msb](#)
ip addr. MSB.
- [u08 Addr_2](#)
ip addr.
- [u08 Addr_3](#)
ip addr.
- [u08 Addr_lsb](#)
ip addr. LSB.
- [u08 Port_msb](#)
port number high byte.
- [u08 Port_lsb](#)
port number low byte.

3.1.1 Detailed Description

structure for holding all connection parameters for a TCP or UDP connection.

Definition at line 171 of file ICHIP.H.

3.1.2 Field Documentation

3.1.2.1 **u08 IP_PORT::Addr_2**

ip addr.

Definition at line 176 of file ICHIP.H.

Referenced by Init_Socket().

3.1.2.2 **u08 IP_PORT::Addr_3**

ip addr.

Definition at line 178 of file ICHIP.H.

Referenced by Init_Socket().

3.1.2.3 **u08 IP_PORT::Addr_lsb**

ip addr. LSB.

Definition at line 180 of file ICHIP.H.

Referenced by Init_Socket().

3.1.2.4 **u08 IP_PORT::Addr_msb**

ip addr. MSB.

Definition at line 174 of file ICHIP.H.

Referenced by Init_Socket().

3.1.2.5 **u08 IP_PORT::Port_lsb**

port number low byte.

Definition at line 184 of file ICHIP.H.

Referenced by Init_Socket().

3.1.2.6 **u08 IP_PORT::Port_msb**

port number high byte.

Definition at line 182 of file ICHIP.H.

Referenced by Init_Socket().

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

- [ICHIP.H](#)

Chapter 4

i-easy lib File Documentation

4.1 DEMO.C File Reference

Sample Application for i-easy board V 1.0 (e-mail and time server test).

```
#include <io.h>
#include <interrupt.h>
#include <sig-avr.h>
#include <eeprom.h>
#include <progmem.h>
#include "global.h"
#include "timer.h"
#include "uart.h"
#include "ichip.h"
#include "dial.h"
#include "socket.h"
#include <stdarg.h>
#include <ctype.h>
#include <string.h>
```

Defines

- #define [F_CPU](#) 3686400
- #define [EEPROM_SIZE](#) (E2END+1)

Functions

- void [check_ichip](#) (void)
 - void [time_test](#) (void)
 - [u08_email_test](#) (void)
-

- void [print_ip](#) (void)
- int [main](#) (void)

4.1.1 Detailed Description

Sample Application for i-easy board V 1.0 (e-mail and time server test).

Definition in file [DEMO.C](#).

4.1.2 Define Documentation

4.1.2.1 `#define EEPROM_SIZE (E2END+1)`

Definition at line 45 of file DEMO.C.

4.1.2.2 `#define F_CPU 3686400`

Definition at line 42 of file DEMO.C.

4.1.3 Function Documentation

4.1.3.1 void [check_ichip](#) (void)

send Hello World thru the AVR UART and print out the S7600A version number

Definition at line 49 of file DEMO.C.

References [ReadSeiko\(\)](#), [Revision](#), [sec_delay\(\)](#), [sprintf\(\)](#), [u08](#), and [uart_putstr\(\)](#).

Referenced by [main\(\)](#).

4.1.3.2 [u08 email_test](#) (void)

Example 2: i-easy connects to a sendmailserver (SMTP) using socket 0, read the state of AVR port A and write the result in the e-mail body.

Definition at line 93 of file DEMO.C.

References [CLIENT](#), [FALSE](#), [Init_Socket\(\)](#), [MyIPAddr](#), [sec_delay\(\)](#), [SMTP_PORT_H](#), [SMTP_PORT_L](#), [SOCKET_0](#), [sprintf\(\)](#), [TCP_CLIENT_MODE](#), [Tcp_Close\(\)](#), [Tcp_Connect\(\)](#), [Tcp_Receive\(\)](#), [Tcp_Send\(\)](#), [u08](#), and [uart_putstr\(\)](#).

Referenced by [main\(\)](#).

4.1.3.3 int [main](#) (void)

main program, call the example and preparation functions

Definition at line 171 of file DEMO.C.

References [check_ichip\(\)](#), [connectModem\(\)](#), [disconnectModem\(\)](#), [email_test\(\)](#), [InitSeiko\(\)](#), [PPP_close\(\)](#), [PPP_open\(\)](#), [print_ip\(\)](#), [sec_delay\(\)](#), [time_test\(\)](#), [u08](#), and [uart_init\(\)](#).

4.1.3.4 void print_ip (void)

print out the ip address, given from the PPP Server

Definition at line 161 of file DEMO.C.

References MyIPAddr, sprintf(), u08, and uart_putstr().

Referenced by main().

4.1.3.5 void time_test (void)

Example 1: i-easy connects to a TCP DAY TIME SERVER using socket 1 and print out time and date

Definition at line 63 of file DEMO.C.

References CLIENT, DAYTIME_PORT_H, DAYTIME_PORT_L, FALSE, Init_Socket(), MyIPAddr, sec_delay(), SOCKET_1, TCP_CLIENT_MODE, Tcp_Close(), Tcp_Connect(), Tcp_Receive(), u08, and uart_putstr().

Referenced by main().

4.2 dial.c File Reference

dialup functions for dialing and PPP login (i-easy library).

```
#include "global.h"
#include "timer.h"
#include "uart.h"
#include "ichip.h"
#include "dial.h"
#include <string-avr.h>
```

Functions

- void [atcommand](#) (char *atc)
- [s08 getModemAnswer](#) (u08 block)
- [s08 disconnectModem](#) (void)
- [s08 connectModem](#) (char *iniStr, char *dialStr)
- void [PPP_open](#) (char *name, char *password)
- void [PPP_close](#) (void)

4.2.1 Detailed Description

dialup functions for dialing and PPP login (i-easy library).

Definition in file [dial.c](#).

4.2.2 Function Documentation

4.2.2.1 void atcommand (char *atc) [static]

this function send a AT command to the modem

Parameters:

atc string with AT command + CR

Definition at line 35 of file dial.c.

References [delay\(\)](#), [ReadSeiko\(\)](#), [Serial_Port_Config](#), [Serial_Port_Data](#), and [WriteSeiko\(\)](#).

Referenced by [connectModem\(\)](#), and [disconnectModem\(\)](#).

4.2.2.2 s08 connectModem (char *iniStr, char *dialStr)

this function resets your modem, init your modem with the given modem init string and dial the number with the given dial string

Parameters:

iniStr give here your init string

dialStr give here your dialstring including number (in most cases ATDT + number)

Returns:

status (OK,MODEM_ABSENT,...)

Definition at line 101 of file dial.c.

References AT_RETRY_NUMBER, atcommand(), ATE0V0_RETRY_NUMBER, delay(), disconnectModem(), FAIL_MODEM_CONFIG, getModemAnswer(), MDM_ANSWER, MDM_ANSWER_ANSWER, MDM_BUSY, MDM_BUSY_ANSWER, MDM_CARRIER, MDM_CARRIER_ANSWER, MDM_DIALTONE, MDM_DIALTONE_ANSWER, MDM_ERROR, MDM_ERROR_ANSWER, MDM_RING, MDM_RING_ANSWER, MODEM_ABSENT, OK, s08, and WriteSeiko().

4.2.2.3 s08 disconnectModem (void)

this function hook up the modem and soft reset it

Definition at line 87 of file dial.c.

References atcommand(), getModemAnswer(), OK, and s08.

Referenced by connectModem(), and main().

4.2.2.4 s08 getModemAnswer (u08 block) [static]

this function returns an s08 equal to the numeric value of the modem answer

Parameters:

block for block=0 then ignore modem response

Returns:

retry count value

Definition at line 51 of file dial.c.

References delay(), MODEM_ANSWER_RETRY_NUMBER, ReadSeiko(), s08, TEN, and u08.

Referenced by connectModem(), and disconnectModem().

4.2.2.5 void PPP_close (void)

This function closes the PPP connection . Disabling the PPP should be sufficient under normal conditions but in case we have trouble we have the watchdog timer running and an error handle routine.

Definition at line 240 of file dial.c.

References ReadSeiko(), sec_delay(), u08, and WriteSeiko().

Referenced by main().

4.2.2.6 void PPP_open (char * name, char * password)

program the ppp registers and then start the PPP connection.

Parameters:

name ppp username as string

password ppp password as string

Definition at line 189 of file dial.c.

References MyIPAddr, Our_IP_Address_H, Our_IP_Address_L, Our_IP_Address_M, Our_IP_Address_U, ReadSeiko(), sec_delay(), u08, and WriteSeiko().

4.3 dial.h File Reference

dialup definitions for dialing and PPP login (i-easy library).

Functions

- [s08 connectModem](#) (char *iniStr, char *dialStr)
- [s08 disconnectModem](#) (void)
- void [PPP_open](#) (char *name, char *password)
- void [PPP_close](#) (void)

4.3.1 Detailed Description

dialup definitions for dialing and PPP login (i-easy library).

Definition in file [dial.h](#).

4.3.2 Function Documentation

4.3.2.1 [s08 connectModem](#) (char * *iniStr*, char * *dialStr*)

this function resets your modem, init your modem with the given modem init string and dial the number with the given dial string

Parameters:

iniStr give here your init string

dialStr give here your dialstring including number (in most cases ATDT + number)

Returns:

status (OK,MODEM_ABSENT,...)

Definition at line 101 of file dial.c.

Referenced by main().

4.3.2.2 [s08 disconnectModem](#) (void)

this function hook up the modem and soft reset it

Definition at line 87 of file dial.c.

4.3.2.3 void [PPP_close](#) (void)

This function closes the PPP connection . Disabling the PPP should be sufficient under normal conditions but in case we have trouble we have the watchdog timer running and an error handle routine.

Definition at line 240 of file dial.c.

4.3.2.4 void PPP_open (char * *name*, char * *password*)

program the ppp registers and then start the PPP connection.

Parameters:

name ppp username as string

password ppp password as string

Definition at line 189 of file dial.c.

Referenced by main().

4.4 GLOBAL.H File Reference

global define and typedefs.

Defines

- #define [MAX_U16](#) 65535
- #define [MAX_S16](#) 32767
- #define [IN](#) 0
- #define [OUT](#) 1
- #define [LOW](#) 0
- #define [HIGH](#) 1
- #define [FALLING](#) 0
- #define [RISING](#) 1
- #define [DDR\(x\)](#) ((x)-1)
- #define [PIN\(x\)](#) ((x)-2)
- #define [AVR_MEGA](#) 0

Typedefs

- typedef unsigned char [u08](#)
- typedef char [s08](#)
- typedef unsigned short [u16](#)
- typedef short [s16](#)
- typedef [u08](#) [bool](#)

4.4.1 Detailed Description

global define and typedefs.

Definition in file [GLOBAL.H](#).

4.4.2 Define Documentation

4.4.2.1 #define AVR_MEGA 0

Definition at line 38 of file [GLOBAL.H](#).

4.4.2.2 #define DDR(x) ((x)-1)

Definition at line 25 of file [GLOBAL.H](#).

4.4.2.3 #define FALLING 0

Definition at line 16 of file [GLOBAL.H](#).

4.4.2.4 **#define HIGH 1**

Definition at line 14 of file GLOBAL.H.

4.4.2.5 **#define IN 0**

Definition at line 10 of file GLOBAL.H.

4.4.2.6 **#define LOW 0**

Definition at line 13 of file GLOBAL.H.

4.4.2.7 **#define MAX_S16 32767**

Definition at line 8 of file GLOBAL.H.

4.4.2.8 **#define MAX_U16 65535**

Definition at line 7 of file GLOBAL.H.

Referenced by Tcp_Receive().

4.4.2.9 **#define OUT 1**

Definition at line 11 of file GLOBAL.H.

4.4.2.10 **#define PIN(x) ((x)-2)**

Definition at line 26 of file GLOBAL.H.

4.4.2.11 **#define RISING 1**

Definition at line 17 of file GLOBAL.H.

4.4.3 **Typedef Documentation**

4.4.3.1 **typedef u08 bool**

Definition at line 23 of file GLOBAL.H.

4.4.3.2 **typedef char s08**

Definition at line 20 of file GLOBAL.H.

Referenced by connectModem(), disconnectModem(), and getModemAnswer().

4.4.3.3 typedef short s16

Definition at line 22 of file GLOBAL.H.

Referenced by `uart_getchar()`.

4.4.3.4 typedef unsigned char u08

Definition at line 19 of file GLOBAL.H.

Referenced by `check_ichip()`, `email_test()`, `getModemAnswer()`, `Init_Socket()`, `main()`, `PPP_close()`, `PPP_open()`, `print_ip()`, `S_Putc()`, `send_busy()`, `socket_action()`, `sprintf()`, `Tcp_Close()`, `Tcp_Connect()`, `Tcp_Receive()`, `Tcp_Send()`, `Tcp_State()`, `time_test()`, `uart_getchar()`, `uart_init()`, `uart_putchar()`, and `uart_putstr()`.

4.4.3.5 typedef unsigned short u16

Definition at line 21 of file GLOBAL.H.

Referenced by `delay()`, `S_Putc()`, `sec_delay()`, `sprintf()`, and `Tcp_Receive()`.

4.5 ICHIP.C File Reference

low level access functions (i-easy library).

```
#include <interrupt.h>
#include <io.h>
#include <string-avr.h>
#include "global.h"
#include "timer.h"
#include "ichip.h"
```

Defines

- #define **CSHI** outp((inp(PORTD) | 0x8),PORTD)
take PD-Bit 3 HI.
- #define **CSLO** outp((inp(PORTD) & 0xf7),PORTD)
take PD-Bit 3 LO.
- #define **RSHI** outp((inp(PORTD) | 0x4),PORTD)
take PD-Bit 2 HI.
- #define **RSLO** outp((inp(PORTD) & 0xfb),PORTD)
take PD-Bit 2 LO.
- #define **WRITEXHI** outp((inp(PORTD) | 0x20),PORTD)
take PD-Bit 5 HI.
- #define **WRITEXLO** outp((inp(PORTD) & 0xdf),PORTD)
take PD-Bit 5 LO.
- #define **READXHI** outp((inp(PORTD) | 0x10),PORTD)
take PD-Bit 4 HI.
- #define **READXLO** outp((inp(PORTD) & 0xef),PORTD)
take PD-Bit 4 LO.
- #define **DATAIN** outp(0x00,DDRC)
8bit Data input for port C.
- #define **DATAOUT** outp(0xff,DDRC)
8bit Data output for port C.
- #define **Init_Port_D** outp(0x3c,DDRD);
init the remote port D.
- #define **Init_Port_B** outp(0xFF,DDRB);
init the remote port B.

Functions

- void [InitSeiko](#) (void)
- void [WriteSeiko](#) (char address, char data)
- char [ReadSeiko](#) (char address)
- char [DataAvailable](#) (void)
- void [S_Putc](#) (char *mystring)
- void [W_Putc](#) (char data)

4.5.1 Detailed Description

low level access functions (i-easy library).

Definition in file [ICHIP.C](#).

4.5.2 Define Documentation

4.5.2.1 `#define CSHI outp((inp(PORTD) | 0x8),PORTD)`

take PD-Bit 3 HI.

Definition at line 49 of file [ICHIP.C](#).

Referenced by [ReadSeiko\(\)](#), and [WriteSeiko\(\)](#).

4.5.2.2 `#define CSLO outp((inp(PORTD) & 0xf7),PORTD)`

take PD-Bit 3 LO.

Definition at line 51 of file [ICHIP.C](#).

Referenced by [ReadSeiko\(\)](#), and [WriteSeiko\(\)](#).

4.5.2.3 `#define DATAIN outp(0x00,DDRC)`

8bit Data input for port C.

Definition at line 66 of file [ICHIP.C](#).

Referenced by [InitSeiko\(\)](#), [ReadSeiko\(\)](#), and [WriteSeiko\(\)](#).

4.5.2.4 `#define DATAOUT outp(0xff,DDRC)`

8bit Data output for port C.

Definition at line 68 of file [ICHIP.C](#).

Referenced by [ReadSeiko\(\)](#), and [WriteSeiko\(\)](#).

4.5.2.5 `#define Init_Port_B outp(0xFF,DDRB);`

init the remote port B.

Definition at line 73 of file [ICHIP.C](#).

Referenced by InitSeiko().

4.5.2.6 #define Init_Port_D outp(0x3c,DDRD);

init the remote port D.

Definition at line 71 of file ICHIP.C.

Referenced by InitSeiko().

4.5.2.7 #define READXHI outp((inp(PORTD) | 0x10),PORTD)

take PD-Bit 4 HI.

Definition at line 61 of file ICHIP.C.

Referenced by ReadSeiko(), and WriteSeiko().

4.5.2.8 #define READXLO outp((inp(PORTD) & 0xef),PORTD)

take PD-Bit 4 LO.

Definition at line 63 of file ICHIP.C.

Referenced by ReadSeiko(), and WriteSeiko().

4.5.2.9 #define RSHI outp((inp(PORTD) | 0x4),PORTD)

take PD-Bit 2 HI.

Definition at line 53 of file ICHIP.C.

Referenced by ReadSeiko(), and WriteSeiko().

4.5.2.10 #define RSLO outp((inp(PORTD) & 0xfb),PORTD)

take PD-Bit 2 LO.

Definition at line 55 of file ICHIP.C.

Referenced by ReadSeiko(), and WriteSeiko().

4.5.2.11 #define WRITEXHI outp((inp(PORTD) | 0x20),PORTD)

take PD-Bit 5 HI.

Definition at line 57 of file ICHIP.C.

Referenced by ReadSeiko(), and WriteSeiko().

4.5.2.12 #define WRITEXLO outp((inp(PORTD) & 0xdf),PORTD)

take PD-Bit 5 LO.

Definition at line 59 of file ICHIP.C.

Referenced by ReadSeiko(), and WriteSeiko().

4.5.3 Function Documentation

4.5.3.1 char DataAvailable (void)

Read the Serial Config Register of S7600A

Returns:

value of the Serial Port Config Register

Definition at line 201 of file ICHIP.C.

References ReadSeiko(), and Serial_Port_Config.

4.5.3.2 void InitSeiko (void)

Initialized the S7600A and setup the Baud and Clock Registers

Definition at line 79 of file ICHIP.C.

References BAUD_Rate_Div_H, BAUD_Rate_Div_L, BaudHigh, BaudLow, Clock_Div_H, Clock_Div_L, DATAIN, delay(), DivCountHigh, DivCountLow, General_Control, Init_Port_B, Init_Port_D, ReadSeiko(), Serial_Port_Config, Serial_Port_Int_Mask, and WriteSeiko().

Referenced by main().

4.5.3.3 char ReadSeiko (char *address*)

Read a Byte from a given Address of the S7600A

Parameters:

address an S7600A register address

Returns:

DATA of the given address

Definition at line 147 of file ICHIP.C.

References CSHI, CSLO, DATAIN, DATAOUT, READXHI, READXLO, RSHI, RSLO, WRITEXHI, and WRITEXLO.

4.5.3.4 void S_Putc (char * *string*)

Send String direct to UART of S7600A

Parameters:

mystring pointer to your string to send

Definition at line 210 of file ICHIP.C.

References ReadSeiko(), Serial_Port_Data, Serial_Port_Int, Serial_Port_Int_Mask, u08, u16, and WriteSeiko().

4.5.3.5 void W_Putc (char *data*)

Send Char direct to TCP_Data_Send Register of S7600A

Parameters:

data Char to send

Definition at line 233 of file ICHIP.C.

References ReadSeiko(), Socket_Data, Socket_Status_H, TCP_Data_Send, and WriteSeiko().

4.5.3.6 void WriteSeiko (char *address*, char *data*)

Write a Byte to a given Address of the S7600A

Parameters:

address an S7600A register address

data an DATA value write to the given address

Definition at line 108 of file ICHIP.C.

References CSHI, CSLO, DATAIN, DATAOUT, READXHI, READXLO, RSHI, RSLO, WRITEXHI, and WRITEXLO.

4.6 ICHIP.H File Reference

low level access definitions (i-easy library).

Data Structures

- struct [IP_PORT](#)
structure for holding all connection parameters for a TCP or UDP connection.

Defines

- #define [Revision](#) 0x00
- #define [General_Control](#) 0x01
- #define [General_Socket_Location](#) 0x02
- #define [Master_Interrupt](#) 0x04
- #define [Serial_Port_Config](#) 0x08
- #define [Serial_Port_Int](#) 0x09
- #define [Serial_Port_Int_Mask](#) 0x0a
- #define [Serial_Port_Data](#) 0x0b
- #define [BAUD_Rate_Div_L](#) 0x0c
- #define [BAUD_Rate_Div_H](#) 0x0d
- #define [Our_IP_Address_L](#) 0x10
- #define [Our_IP_Address_M](#) 0x11
- #define [Our_IP_Address_H](#) 0x12
- #define [Our_IP_Address_U](#) 0x13
- #define [Clock_Div_L](#) 0x1c
- #define [Clock_Div_H](#) 0x1d
- #define [Socket_Index](#) 0x20
- #define [Socket_TOS](#) 0x21
- #define [Socket_Config_Status_L](#) 0x22
- #define [Socket_Status_M](#) 0x23
- #define [Socket_Activate](#) 0x24
- #define [Socket_Interrupt](#) 0x26
- #define [Socket_Data_Avail](#) 0x28
- #define [Socket_Interrupt_Mask_L](#) 0x2a
- #define [Socket_Interrupt_Mask_H](#) 0x2b
- #define [Socket_Interrupt_L](#) 0x2c
- #define [Socket_Interrupt_H](#) 0x2d
- #define [Socket_Data](#) 0x2e
- #define [TCP_Data_Send](#) 0x30
- #define [Buffer_Out_L](#) 0x30
- #define [Buffer_Out_H](#) 0x31
- #define [Buffer_In_L](#) 0x32
- #define [Buffer_In_H](#) 0x33
- #define [Urgent_Data_Pointer_L](#) 0x34
- #define [Urgent_Data_Pointer_H](#) 0x35
- #define [Their_Port_L](#) 0x36
- #define [Their_Port_H](#) 0x37

- #define `Our_Port_L` 0x38
- #define `Our_Port_H` 0x39
- #define `Socket_Status_H` 0x3a
- #define `Their_IP_Address_L` 0x3c
- #define `Their_IP_Address_M` 0x3d
- #define `Their_IP_Address_H` 0x3e
- #define `Their_IP_Address_U` 0x3f
- #define `PPP_Control_Status` 0x60
- #define `PPP_Interrupt_Code` 0x61
- #define `PPP_Max_Retry` 0x62
- #define `PAP_String` 0x64
- #define `BaudRate` 19200
- #define `S7600clock` 230400
- #define `DivCountLow` (((S7600clock / 1000) -1) & 0xff)
- #define `DivCountHigh` (((S7600clock / 1000) -1) >> 8) & 0xff)
- #define `BaudDivider` ((S7600clock / BaudRate) -1)
- #define `BaudLow` BaudDivider&0xff
- #define `BaudHigh` BaudDivider>>8
- #define `FALSE` 0
- #define `TRUE` 1
- #define `SOCKET_0` 0x00
- #define `SOCKET_1` 0x01
- #define `CLIENT` 1
- #define `SERVER` 0
- #define `ACTIVATE` 1
- #define `DEACTIVATE` 0
- #define `TCP_CLIENT_MODE` 0x02
- #define `TCP_SERVER_MODE` 0x06
- #define `UDP_MODE` 0x05
- #define `PPP_TIMEOUT` 10
- #define `SNDBSY_TIMEOUT` 10
- #define `TCP_TIMEOUT` 10
- #define `UDP_TIMEOUT` 50
- #define `CLOSED` 0x0
- #define `SYN_SENT` 0x1
- #define `ESTABLISHED` 0x2
- #define `CLOSE_WAIT` 0x3
- #define `LAST_ACK` 0x4
- #define `FIN_WAIT1` 0x5
- #define `FIN_WAIT2` 0x6
- #define `CLOSING` 0x7
- #define `TIME_WAIT` 0x8
- #define `LISTEN` 0x9
- #define `SYN_RECVD` 0xa
- #define `HTTP_PORT_L` 0x50
- #define `HTTP_PORT_H` 0x00
- #define `ECHO_PORT_L` 0x07
- #define `ECHO_PORT_H` 0x00
- #define `DAYTIME_PORT_L` 0x0D
- #define `DAYTIME_PORT_H` 0x00

- #define `SMTP_PORT_L` 0x19
- #define `SMTP_PORT_H` 0x00
- #define `TEN` 10
- #define `SMALL_TIME` 20
- #define `MODEM_ANSWER_RETRY_NUMBER` 20
- #define `AT_RETRY_NUMBER` 12
- #define `ATE0V0_RETRY_NUMBER` 6
- #define `MDM_RING_ANSWER` 2
- #define `MDM_CARRIER_ANSWER` 3
- #define `MDM_ERROR_ANSWER` 4
- #define `MDM_DIALTONE_ANSWER` 6
- #define `MDM_BUSY_ANSWER` 7
- #define `MDM_ANSWER_ANSWER` 8
- #define `OK` 1
- #define `ISRC_STAC_BASE` -100
- #define `ISRC_HW_ABSENT` (`ISRC_STAC_BASE` - 1)
- #define `MODEM_ABSENT` (`ISRC_STAC_BASE` - 2)
- #define `MDM_RING` (`ISRC_STAC_BASE` - 3)
- #define `MDM_BUSY` (`ISRC_STAC_BASE` - 4)
- #define `MDM_CARRIER` (`ISRC_STAC_BASE` - 5)
- #define `MDM_DIALTONE` (`ISRC_STAC_BASE` - 6)
- #define `MDM_ERROR` (`ISRC_STAC_BASE` - 7)
- #define `MDM_ANSWER` (`ISRC_STAC_BASE` - 8)
- #define `BAD_PARAM` (`ISRC_STAC_BASE` - 9)
- #define `FAIL_MODEM_CONFIG` (`ISRC_STAC_BASE` - 10)

Functions

- void `InitSeiko` (void)
- void `WriteSeiko` (char address, char data)
- char `ReadSeiko` (char address)
- char `DataAvailable` (void)
- void `S_Putc` (char *string)
- void `W_Putc` (char data)

Variables

- `u08 MyIPAddr` [6]
after PPP dialup you find in this global field your IP address.

4.6.1 Detailed Description

low level access definitions (i-easy library).

Definition in file `ICHIPH`.

4.6.2 Define Documentation

4.6.2.1 #define ACTIVATE 1

Definition at line 93 of file ICHIP.H.

Referenced by Init_Socket(), socket_action(), and Tcp_Connect().

4.6.2.2 #define AT_RETRY_NUMBER 12

Definition at line 141 of file ICHIP.H.

Referenced by connectModem().

4.6.2.3 #define ATE0V0_RETRY_NUMBER 6

Definition at line 142 of file ICHIP.H.

Referenced by connectModem().

4.6.2.4 #define BAD_PARAM (ISRC_STAC_BASE - 9)

Definition at line 159 of file ICHIP.H.

4.6.2.5 #define BAUD_Rate_Div_H 0x0d

Definition at line 35 of file ICHIP.H.

Referenced by InitSeiko().

4.6.2.6 #define BAUD_Rate_Div_L 0x0c

Definition at line 34 of file ICHIP.H.

Referenced by InitSeiko().

4.6.2.7 #define BaudDivider ((S7600clock / BaudRate) -1)

Definition at line 80 of file ICHIP.H.

4.6.2.8 #define BaudHigh BaudDivider>>8

Definition at line 82 of file ICHIP.H.

Referenced by InitSeiko().

4.6.2.9 #define BaudLow BaudDivider&0xff

Definition at line 81 of file ICHIP.H.

Referenced by InitSeiko().

4.6.2.10 #define BaudRate 19200

Definition at line 75 of file ICHIP.H.

4.6.2.11 #define Buffer_In_H 0x33

Definition at line 58 of file ICHIP.H.

4.6.2.12 #define Buffer_In_L 0x32

Definition at line 57 of file ICHIP.H.

4.6.2.13 #define Buffer_Out_H 0x31

Definition at line 56 of file ICHIP.H.

4.6.2.14 #define Buffer_Out_L 0x30

Definition at line 55 of file ICHIP.H.

4.6.2.15 #define CLIENT 1

Definition at line 91 of file ICHIP.H.

Referenced by email_test(), and time_test().

4.6.2.16 #define Clock_Div_H 0x1d

Definition at line 41 of file ICHIP.H.

Referenced by InitSeiko().

4.6.2.17 #define Clock_Div_L 0x1c

Definition at line 40 of file ICHIP.H.

Referenced by InitSeiko().

4.6.2.18 #define CLOSE_WAIT 0x3

Definition at line 111 of file ICHIP.H.

Referenced by Tcp_Send().

4.6.2.19 #define CLOSED 0x0

Definition at line 108 of file ICHIP.H.

4.6.2.20 #define CLOSING 0x7

Definition at line 115 of file ICHIP.H.

4.6.2.21 #define DAYTIME_PORT_H 0x00

Definition at line 126 of file ICHIP.H.

Referenced by time_test().

4.6.2.22 #define DAYTIME_PORT_L 0x0D

Definition at line 125 of file ICHIP.H.

Referenced by time_test().

4.6.2.23 #define DEACTIVATE 0

Definition at line 94 of file ICHIP.H.

Referenced by socket_action(), and Tcp_Close().

4.6.2.24 #define DivCountHigh (((S7600clock / 1000) - 1) >> 8) & 0xff)

Definition at line 79 of file ICHIP.H.

Referenced by InitSeiko().

4.6.2.25 #define DivCountLow (((S7600clock / 1000) - 1) & 0xff)

Definition at line 78 of file ICHIP.H.

Referenced by InitSeiko().

4.6.2.26 #define ECHO_PORT_H 0x00

Definition at line 124 of file ICHIP.H.

4.6.2.27 #define ECHO_PORT_L 0x07

Definition at line 123 of file ICHIP.H.

4.6.2.28 #define ESTABLISHED 0x2

Definition at line 110 of file ICHIP.H.

Referenced by Tcp_Send().

4.6.2.29 #define FAIL_MODEM_CONFIG (ISRC_STAC_BASE - 10)

Definition at line 160 of file ICHIP.H.

Referenced by connectModem().

4.6.2.30 #define FALSE 0

Definition at line 84 of file ICHIP.H.

Referenced by email_test(), Tcp_Close(), Tcp_Connect(), and time_test().

4.6.2.31 #define FIN_WAIT1 0x5

Definition at line 113 of file ICHIP.H.

4.6.2.32 #define FIN_WAIT2 0x6

Definition at line 114 of file ICHIP.H.

4.6.2.33 #define General_Control 0x01

Definition at line 27 of file ICHIP.H.

Referenced by InitSeiko().

4.6.2.34 #define General_Socket_Location 0x02

Definition at line 28 of file ICHIP.H.

4.6.2.35 #define HTTP_PORT_H 0x00

Definition at line 122 of file ICHIP.H.

4.6.2.36 #define HTTP_PORT_L 0x50

Definition at line 121 of file ICHIP.H.

4.6.2.37 #define ISRC_HW_ABSENT (ISRC_STAC_BASE - 1)

Definition at line 151 of file ICHIP.H.

4.6.2.38 #define ISRC_STAC_BASE -100

Definition at line 150 of file ICHIP.H.

4.6.2.39 #define LAST_ACK 0x4

Definition at line 112 of file ICHIP.H.

4.6.2.40 #define LISTEN 0x9

Definition at line 117 of file ICHIP.H.

Referenced by Tcp_Connect().

4.6.2.41 #define Master_Interrupt 0x04

Definition at line 29 of file ICHIP.H.

4.6.2.42 #define MDM_ANSWER (ISRC_STAC_BASE - 8)

Definition at line 158 of file ICHIP.H.

Referenced by connectModem().

4.6.2.43 #define MDM_ANSWER_ANSWER 8

Definition at line 148 of file ICHIP.H.

Referenced by connectModem().

4.6.2.44 #define MDM_BUSY (ISRC_STAC_BASE - 4)

Definition at line 154 of file ICHIP.H.

Referenced by connectModem().

4.6.2.45 #define MDM_BUSY_ANSWER 7

Definition at line 147 of file ICHIP.H.

Referenced by connectModem().

4.6.2.46 #define MDM_CARRIER (ISRC_STAC_BASE - 5)

Definition at line 155 of file ICHIP.H.

Referenced by connectModem().

4.6.2.47 #define MDM_CARRIER_ANSWER 3

Definition at line 144 of file ICHIP.H.

Referenced by connectModem().

4.6.2.48 #define MDM_DIALTONE (ISRC_STAC_BASE - 6)

Definition at line 156 of file ICHIP.H.

Referenced by connectModem().

4.6.2.49 #define MDM_DIALTONE_ANSWER 6

Definition at line 146 of file ICHIP.H.

Referenced by connectModem().

4.6.2.50 #define MDM_ERROR (ISRC_STAC_BASE - 7)

Definition at line 157 of file ICHIP.H.

Referenced by connectModem().

4.6.2.51 #define MDM_ERROR_ANSWER 4

Definition at line 145 of file ICHIP.H.

Referenced by connectModem().

4.6.2.52 #define MDM_RING (ISRC_STAC_BASE - 3)

Definition at line 153 of file ICHIP.H.

Referenced by connectModem().

4.6.2.53 #define MDM_RING_ANSWER 2

Definition at line 143 of file ICHIP.H.

Referenced by connectModem().

4.6.2.54 #define MODEM_ABSENT (ISRC_STAC_BASE - 2)

Definition at line 152 of file ICHIP.H.

Referenced by connectModem().

4.6.2.55 #define MODEM_ANSWER_RETRY_NUMBER 20

Definition at line 140 of file ICHIP.H.

Referenced by getModemAnswer().

4.6.2.56 #define OK 1

Definition at line 149 of file ICHIP.H.

Referenced by connectModem(), and disconnectModem().

4.6.2.57 #define Our_IP_Address_H 0x12

Definition at line 38 of file ICHIP.H.

Referenced by PPP_open().

4.6.2.58 #define Our_IP_Address_L 0x10

Definition at line 36 of file ICHIP.H.

Referenced by PPP_open().

4.6.2.59 #define Our_IP_Address_M 0x11

Definition at line 37 of file ICHIP.H.

Referenced by PPP_open().

4.6.2.60 #define Our_IP_Address_U 0x13

Definition at line 39 of file ICHIP.H.

Referenced by PPP_open().

4.6.2.61 #define Our_Port_H 0x39

Definition at line 64 of file ICHIP.H.

4.6.2.62 #define Our_Port_L 0x38

Definition at line 63 of file ICHIP.H.

4.6.2.63 #define PAP_String 0x64

Definition at line 73 of file ICHIP.H.

4.6.2.64 #define PPP_Control_Status 0x60

Definition at line 70 of file ICHIP.H.

4.6.2.65 #define PPP_Interrupt_Code 0x61

Definition at line 71 of file ICHIP.H.

4.6.2.66 #define PPP_Max_Retry 0x62

Definition at line 72 of file ICHIP.H.

4.6.2.67 #define PPP_TIMEOUT 10

Definition at line 102 of file ICHIP.H.

4.6.2.68 #define Revision 0x00

Definition at line 26 of file ICHIP.H.

Referenced by check_ichip().

4.6.2.69 #define S7600clock 230400

Definition at line 76 of file ICHIP.H.

4.6.2.70 #define Serial_Port_Config 0x08

Definition at line 30 of file ICHIP.H.

Referenced by atcommand(), DataAvailable(), and InitSeiko().

4.6.2.71 #define Serial_Port_Data 0x0b

Definition at line 33 of file ICHIP.H.

Referenced by atcommand(), and S_Putc().

4.6.2.72 #define Serial_Port_Int 0x09

Definition at line 31 of file ICHIP.H.

Referenced by S_Putc().

4.6.2.73 #define Serial_Port_Int_Mask 0x0a

Definition at line 32 of file ICHIP.H.

Referenced by InitSeiko(), and S_Putc().

4.6.2.74 #define SERVER 0

Definition at line 92 of file ICHIP.H.

Referenced by Tcp_Connect().

4.6.2.75 #define SMALL_TIME 20

Definition at line 139 of file ICHIP.H.

4.6.2.76 #define SMTP_PORT_H 0x00

Definition at line 128 of file ICHIP.H.

Referenced by email_test().

4.6.2.77 #define SMTP_PORT_L 0x19

Definition at line 127 of file ICHIP.H.

Referenced by email_test().

4.6.2.78 #define SNDBSY_TIMEOUT 10

Definition at line 103 of file ICHIP.H.

4.6.2.79 #define SOCKET_0 0x00

Definition at line 89 of file ICHIP.H.

Referenced by email_test(), and socket_action().

4.6.2.80 #define SOCKET_1 0x01

Definition at line 90 of file ICHIP.H.

Referenced by socket_action(), and time_test().

4.6.2.81 #define Socket_Activate 0x24

Definition at line 46 of file ICHIP.H.

4.6.2.82 #define Socket_Config_Status_L 0x22

Definition at line 44 of file ICHIP.H.

4.6.2.83 #define Socket_Data 0x2e

Definition at line 53 of file ICHIP.H.

Referenced by W_Putc().

4.6.2.84 #define Socket_Data_Avail 0x28

Definition at line 48 of file ICHIP.H.

4.6.2.85 #define Socket_Index 0x20

Definition at line 42 of file ICHIP.H.

4.6.2.86 #define Socket_Interrupt 0x26

Definition at line 47 of file ICHIP.H.

4.6.2.87 #define Socket_Interrupt_H 0x2d

Definition at line 52 of file ICHIP.H.

4.6.2.88 #define Socket_Interrupt_L 0x2c

Definition at line 51 of file ICHIP.H.

4.6.2.89 #define Socket_Interrupt_Mask_H 0x2b

Definition at line 50 of file ICHIP.H.

4.6.2.90 #define Socket_Interrupt_Mask_L 0x2a

Definition at line 49 of file ICHIP.H.

4.6.2.91 #define Socket_Status_H 0x3a

Definition at line 65 of file ICHIP.H.

Referenced by W_Putc().

4.6.2.92 #define Socket_Status_M 0x23

Definition at line 45 of file ICHIP.H.

4.6.2.93 #define Socket_TOS 0x21

Definition at line 43 of file ICHIP.H.

4.6.2.94 #define SYN_RECVD 0xa

Definition at line 118 of file ICHIP.H.

4.6.2.95 #define SYN_SENT 0x1

Definition at line 109 of file ICHIP.H.

4.6.2.96 #define TCP_CLIENT_MODE 0x02

Definition at line 97 of file ICHIP.H.

Referenced by email_test(), Init_Socket(), and time_test().

4.6.2.97 #define TCP_Data_Send 0x30

Definition at line 54 of file ICHIP.H.

Referenced by W_Putc().

4.6.2.98 #define TCP_SERVER_MODE 0x06

Definition at line 98 of file ICHIP.H.

Referenced by Init.Socket().

4.6.2.99 #define TCP_TIMEOUT 10

Definition at line 104 of file ICHIP.H.

Referenced by Tcp_Close(), and Tcp_Send().

4.6.2.100 #define TEN 10

Definition at line 135 of file ICHIP.H.

Referenced by getModemAnswer().

4.6.2.101 #define Their_IP_Address_H 0x3e

Definition at line 68 of file ICHIP.H.

4.6.2.102 #define Their_IP_Address_L 0x3c

Definition at line 66 of file ICHIP.H.

4.6.2.103 #define Their_IP_Address_M 0x3d

Definition at line 67 of file ICHIP.H.

4.6.2.104 #define Their_IP_Address_U 0x3f

Definition at line 69 of file ICHIP.H.

4.6.2.105 #define Their_Port_H 0x37

Definition at line 62 of file ICHIP.H.

4.6.2.106 #define Their_Port_L 0x36

Definition at line 61 of file ICHIP.H.

4.6.2.107 #define TIME_WAIT 0x8

Definition at line 116 of file ICHIP.H.

Referenced by Tcp_Close().

4.6.2.108 #define TRUE 1

Definition at line 85 of file ICHIP.H.

Referenced by Tcp_Close(), and Tcp_Connect().

4.6.2.109 #define UDP_MODE 0x05

Definition at line 99 of file ICHIP.H.

Referenced by Init_Socket().

4.6.2.110 #define UDP_TIMEOUT 50

Definition at line 105 of file ICHIP.H.

4.6.2.111 #define Urgent_Data_Pointer_H 0x35

Definition at line 60 of file ICHIP.H.

4.6.2.112 #define Urgent_Data_Pointer_L 0x34

Definition at line 59 of file ICHIP.H.

4.6.3 Function Documentation**4.6.3.1 char DataAvailable (void)**

Read the Serial Config Register of S7600A

Returns:

value of the Serial Port Config Register

Definition at line 201 of file ICHIP.C.

4.6.3.2 void InitSeiko (void)

Initialized the S7600A and setup the Baud and Clock Registers

Definition at line 79 of file ICHIP.C.

4.6.3.3 char ReadSeiko (char *address*)

Read a Byte from a given Address of the S7600A

Parameters:

address an S7600A register address

Returns:

DATA of the given address

Definition at line 147 of file ICHIP.C.

Referenced by atcommand(), check_ichip(), DataAvailable(), getModemAnswer(), InitSeiko(), PPP_close(), PPP_open(), S_Putc(), send_busy(), socket_action(), Tcp_Close(), Tcp_Connect(), Tcp_Receive(), Tcp_Send(), Tcp_State(), and W_Putc().

4.6.3.4 void S_Putc (char * *string*)

Send String direct to UART of S7600A

Parameters:

mystring pointer to your string to send

Definition at line 210 of file ICHIP.C.

4.6.3.5 void W_Putc (char *data*)

Send Char direct to TCP_Data_Send Register of S7600A

Parameters:

data Char to send

Definition at line 233 of file ICHIP.C.

4.6.3.6 void WriteSeiko (char *address*, char *data*)

Write a Byte to a given Address of the S7600A

Parameters:

address an S7600A register address

data an DATA value write to the given address

Definition at line 108 of file ICHIP.C.

Referenced by atcommand(), connectModem(), Init_Socket(), InitSeiko(), PPP_close(), PPP_open(), S_Putc(), Tcp_Close(), Tcp_Connect(), Tcp_Send(), and W_Putc().

4.6.4 Variable Documentation

4.6.4.1 u08 MyIPAddr[6]

after PPP dialup you find in this global field your IP address.

Definition at line 166 of file ICHIP.H.

Referenced by `email_test()`, `PPP_open()`, `print_ip()`, and `time_test()`.

4.7 SOCKET.C File Reference

socket and TCP/IP connection functions (i-easy library).

```
#include "global.h"
#include "timer.h"
#include "uart.h"
#include "ichip.h"
#include "dial.h"
#include "socket.h"
#include <string-avr.h>
```

Functions

- void [send_busy](#) (void)
- [u08 socket_action](#) ([u08](#) socket_number, int action)
- void [Init_Socket](#) (int mode, [u08](#) socket_number, [IP_PORT](#) our_ip, [IP_PORT](#) their_ip)
- [u08 Tcp_State](#) (void)
- [u08 Tcp_Connect](#) ([u08](#) client_server_flag, [u08](#) socket_number)
- [u08 Tcp_Close](#) ([u08](#) socket_number)
- int [Tcp_Receive](#) (char *rec_string)
- int [Tcp_Send](#) (char *send_string)

4.7.1 Detailed Description

socket and TCP/IP connection functions (i-easy library).

Definition in file [SOCKET.C](#).

4.7.2 Function Documentation

4.7.2.1 void [Init_Socket](#) (int mode, [u08](#) socket_number, [IP_PORT](#) our_ip, [IP_PORT](#) their_ip)

This function initializes a socket (0/1).

Parameters:

mode used mode for socket (TCP_SERVER_MODE, TCP_CLIENT_MODE, UDP_MODE)

socket_number socket number (0 or 1)

our_ip our ip address structure, i-easy IP and Port number you like to use (only important for server mode)

their_ip their ip address structure, the IP and Port number of the host you like to connect (only important for client mode)

Definition at line 93 of file SOCKET.C.

References [ACTIVATE](#), [IP_PORT::Addr_2](#), [IP_PORT::Addr_3](#), [IP_PORT::Addr_lsb](#), [IP_PORT::Addr_msb](#), [IP_PORT::Port_lsb](#), [IP_PORT::Port_msb](#), [send_busy\(\)](#), [socket_action\(\)](#), [TCP_CLIENT_MODE](#), [TCP_SERVER_MODE](#), [u08](#), [UDP_MODE](#), and [WriteSeiko\(\)](#).

4.7.2.2 void send_busy (void)

this function waits for the busy bit to deassert in a certain period of time. The busy bit is important when using sockets - see TCP/UDP in the Application note

Definition at line 35 of file SOCKET.C.

References ReadSeiko(), sec_delay(), and u08.

Referenced by Init_Socket().

4.7.2.3 u08 socket_action (u08 socket_number, int action)

function that calculates the value that needs to be written to register 0x24 for activating/deactivating a socket

Parameters:

socket_number used socket number (0 or 1)

action action parameter (ACTIVATE or DEACTIVATE)

Returns:

the socket value for register 0x24 (Socket_Activate)

Definition at line 52 of file SOCKET.C.

References ACTIVATE, DEACTIVATE, ReadSeiko(), SOCKET_0, SOCKET_1, and u08.

4.7.2.4 u08 Tcp_Close (u08 socket_number)

When closing a TCP connection - if all goes well deactivating the socket should be enough. It recommended to flush the send buffer before closing the connection and monitor the snd_bsy bit before resetting and starting a new one.

Parameters:

socket_number socket number of the socket

Returns:

TRUE means close was successful, FALSE means we had a time-out ...

Definition at line 245 of file SOCKET.C.

References DEACTIVATE, FALSE, ReadSeiko(), sec_delay(), socket_action(), Tcp_State(), TCP_TIMEOUT, TIME_WAIT, TRUE, u08, and WriteSeiko().

4.7.2.5 u08 Tcp_Connect (u08 client_server_flag, u08 socket_number)

starts a TCP/IP connection on the given socket on client mode or switch socket to server mode

Parameters:

client_server_flag client or server connection (1 or 0)

socket_number the socket number (0 or 1)

Returns:

connect status FALSE or TRUE

Definition at line 198 of file SOCKET.C.

References ACTIVATE, delay(), FALSE, LISTEN, ReadSeiko(), SERVER, socket_action(), Tcp_State(), TRUE, u08, and WriteSeiko().

4.7.2.6 int Tcp_Receive (char * *rec_string*)

reads socket data, write it to the given field pointer (take care that the field is big enough)

Parameters:

rec_string empty string pointer, big enough to hold received data

Returns:

number of received bytes

Definition at line 281 of file SOCKET.C.

References delay(), MAX_U16, ReadSeiko(), u08, u16, and uart_putchar().

4.7.2.7 int Tcp_Send (char * *send_string*)

send Socket data, Then check the send buffer size write as much data as we can until the buffer is full and hit send! In TCP as long as there is room in the buffer we can write data - unlike UDP where we need to wait until the buffer is empty.

Parameters:

send_string string with data to send

Returns:

number of sent bytes

Definition at line 323 of file SOCKET.C.

References CLOSE_WAIT, ESTABLISHED, ReadSeiko(), sec_delay(), Tcp_State(), TCP_TIMEOUT, u08, and WriteSeiko().

4.7.2.8 u08 Tcp_State (void)

this function returns the state of the TCP state machine

Returns:

state of the TCP connection (show definitions CLOSED,ESTABLISHED,LISTEN...)

Definition at line 155 of file SOCKET.C.

References ReadSeiko(), and u08.

Referenced by Tcp_Close(), Tcp_Connect(), and Tcp_Send().

4.8 SOCKET.H File Reference

socket and TCP/IP connection definitions (i-easy library).

Functions

- void `send_busy` (void)
- `u08 socket_action` (`u08` socket_number, int action)
- void `Init_Socket` (int mode, `u08` socket_number, `IP_PORT` our_ip, `IP_PORT` their_ip)
- `u08 Tcp_State` (void)
- `u08 Tcp_Connect` (`u08` client_server_flag, `u08` socket_number)
- `u08 Tcp_Close` (`u08` socket_number)
- int `Tcp_Receive` (char *rec_string)
- int `Tcp_Send` (char *send_string)

4.8.1 Detailed Description

socket and TCP/IP connection definitions (i-easy library).

Definition in file [SOCKET.H](#).

4.8.2 Function Documentation

4.8.2.1 void `Init_Socket` (int mode, `u08` socket_number, `IP_PORT` our_ip, `IP_PORT` their_ip)

This function initializes a socket (0/1).

Parameters:

mode used mode for socket (TCP_SERVER_MODE, TCP_CLIENT_MODE, UDP_MODE)

socket_number socket number (0 or 1)

our_ip our ip address structure, i-easy IP and Port number you like to use (only important for server mode)

their_ip their ip address structure, the IP and Port number of the host you like to connect (only important for client mode)

Definition at line 93 of file SOCKET.C.

Referenced by `email_test()`, and `time_test()`.

4.8.2.2 void `send_busy` (void)

this function waits for the busy bit to deassert in a certain period of time. The busy bit is important when using sockets - see TCP/UDP in the Application note

Definition at line 35 of file SOCKET.C.

4.8.2.3 **u08** socket_action (**u08** socket_number, int action)

function that calculates the value that needs to be written to register 0x24 for activating/deactivating a socket

Parameters:

socket_number used socket number (0 or 1)

action action parameter (ACTIVATE or DEACTIVATE)

Returns:

the socket value for register 0x24 (Socket_Activate)

Definition at line 52 of file SOCKET.C.

Referenced by Init_Socket(), Tcp_Close(), and Tcp_Connect().

4.8.2.4 **u08** Tcp_Close (**u08** socket_number)

When closing a TCP connection - if all goes well deactivating the socket should be enough. It recommended to flush the send buffer before closing the connection and monitor the snd_bsy bit before resetting and starting a new one.

Parameters:

socket_number socket number of the socket

Returns:

TRUE means close was successful, FALSE means we had a time-out ...

Definition at line 245 of file SOCKET.C.

Referenced by email_test(), and time_test().

4.8.2.5 **u08** Tcp_Connect (**u08** client_server_flag, **u08** socket_number)

starts a TCP/IP connection on the given socket on client mode or switch socket to server mode

Parameters:

client_server_flag client or server connection (1 or 0)

socket_number the socket number (0 or 1)

Returns:

connect status FALSE or TRUE

Definition at line 198 of file SOCKET.C.

Referenced by email_test(), and time_test().

4.8.2.6 **int** Tcp_Receive (**char *** rec_string)

reads socket data, write it to the given field pointer (take care that the field is big enough)

Parameters:

rec_string empty string pointer, big enough to hold received data

Returns:

number of received bytes

Definition at line 281 of file SOCKET.C.

Referenced by email_test(), and time_test().

4.8.2.7 int Tcp_Send (char * *send_string*)

send Socket data, Then check the send buffer size write as much data as we can until the buffer is full and hit send! In TCP as long as there is room in the buffer we can write data - unlike UDP where we need to wait until the buffer is empty.

Parameters:

send_string string with data to send

Returns:

number of sent bytes

Definition at line 323 of file SOCKET.C.

Referenced by email_test().

4.8.2.8 u08 Tcp_State (void)

this function returns the state of the TCP state machine

Returns:

state of the TCP connection (show definitions CLOSED,ESTABLISHED,LISTEN...)

Definition at line 155 of file SOCKET.C.

4.9 TIMER.C File Reference

timer library by Volker Oth (modified).

```
#include <io.h>
#include "global.h"
#include "timer.h"
```

Functions

- void [delay](#) ([u16](#) us)
- void [sec_delay](#) ([u16](#) sec)

4.9.1 Detailed Description

timer library by Volker Oth (modified).

Definition in file [TIMER.C](#).

4.9.2 Function Documentation

4.9.2.1 void [delay](#) ([u16](#) *us*)

[delay](#) for a minimum of `<us>` microseconds the time resolution is dependent on the time the loop takes e.g. with 4Mhz and 5 cycles per loop, the resolution is 1.25 us

Parameters:

us delay in microseconds

Definition at line 30 of file [TIMER.C](#).

References [CYCLES_PER_US](#), and [u16](#).

4.9.2.2 void [sec_delay](#) ([u16](#) *sec*)

[delay](#) for a minimum of `<sec>` seconds the time resolution is dependent on the delay function

Parameters:

sec delay in seconds

Definition at line 48 of file [TIMER.C](#).

References [delay\(\)](#), and [u16](#).

4.10 TIMER.H File Reference

timer library by Volker Oth (modified).

Defines

- #define `F_CPU` 3686400
3686400 Hz processor (Crystal frequency).
- #define `CYCLES_PER_US` ((F_CPU+500000)/1000000)

Functions

- void `delay` (u16 us)
- void `sec_delay` (u16 sec)

4.10.1 Detailed Description

timer library by Volker Oth (modified).

Definition in file [TIMER.H](#).

4.10.2 Define Documentation

4.10.2.1 #define `CYCLES_PER_US` ((F_CPU+500000)/1000000)

auto calculation doesn't work for prescalers /1 (1) and /8 (2) cpu cycles per microsecond

Definition at line 29 of file [TIMER.H](#).

Referenced by `delay()`.

4.10.2.2 #define `F_CPU` 3686400

3686400 Hz processor (Crystal frequency).

Definition at line 25 of file [TIMER.H](#).

4.10.3 Function Documentation

4.10.3.1 void `delay` (u16 us)

delay for a minimum of <us> microseconds the time resolution is dependent on the time the loop takes
e.g. with 4Mhz and 5 cycles per loop, the resolution is 1.25 us

Parameters:

us delay in microseconds

Definition at line 30 of file [TIMER.C](#).

References `u16`.

Referenced by `atcommand()`, `connectModem()`, `getModemAnswer()`, `InitSeiko()`, `sec_delay()`, `Tcp_Connect()`, `Tcp_Receive()`, and `uart_putstr()`.

4.10.3.2 void `sec_delay` (**u16** *sec*)

delay for a minimum of `<sec>` seconds the time resolution is dependent on the delay function

Parameters:

sec delay in seconds

Definition at line 48 of file `TIMER.C`.

Referenced by `check_ichip()`, `email_test()`, `main()`, `PPP_close()`, `PPP_open()`, `send_busy()`, `Tcp_Close()`, `Tcp_Send()`, and `time_test()`.

4.11 UART.C File Reference

UART library by Volker Oth (modified).

```
#include <io.h>
#include <sig-avr.h>
#include <interrupt.h>
#include "global.h"
#include "timer.h"
#include "uart.h"
#include <stdarg.h>
#include <ctype.h>
#include <string-avr.h>
```

Functions

- void [uart_init](#) (void)
- [SIGNAL](#) (SIG_UART_DATA)
- [s16 uart_getchar](#) (void)
- [bool uart_putchar](#) ([u08](#) c)
- [bool uart_putstr](#) ([u08](#) s[])
- void [uart_clr](#) (void)
- void [uart_nl](#) (void)
- void [sprintf](#) ([u08](#) *buf, const [u08](#) *format,...)

Variables

- volatile [u08 uart_txd_buf_cnt](#)
- volatile [u08 uart_rxd_buf_cnt](#)
- [u08 * uart_txd_in_ptr](#)
- [u08 * uart_txd_out_ptr](#)
- [u08 * uart_rxd_in_ptr](#)
- [u08 * uart_rxd_out_ptr](#)
- [u08 UART_CLR](#) [] = {ESC, '[','H', ESC, '[','2', 'J',0}
- [u08 UART_NL](#) [] = {0x0d,0x0a,0}
- [u08 uart_txd_buffer](#) [UART_BUF_SIZE]
- [u08 uart_rxd_buffer](#) [UART_BUF_SIZE]

4.11.1 Detailed Description

UART library by Volker Oth (modified).

Definition in file [UART.C](#).

4.11.2 Function Documentation

4.11.2.1 SIGNAL (SIG_UART_DATA)

signal handler for uart data buffer empty interrupt

Definition at line 57 of file UART.C.

References UART_BUF_SIZE, uart_rxd_buf_cnt, uart_rxd_buffer, uart_rxd_in_ptr, uart_txd_buf_cnt, uart_txd_buffer, and uart_txd_out_ptr.

4.11.2.2 void sprintf (u08 * buf, const u08 * format, ...)

simplified sprintf

Definition at line 139 of file UART.C.

References SCRATCH, u08, and u16.

4.11.2.3 void uart_clr (void)

Send a 'clear screen' to a VT100 terminal

Definition at line 126 of file UART.C.

References UART_CLR, and uart_putstr().

4.11.2.4 s16 uart_getchar (void)

Definition at line 79 of file UART.C.

References s16, u08, UART_BUF_SIZE, uart_rxd_buf_cnt, uart_rxd_buffer, and uart_rxd_out_ptr.

4.11.2.5 void uart_init (void)

initialize uart of AVR controller

Definition at line 42 of file UART.C.

References u08, UART_BAUD_SELECT, uart_rxd_buf_cnt, uart_rxd_buffer, uart_rxd_in_ptr, uart_rxd_out_ptr, uart_txd_buf_cnt, uart_txd_buffer, uart_txd_in_ptr, and uart_txd_out_ptr.

Referenced by main().

4.11.2.6 void uart_nl (void)

Send a 'new line'

Definition at line 132 of file UART.C.

References UART_NL, and uart_putstr().

4.11.2.7 bool uart_putchar (u08 c)

Definition at line 97 of file UART.C.

References `u08`, `UART_BUF_SIZE`, `uart_txd_buf_cnt`, `uart_txd_buffer`, and `uart_txd_in_ptr`.

4.11.2.8 `bool` `uart_putstr` (`u08` `s`[])

Definition at line 114 of file `UART.C`.

References `delay()`, `u08`, and `uart_putchar()`.

4.11.3 Variable Documentation

4.11.3.1 `u08` `UART_CLR`[] = {`ESC`, `'[`, `'H`, `ESC`, `'[`, `'2`, `'J`, `0`}

Definition at line 34 of file `UART.C`.

Referenced by `uart_clr()`.

4.11.3.2 `u08` `UART_NL`[] = {`0x0d`, `0x0a`, `0`}

Definition at line 35 of file `UART.C`.

Referenced by `uart_nl()`.

4.11.3.3 `volatile u08` `uart_rxd_buf_cnt`

Definition at line 31 of file `UART.C`.

Referenced by `SIGNAL()`, `uart_getchar()`, and `uart_init()`.

4.11.3.4 `u08` `uart_rxd_buffer`[`UART_BUF_SIZE`]

Definition at line 37 of file `UART.C`.

Referenced by `SIGNAL()`, `uart_getchar()`, and `uart_init()`.

4.11.3.5 `u08*` `uart_rxd_in_ptr`

Definition at line 33 of file `UART.C`.

Referenced by `SIGNAL()`, and `uart_init()`.

4.11.3.6 `u08 *` `uart_rxd_out_ptr`

Definition at line 33 of file `UART.C`.

Referenced by `uart_getchar()`, and `uart_init()`.

4.11.3.7 `volatile u08` `uart_txd_buf_cnt`

Definition at line 30 of file `UART.C`.

Referenced by `SIGNAL()`, `uart_init()`, and `uart_putchar()`.

4.11.3.8 u08 uart_txd_buffer[UART_BUF_SIZE]

Definition at line 36 of file UART.C.

Referenced by SIGNAL(), uart_init(), and uart_putchar().

4.11.3.9 u08* uart_txd_in_ptr

Definition at line 32 of file UART.C.

Referenced by uart_init(), and uart_putchar().

4.11.3.10 u08 * uart_txd_out_ptr

Definition at line 32 of file UART.C.

Referenced by SIGNAL(), and uart_init().

4.12 UART.H File Reference

UART library by Volker Oth (modified).

Defines

- #define [F_CPU](#) 3686400
- #define [UART_BAUD_RATE](#) 115200
- #define [ESC](#) 0x1b
- #define [UART_BUF_SIZE](#) 16
- #define [SCRATCH](#) 16
- #define [UART_BAUD_SELECT](#) (F_CPU/(UART_BAUD_RATE*16l)-1)

Functions

- void [uart_init](#) (void)
- void [uart_clr](#) (void)
- void [uart_nl](#) (void)
- bool [uart_putchar](#) (u08 c)
- s16 [uart_getchar](#) (void)
- bool [uart_putstr](#) (u08 s[])
- void [sprintf](#) (u08 *buf, const u08 *format,...)

4.12.1 Detailed Description

UART library by Volker Oth (modified).

Definition in file [UART.H](#).

4.12.2 Define Documentation

4.12.2.1 #define ESC 0x1b

Definition at line 25 of file UART.H.

4.12.2.2 #define F_CPU 3686400

Definition at line 21 of file UART.H.

4.12.2.3 #define SCRATCH 16

Definition at line 29 of file UART.H.

Referenced by [sprintf\(\)](#).

4.12.2.4 #define UART_BAUD_RATE 115200

Definition at line 24 of file UART.H.

4.12.2.5 **#define UART_BAUD_SELECT (F_CPU/(UART_BAUD_RATE*16l)-1)**

Definition at line 32 of file UART.H.

Referenced by `uart_init()`.

4.12.2.6 **#define UART_BUF_SIZE 16**

Definition at line 26 of file UART.H.

Referenced by `SIGNAL()`, `uart_getchar()`, and `uart_putchar()`.

4.12.3 **Function Documentation**

4.12.3.1 **void sprintf (u08 * buf, const u08 * format, ...)**

simplified sprintf

Definition at line 139 of file UART.C.

Referenced by `check_ichip()`, `email_test()`, and `print_ip()`.

4.12.3.2 **void uart_clr (void)**

Send a 'clear screen' to a VT100 terminal

Definition at line 126 of file UART.C.

4.12.3.3 **s16 uart_getchar (void)**

Definition at line 79 of file UART.C.

4.12.3.4 **void uart_init (void)**

initialize uart of AVR controller

Definition at line 42 of file UART.C.

References `u08`.

4.12.3.5 **void uart_nl (void)**

Send a 'new line'

Definition at line 132 of file UART.C.

4.12.3.6 **bool uart_putchar (u08 c)**

Definition at line 97 of file UART.C.

Referenced by `Tcp_Receive()`, and `uart_putstr()`.

4.12.3.7 `bool` `uart_putstr (u08 s[])`

Definition at line 114 of file UART.C.

Referenced by `check_ichip()`, `email_test()`, `print_ip()`, `time_test()`, `uart_clr()`, and `uart_nl()`.

Index

ACTIVATE
 ICHIP.H, 26

Addr_2
 IP_PORT, 6

Addr_3
 IP_PORT, 6

Addr_lsb
 IP_PORT, 6

Addr_msb
 IP_PORT, 6

AT_RETRY_NUMBER
 ICHIP.H, 26

atcommand
 dial.c, 10

ATE0V0_RETRY_NUMBER
 ICHIP.H, 26

AVR_MEGA
 GLOBAL.H, 15

BAD_PARAM
 ICHIP.H, 26

BAUD_Rate_Div_H
 ICHIP.H, 26

BAUD_Rate_Div_L
 ICHIP.H, 26

BaudDivider
 ICHIP.H, 26

BaudHigh
 ICHIP.H, 26

BaudLow
 ICHIP.H, 26

BaudRate
 ICHIP.H, 26

bool
 GLOBAL.H, 16

Buffer_In_H
 ICHIP.H, 27

Buffer_In_L
 ICHIP.H, 27

Buffer_Out_H
 ICHIP.H, 27

Buffer_Out_L
 ICHIP.H, 27

check_ichip
 DEMO.C, 8

CLIENT
 ICHIP.H, 27

Clock_Div_H
 ICHIP.H, 27

Clock_Div_L
 ICHIP.H, 27

CLOSE_WAIT
 ICHIP.H, 27

CLOSED
 ICHIP.H, 27

CLOSING
 ICHIP.H, 27

connectModem
 dial.c, 10
 dial.h, 13

CSHI
 ICHIP.C, 19

CSLO
 ICHIP.C, 19

CYCLES_PER_US
 TIMER.H, 47

DataAvailable
 ICHIP.C, 21
 ICHIP.H, 37

DATIN
 ICHIP.C, 19

DATAOUT
 ICHIP.C, 19

DAYTIME_PORT_H
 ICHIP.H, 28

DAYTIME_PORT_L
 ICHIP.H, 28

DDR
 GLOBAL.H, 15

DEACTIVATE
 ICHIP.H, 28

delay
 TIMER.C, 46
 TIMER.H, 47

DEMO.C, 7
 check_ichip, 8
 EEPROM_SIZE, 8
 email.test, 8

- F_CPU, 8
 - main, 8
 - print_ip, 8
 - time_test, 9
- dial.c, 10
 - atcommand, 10
 - connectModem, 10
 - disconnectModem, 11
 - getModemAnswer, 11
 - PPP_close, 11
 - PPP_open, 11
- dial.h, 13
 - connectModem, 13
 - disconnectModem, 13
 - PPP_close, 13
 - PPP_open, 13
- disconnectModem
 - dial.c, 11
 - dial.h, 13
- DivCountHigh
 - ICHIP.H, 28
- DivCountLow
 - ICHIP.H, 28
- ECHO_PORT_H
 - ICHIP.H, 28
- ECHO_PORT_L
 - ICHIP.H, 28
- EEPROM_SIZE
 - DEMO.C, 8
- email_test
 - DEMO.C, 8
- ESC
 - UART.H, 53
- ESTABLISHED
 - ICHIP.H, 28
- F_CPU
 - DEMO.C, 8
 - TIMER.H, 47
 - UART.H, 53
- FAIL_MODEM_CONFIG
 - ICHIP.H, 28
- FALLING
 - GLOBAL.H, 15
- FALSE
 - ICHIP.H, 29
- FIN_WAIT1
 - ICHIP.H, 29
- FIN_WAIT2
 - ICHIP.H, 29
- General_Control
 - ICHIP.H, 29
- General_Socket_Location
 - ICHIP.H, 29
- getModemAnswer
 - dial.c, 11
- GLOBAL.H, 15
 - AVR_MEGA, 15
 - bool, 16
 - DDR, 15
 - FALLING, 15
 - HIGH, 15
 - IN, 16
 - LOW, 16
 - MAX_S16, 16
 - MAX_U16, 16
 - OUT, 16
 - PIN, 16
 - RISING, 16
 - s08, 16
 - s16, 16
 - u08, 17
 - u16, 17
- HIGH
 - GLOBAL.H, 15
- HTTP_PORT_H
 - ICHIP.H, 29
- HTTP_PORT_L
 - ICHIP.H, 29
- ICHIP.C, 18
 - CSHI, 19
 - CSLO, 19
 - DataAvailable, 21
 - DATAIN, 19
 - DATAOUT, 19
 - Init_Port_B, 19
 - Init_Port_D, 20
 - InitSeiko, 21
 - ReadSeiko, 21
 - READXHI, 20
 - READXLO, 20
 - RSHI, 20
 - RSLO, 20
 - S_Putc, 21
 - W_Putc, 21
 - WriteSeiko, 22
 - WRITEXHI, 20
 - WRITEXLO, 20
- ICHIP.H, 23
 - ACTIVATE, 26
 - AT_RETRY_NUMBER, 26
 - ATE0V0_RETRY_NUMBER, 26
 - BAD_PARAM, 26
 - BAUD_Rate_Div_H, 26

- BAUD_Rate_Div_L, 26
- BaudDivider, 26
- BaudHigh, 26
- BaudLow, 26
- BaudRate, 26
- Buffer_In_H, 27
- Buffer_In_L, 27
- Buffer_Out_H, 27
- Buffer_Out_L, 27
- CLIENT, 27
- Clock_Div_H, 27
- Clock_Div_L, 27
- CLOSE_WAIT, 27
- CLOSED, 27
- CLOSING, 27
- DataAvailable, 37
- DAYTIME_PORT_H, 28
- DAYTIME_PORT_L, 28
- DEACTIVATE, 28
- DivCountHigh, 28
- DivCountLow, 28
- ECHO_PORT_H, 28
- ECHO_PORT_L, 28
- ESTABLISHED, 28
- FAIL_MODEM_CONFIG, 28
- FALSE, 29
- FIN_WAIT1, 29
- FIN_WAIT2, 29
- General_Control, 29
- General.Socket_Location, 29
- HTTP_PORT_H, 29
- HTTP_PORT_L, 29
- InitSeiko, 37
- ISRC_HW_ABSENT, 29
- ISRC_STAC_BASE, 29
- LAST_ACK, 29
- LISTEN, 30
- Master_Interrupt, 30
- MDM_ANSWER, 30
- MDM_ANSWER_ANSWER, 30
- MDM_BUSY, 30
- MDM_BUSY_ANSWER, 30
- MDM_CARRIER, 30
- MDM_CARRIER_ANSWER, 30
- MDM_DIALTONE, 30
- MDM_DIALTONE_ANSWER, 31
- MDM_ERROR, 31
- MDM_ERROR_ANSWER, 31
- MDM_RING, 31
- MDM_RING_ANSWER, 31
- MODEM_ABSENT, 31
- MODEM_ANSWER_RETRY_NUMBER,
31
- MyIPAddr, 38
- OK, 31
- Our_IP_Address_H, 31
- Our_IP_Address_L, 32
- Our_IP_Address_M, 32
- Our_IP_Address_U, 32
- Our_Port_H, 32
- Our_Port_L, 32
- PAP_String, 32
- PPP_Control_Status, 32
- PPP_Interrupt_Code, 32
- PPP_Max_Retry, 32
- PPP_TIMEOUT, 32
- ReadSeiko, 37
- Revision, 33
- S7600clock, 33
- S_Putc, 38
- Serial_Port_Config, 33
- Serial_Port_Data, 33
- Serial_Port_Int, 33
- Serial_Port_Int_Mask, 33
- SERVER, 33
- SMALL_TIME, 33
- SMTP_PORT_H, 33
- SMTP_PORT_L, 34
- SNDBSY_TIMEOUT, 34
- SOCKET_0, 34
- SOCKET_1, 34
- Socket_Activate, 34
- Socket_Config_Status_L, 34
- Socket_Data, 34
- Socket_Data_Avail, 34
- Socket_Index, 34
- Socket_Interrupt, 34
- Socket_Interrupt_H, 35
- Socket_Interrupt_L, 35
- Socket_Interrupt_Mask_H, 35
- Socket_Interrupt_Mask_L, 35
- Socket_Status_H, 35
- Socket_Status_M, 35
- Socket_TOS, 35
- SYN_RECV, 35
- SYN_SENT, 35
- TCP_CLIENT_MODE, 35
- TCP_Data_Send, 35
- TCP_SERVER_MODE, 36
- TCP_TIMEOUT, 36
- TEN, 36
- Their_IP_Address_H, 36
- Their_IP_Address_L, 36
- Their_IP_Address_M, 36
- Their_IP_Address_U, 36
- Their_Port_H, 36
- Their_Port_L, 36
- TIME_WAIT, 36

- TRUE, 37
- UDP_MODE, 37
- UDP_TIMEOUT, 37
- Urgent_Data_Pointer_H, 37
- Urgent_Data_Pointer_L, 37
- W_Putc, 38
- WriteSeiko, 38
- IN
 - GLOBAL.H, 16
- Init_Port_B
 - ICHIP.C, 19
- Init_Port_D
 - ICHIP.C, 20
- Init_Socket
 - SOCKET.C, 40
 - SOCKET.H, 43
- InitSeiko
 - ICHIP.C, 21
 - ICHIP.H, 37
- IP_PORT, 5
 - Addr_2, 6
 - Addr_3, 6
 - Addr_lsb, 6
 - Addr_msb, 6
 - Port_lsb, 6
 - Port_msb, 6
- ISRC_HW_ABSENT
 - ICHIP.H, 29
- ISRC_STAC_BASE
 - ICHIP.H, 29
- LAST_ACK
 - ICHIP.H, 29
- LISTEN
 - ICHIP.H, 30
- LOW
 - GLOBAL.H, 16
- main
 - DEMO.C, 8
- Master_Interrupt
 - ICHIP.H, 30
- MAX_S16
 - GLOBAL.H, 16
- MAX_U16
 - GLOBAL.H, 16
- MDM_ANSWER
 - ICHIP.H, 30
- MDM_ANSWER_ANSWER
 - ICHIP.H, 30
- MDM_BUSY
 - ICHIP.H, 30
- MDM_BUSY_ANSWER
 - ICHIP.H, 30
- MDM_CARRIER
 - ICHIP.H, 30
- MDM_CARRIER_ANSWER
 - ICHIP.H, 30
- MDM_DIALTONE
 - ICHIP.H, 30
- MDM_DIALTONE_ANSWER
 - ICHIP.H, 31
- MDM_ERROR
 - ICHIP.H, 31
- MDM_ERROR_ANSWER
 - ICHIP.H, 31
- MDM_RING
 - ICHIP.H, 31
- MDM_RING_ANSWER
 - ICHIP.H, 31
- MODEM_ABSENT
 - ICHIP.H, 31
- MODEM_ANSWER_RETRY_NUMBER
 - ICHIP.H, 31
- MyIPAddr
 - ICHIP.H, 38
- OK
 - ICHIP.H, 31
- Our_IP_Address_H
 - ICHIP.H, 31
- Our_IP_Address_L
 - ICHIP.H, 32
- Our_IP_Address_M
 - ICHIP.H, 32
- Our_IP_Address_U
 - ICHIP.H, 32
- Our_Port_H
 - ICHIP.H, 32
- Our_Port_L
 - ICHIP.H, 32
- OUT
 - GLOBAL.H, 16
- PAP_String
 - ICHIP.H, 32
- PIN
 - GLOBAL.H, 16
- Port_lsb
 - IP_PORT, 6
- Port_msb
 - IP_PORT, 6
- PPP_close
 - dial.c, 11
 - dial.h, 13
- PPP_Control_Status
 - ICHIP.H, 32
- PPP_Interrupt_Code

- ICHIP.H, 32
- PPP_Max_Retry
 - ICHIP.H, 32
- PPP_open
 - dial.c, 11
 - dial.h, 13
- PPP_TIMEOUT
 - ICHIP.H, 32
- print_ip
 - DEMO.C, 8
- ReadSeiko
 - ICHIP.C, 21
 - ICHIP.H, 37
- READXHI
 - ICHIP.C, 20
- READXLO
 - ICHIP.C, 20
- Revision
 - ICHIP.H, 33
- RISING
 - GLOBAL.H, 16
- RSHI
 - ICHIP.C, 20
- RSLO
 - ICHIP.C, 20
- s08
 - GLOBAL.H, 16
- s16
 - GLOBAL.H, 16
- S7600clock
 - ICHIP.H, 33
- S_Putc
 - ICHIP.C, 21
 - ICHIP.H, 38
- SCRATCH
 - UART.H, 53
- sec_delay
 - TIMER.C, 46
 - TIMER.H, 48
- send_busy
 - SOCKET.C, 40
 - SOCKET.H, 43
- Serial_Port_Config
 - ICHIP.H, 33
- Serial_Port_Data
 - ICHIP.H, 33
- Serial_Port_Int
 - ICHIP.H, 33
- Serial_Port_Int_Mask
 - ICHIP.H, 33
- SERVER
 - ICHIP.H, 33
- SIGNAL
 - UART.C, 50
- SMALL_TIME
 - ICHIP.H, 33
- SMTP_PORT_H
 - ICHIP.H, 33
- SMTP_PORT_L
 - ICHIP.H, 34
- SNDBSY_TIMEOUT
 - ICHIP.H, 34
- SOCKET.C, 40
 - Init_Socket, 40
 - send_busy, 40
 - socket_action, 41
 - Tcp_Close, 41
 - Tcp_Connect, 41
 - Tcp_Receive, 42
 - Tcp_Send, 42
 - Tcp_State, 42
- SOCKET.H, 43
 - Init_Socket, 43
 - send_busy, 43
 - socket_action, 43
 - Tcp_Close, 44
 - Tcp_Connect, 44
 - Tcp_Receive, 44
 - Tcp_Send, 45
 - Tcp_State, 45
- SOCKET_0
 - ICHIP.H, 34
- SOCKET_1
 - ICHIP.H, 34
- socket_action
 - SOCKET.C, 41
 - SOCKET.H, 43
- Socket_Activate
 - ICHIP.H, 34
- Socket_Config_Status_L
 - ICHIP.H, 34
- Socket_Data
 - ICHIP.H, 34
- Socket_Data_Avail
 - ICHIP.H, 34
- Socket_Index
 - ICHIP.H, 34
- Socket_Interrupt
 - ICHIP.H, 34
- Socket_Interrupt_H
 - ICHIP.H, 35
- Socket_Interrupt_L
 - ICHIP.H, 35
- Socket_Interrupt_Mask_H
 - ICHIP.H, 35
- Socket_Interrupt_Mask_L

- ICHIP.H, 35
- Socket_Status_H
 - ICHIP.H, 35
- Socket_Status_M
 - ICHIP.H, 35
- Socket_TOS
 - ICHIP.H, 35
- sprintf
 - UART.C, 50
 - UART.H, 54
- SYN_RECVD
 - ICHIP.H, 35
- SYN_SENT
 - ICHIP.H, 35
- TCP_CLIENT_MODE
 - ICHIP.H, 35
- Tcp_Close
 - SOCKET.C, 41
 - SOCKET.H, 44
- Tcp_Connect
 - SOCKET.C, 41
 - SOCKET.H, 44
- TCP_Data_Send
 - ICHIP.H, 35
- Tcp_Receive
 - SOCKET.C, 42
 - SOCKET.H, 44
- Tcp_Send
 - SOCKET.C, 42
 - SOCKET.H, 45
- TCP_SERVER_MODE
 - ICHIP.H, 36
- Tcp_State
 - SOCKET.C, 42
 - SOCKET.H, 45
- TCP_TIMEOUT
 - ICHIP.H, 36
- TEN
 - ICHIP.H, 36
- Their_IP_Address_H
 - ICHIP.H, 36
- Their_IP_Address_L
 - ICHIP.H, 36
- Their_IP_Address_M
 - ICHIP.H, 36
- Their_IP_Address_U
 - ICHIP.H, 36
- Their_Port_H
 - ICHIP.H, 36
- Their_Port_L
 - ICHIP.H, 36
- time_test
 - DEMO.C, 9
- TIME_WAIT
 - ICHIP.H, 36
- TIMER.C, 46
 - delay, 46
 - sec_delay, 46
- TIMER.H, 47
 - CYCLES_PER_US, 47
 - delay, 47
 - F_CPU, 47
 - sec_delay, 48
- TRUE
 - ICHIP.H, 37
- u08
 - GLOBAL.H, 17
- u16
 - GLOBAL.H, 17
- UART.C, 49
 - SIGNAL, 50
 - sprintf, 50
 - UART_CLR, 51
 - uart_clr, 50
 - uart_getchar, 50
 - uart_init, 50
 - UART_NL, 51
 - uart_nl, 50
 - uart_putchar, 50
 - uart_putstr, 51
 - uart_rxd_buf_cnt, 51
 - uart_rxd_buffer, 51
 - uart_rxd_in_ptr, 51
 - uart_rxd_out_ptr, 51
 - uart_txd_buf_cnt, 51
 - uart_txd_buffer, 51
 - uart_txd_in_ptr, 52
 - uart_txd_out_ptr, 52
- UART.H, 53
 - ESC, 53
 - F_CPU, 53
 - SCRATCH, 53
 - sprintf, 54
 - UART_BAUD_RATE, 53
 - UART_BAUD_SELECT, 53
 - UART_BUF_SIZE, 54
 - uart_clr, 54
 - uart_getchar, 54
 - uart_init, 54
 - uart_nl, 54
 - uart_putchar, 54
 - uart_putstr, 54
- UART_BAUD_RATE
 - UART.H, 53
- UART_BAUD_SELECT
 - UART.H, 53

- UART_BUF_SIZE
 - UART.H, 54
- UART_CLR
 - UART.C, 51
- uart_clr
 - UART.C, 50
 - UART.H, 54
- uart_getchar
 - UART.C, 50
 - UART.H, 54
- uart_init
 - UART.C, 50
 - UART.H, 54
- UART_NL
 - UART.C, 51
- uart_nl
 - UART.C, 50
 - UART.H, 54
- uart_putchar
 - UART.C, 50
 - UART.H, 54
- uart_putstr
 - UART.C, 51
 - UART.H, 54
- uart_rxd_buf_cnt
 - UART.C, 51
- uart_rxd_buffer
 - UART.C, 51
- uart_rxd_in_ptr
 - UART.C, 51
- uart_rxd_out_ptr
 - UART.C, 51
- uart_txd_buf_cnt
 - UART.C, 51
- uart_txd_buffer
 - UART.C, 51
- uart_txd_in_ptr
 - UART.C, 52
- uart_txd_out_ptr
 - UART.C, 52
- UDP_MODE
 - ICHIP.H, 37
- UDP_TIMEOUT
 - ICHIP.H, 37
- Urgent_Data_Pointer_H
 - ICHIP.H, 37
- Urgent_Data_Pointer_L
 - ICHIP.H, 37
- W_Putc
 - ICHIP.C, 21
 - ICHIP.H, 38
- WriteSeiko
 - ICHIP.C, 22
- ICHIP.H, 38
- WRITEXHI
 - ICHIP.C, 20
- WRITEXLO
 - ICHIP.C, 20