Introduction

This document describes the data link communication protocol and the services provided to the application layer.

This communication protocol is designed from the ST7580 power line modem devices.

This firmware is available by software license agreement only:

- [http://www.st.com/software_license_agreement_liberty_v2](http://www.st.com/software_license_agreement_liberty_v2) addresses conditions to deliver the firmware in source code form
- [http://www.st.com/software_license_agreement_image_v2](http://www.st.com/software_license_agreement_image_v2) addresses conditions to deliver the firmware in binary code form (i.e. .lib, .a, …)

Please, ask your local ST office for further information.
# Contents

1. Overview ................................................................................. 4
2. Firmware architecture .......................................................... 5
3. Framing .................................................................................. 7
4. Service description ................................................................. 9
   4.1 System initialization services .............................................. 9
      4.1.1 Data link initialization .............................................. 9
      4.1.2 PLM modem channel frequency configuration .............. 10
   4.2 System configuration services ............................................. 10
      4.2.1 PLM modem configuration ........................................ 10
      4.2.2 Set MAC address .................................................. 11
      4.2.3 Get MAC address .................................................. 12
   4.3 Data link services .............................................................. 12
      4.3.1 Allow join request .................................................. 12
      4.3.2 Join Indication ...................................................... 12
      4.3.3 Data request ....................................................... 13
      4.3.4 Data indication ..................................................... 13
   4.4 Data link security services ................................................ 14
      4.4.1 Security data request .............................................. 14
      4.4.2 Security data indication ........................................... 14
   4.5 MIB management services ................................................ 15
      4.5.1 MIB reading ........................................................ 15
      4.5.2 MIB writing ......................................................... 15
      4.5.3 MIB erasing ......................................................... 15
      4.5.4 MIB parameters .................................................... 16

Appendix A Join procedure ......................................................... 17

Appendix B The STM32Fx family modem configuration parameter description ....................................................... 18
Appendix C  Example to configure the communication protocol .......................... 19
   C.1  Configure the STM32Fx family serial layer ............................................... 19
   C.2  Configure and run the data link layer ...................................................... 20

Appendix D  Security service use and configuration ......................................... 21
   D.1  How to use the security service .............................................................. 21
   D.2  Example of MIB modem configuration to use the security service ............ 21

5  References ........................................................................................................... 21

6  Revision history .................................................................................................. 22
1 Overview

The ST7580 data link communication protocol provides the services to build and manage a “master multi-slave” power line network.

Two different modes are described in this document:

- **Master multi-slave mode:**
  - In a master-multi-slave network, one node, and only one node, must be configured as a “master” while the other nodes must be configured as “slaves”. Each slave needs a join procedure with the master before starting communication; Appendix A describes the join procedure.

- **Simple node mode:**
  - In simple node mode all the nodes must be configured as “master”; in this case the nodes do not need a join procedure and they can all communicate with any other node.

The protocol has been specified for the STM32Fx family microcontroller. It utilizes a serial communication (UART) with the ST7580 power line device.
2 Firmware architecture

Figure 1 shows the firmware architecture:

![Firmware architecture diagram]

The physical layer of the ST7580 provides a basic MAC service through a serial communication interface.

- **BSP layer:**
  - The “BSP” layer manages the communication interfaces from the physical layer (ST7580) and the data link layer. Moreover, this manages the anti-collision repeating message.

- **Data link layer:**
  - The “Data link” layer manages the framing sent and received, and the power line modem configuration.

- **Service layer:**
  - The “Service” layer provides the data link services to the application layer; it allows messages and commands to be sent and received.

- **Application layer:**
  - The “Application” layer can be customized by the user; in this section the user must design the application to create, manage and maintain the network.
Figure 2 shows the firmware components and their files:

### Figure 2. Firmware components and files

- **Service**
  - dl_services.c
  - dl_services.h

- **Data link**
  - dl_mgt20.c
  - dl_mgt20.h

- **BSP**
  - st7580_stm32_bsp.c
  - st7580_stm32_bsp.h
3 Framing

The communication modulation is an X-PSK, and the physical frame is the typical PSK frame, shown in Figure 3.

**Figure 3. PSK frame**

![PSK frame diagram](image)

where:
- Preamble length: 32 bits
- Unique word: 16 bits
- Mode: 8 bits
- P_SDU length: 8 bits
- P_SDU: max. 255 bytes

The P_SDU contains the data link frame which has the structure shown in Figure 4.

**Figure 4. Data link frame**

![Data link frame diagram](image)

where:
- “Type” defines the type of frame, which can be:
  - AllowJoin
  - JoinReq
  - JoinAck
– Leave (not used in this protocol release)
– LeaveAck (not used in this protocol release)
– Data
  • “Source UID” and “Destination UID” are the sender and receiver MAC address
  • “Source level” (not used in this protocol release)
  • “MAC payload” is the data
  • “CRC” depends on the ST7580, for more details see the ST7580 datasheet.
4 Service description

4.1 System initialization services

These services provide functions to initialize the data link layer, to configure and manage the modulation frequency of the power line modem.

4.1.1 Data link initialization

void DL_Init(u8 DLType)

Description:
This function initializes the data link layer, with default parameters; the default parameters are:

- Frequency: 86000 Hz, 72000 Hz
- Rx mode: high channel only
- Rx H/L channel modulation: all PSK modulation
- Current control: enabled
- Gain: 21
- Zero crossing delay: 0
- PSK preamble length: 32 bits
- FSK baud rate: 2400 bps
- FSK deviation: 1
- FSK preamble length: 32 bit
- FSK unique word length: 16 bit
- FSK unique word: 0x9B58

Input parameters:

- DLType: this parameter defines the data link type, master, slave or simple node; the value accepted is: “DL_MASTER”, “DL_SLAVE” or “DL_SIMPLE_NODE”.

Output parameters:
None.

Return value:
None.

Note:
This function must be called during the system startup.
4.1.2 PLM modem channel frequency configuration

```c
void DL_FrequenciesConfig(u32 HChannel, u32 LChannel, FREQ_TYPE FreqType)
```

**Description:**
This service configures the PLM ST7580 modem frequency channel.

The use of this service has a lasting effect, in fact, it changes the default configuration parameters of the frequency channel. If this service is used, the value of the frequency channel is maintained for all successive system activations.

**Input parameters:**
- `HChannel`: frequency channel to configure the high channel, for the accepted value see the ST7580 datasheet.
- `LChannel`: frequency channel to configure the low channel, for the accepted value see the ST7580 datasheet.
- `FreqType`: type of change, valid values “PERMANENT”, “NOT_PERMANENT”.

**Output parameters:**
None.

**Return value:**
None.

**Note:**
This service can be used to change the default frequency channel in the configuration parameters.

If the parameter “FreqType” is “PERMANENT”, the new modem frequency value is registered in the RAM and in the Flash memory and this is the new default value.

If the parameter “FreqType” is “NOT_PERMANENT”, the new modem frequency value is registered in the RAM memory only; at reset this modem frequency value is lost and the system reloads the old default value.

4.2 System configuration services

These services provide the functions to configure the power line modem and manage the MAC address of the system.

4.2.1 PLM modem configuration

```c
void DL_ModemConfig(MOD_ConfigTypedef* ModConfParam, CONF_TYPE ConfType)
```

**Description:**
This service is used to configure the modem with a different parameter respect to the default values. This service does not have a lasting effect; the effect of this service is valid while a reset occurs or a new startup is performed.
Input parameters:

- ModConfParam: pointer to “MOD_ConfigTypedef” struct, this includes all modem configuration parameters, this struct, all parameters and the valid parameter values are described in the Appendix B.
- ConfType: type of change, valid values “PERMANENT”, “NOT_PERMANENT”.

Output parameters:

None.

Return value:

None.

Note:

This service doesn’t substitute DI_Init. In fact, the DI_Init must be performed at the system initialization layer at all times.

If the parameter “ConfType” is “PERMANENT”, the new modem configuration value is registered in the RAM and in the Flash memory and this is the new default value.

If the parameter “Confype” is “NOT_PERMANENT”, the new modem configuration value is registered only in the RAM memory; at reset this modem configuration value is lost and the system reloads the old default value.

4.2.2 Set MAC address

void DL_AddressSet(u8* NewAddress, CH_ADDR_TYPE AddType)

Description:

This service is used to set a new MAC address. At the first startup, the system gets a UID MAC address, this is the UID of the microcontroller. This service can change this address permanently or temporarily. If the service is used “temporarily” at the reset, the UID MAC address is reloaded; if the service is used as “permanent”, the MAC address is changed permanently.

Input parameters:

- NewAddress: pointer to new address, (6-byte array)
- AddType: type of change, valid value “PERMANENT”, “NOT_PERMANENT” “AddType”.

Output parameters:

None.

Return value:

None.

Note: If the parameter “AddType” is “PERMANENT”, the new address is registered in the RAM and in the Flash memory, and this is the new default value. If the parameter “AddType” is “NOT_PERMANENT”, the new address is registered in the RAM memory only; at reset this address is lost and the system reloads the old default address.
4.2.3 Get MAC address

void DL_AddressGet(u8* Address)

Description:
This service is used to read the current MAC address.

Input parameters:
- Address: pointer to the address (6-byte array).

Output parameters:
None.

Return value:
None.

4.3 Data link services

These services provide the functions to send, to receive messages and to manage the network joining.

4.3.1 Allow join request

s8 DL_AllowJoinRequest(u8* DAddress, u8* Data, u8 Dlen)

Description:
This service is used by the application layer to perform an “allow join request” to another node.

Input parameters:
- DAddress: pointer to destination node MAC address
- Data: pointer to data collection to send to the node (free application data)
- Dlen: length of data, number of bytes.

Output parameters:
None.

Return value:
Data link error code value.

4.3.2 Join Indication

u8* DL_JoinIndication(void)

Description:
This service reports to the application layer if a join indication occurs.
**Service description**

**Input parameters:**
None.

**Output parameters:**
None.

**Return value:**
Join indication identifier.

### 4.3.3 Data request

```
s8 DL_DataRequest(u8* DAddress, u8 Dlen, u8* Data)
```

**Description:**
This service is used by the application layer to send a data collection to another node.

**Input parameters:**
- DAddress: pointer to destination node MAC address
- Data: pointer to data collection to send to the node
- Dlen: length of data, number of bytes.

**Output parameters:**
None.

**Return value:**
Data link error code value.

### 4.3.4 Data indication

```
u8* DL_DataIndication(u8* SAddress, u8* DLen, u8* DiParam)
```

**Description:**
This service reports to the application layer if a data indication occurs.

**Input parameters:**
None.

**Output parameters:**
- SAddress: pointer to source node MAC address
- Dlen: length of data payload, number of bytes
- DiParam: data indication parameters, 4 bytes (see the UM0932 user manual).

**Return value:**
Pointer to data collection (payload).
4.4  **Data link security services**

These services provide the functions to send and receive messages with security features. The security data management is performed by using the ST7580 on-chip 128-bit AES encryption HW block.

For further information about the use of security service, see *Appendix D*.

4.4.1  **Security data request**

```c
s8 DL_SS_DataRequest(u8* DAddress, u8 Dlen, u8* Data)
```

**Description:**

This service is used by application layer to send a data collection in security mode to another node.

**Input parameters:**

- DAddress: pointer to destination node MAC address
- Data: pointer to data collection to send to the node
- Dlen: length of data, number of bytes.

**Output parameters:**

None.

**Return value:**

Data link error code value.

4.4.2  **Security data indication**

```c
u8* DL_SS_DataIndication(u8* SAddress, u8* DLen, u8* DiParam)
```

**Description:**

This service reports to application layer if a security data indication occurs.

**Input parameters:**

None.

**Output parameters:**

- SAddress: pointer to source node MAC address
- Dlen: length of data payload, number of bytes
- DiParam: data indication parameters, 4 bytes (see the UM0932 user manual).

**Return value:**

Pointer to data collection (payload).
### 4.5 MIB management services

These services provide functions to read, write or erase the MIB (management information base) of the power line modem, see Table 1 for details.

For more information about the MIB services and their management, please see the ST7580 datasheet.

#### 4.5.1 MIB reading

```c
s8 DL_MIB_ReadRequest (MIB_INDEX_OBJ_T MIBIndex, u8* Data)
```

**Description:**

This service is used by the application layer to read the MIB object information.

**Input parameters:**

- MIBIndex: index of MIB object, (see Table 1).

**Output parameters:**

- Data: pointer to MIB data object (the length depends on the object, see the ST7580 datasheet).

**Return value:**

Status or error value, (see Table 2).

#### 4.5.2 MIB writing

```c
s8 DL_MIB_WriteRequest (MIB_INDEX_OBJ_T MIBIndex, u8* Data)
```

**Description:**

This service is used by the application layer to write new data to the MIB object.

It is possible to use this service only for the MIB object with the following index: 0x00, 0x01, 0x02, 0x09.

**Input parameters:**

- MIBIndex: index of MIB object (see Table 1)
- Data: pointer to MIB data object (the length depends on the object, see the ST7580 datasheet).

**Output parameters:**

None.

**Return value:**

Status or error value, (see Table 2).

#### 4.5.3 MIB erasing

```c
s8 DL_MIB_Erase (MIB_INDEX_OBJ_T MIBIndex)
```
Description:
This service is used by the application layer to erase the MIB object information. It is possible to use this service only for the MIB object with the following index: 0x06, 0x07, 0x08.

Input parameters:
- MIBindex: index of MIB object, (see Table 1).

Output parameters:
None.

Return value:
Status or error value, (see Table 2).

4.5.4 MIB parameters

<table>
<thead>
<tr>
<th>Index</th>
<th>Name</th>
<th>Length (byte)</th>
<th>R/W/E</th>
</tr>
</thead>
<tbody>
<tr>
<td>00h</td>
<td>Modem configuration</td>
<td>1</td>
<td>R/W</td>
</tr>
<tr>
<td>01h</td>
<td>PHY configuration</td>
<td>14</td>
<td>R/W</td>
</tr>
<tr>
<td>02h</td>
<td>SS key</td>
<td>16</td>
<td>R/W</td>
</tr>
<tr>
<td>03h</td>
<td>Reserved</td>
<td>1</td>
<td>R</td>
</tr>
<tr>
<td>04h</td>
<td>Last data indication</td>
<td>4</td>
<td>R</td>
</tr>
<tr>
<td>05h</td>
<td>Last TX confirm</td>
<td>5</td>
<td>R</td>
</tr>
<tr>
<td>06h</td>
<td>PHY_Data</td>
<td>10</td>
<td>R/E</td>
</tr>
<tr>
<td>07h</td>
<td>DL_Data</td>
<td>12</td>
<td>R/E</td>
</tr>
<tr>
<td>08h</td>
<td>SS_Data</td>
<td>10</td>
<td>R/E</td>
</tr>
<tr>
<td>09h</td>
<td>Host interface timeout</td>
<td>3</td>
<td>R/W</td>
</tr>
<tr>
<td>0Ah</td>
<td>Firmware version</td>
<td>4</td>
<td>R</td>
</tr>
</tbody>
</table>

Table 2. Service return values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DL_BUSY</td>
<td>Data link is busy with another process, the service can't be performed</td>
</tr>
<tr>
<td>DL_WAIT</td>
<td>Wait state, the service is in execution phase</td>
</tr>
<tr>
<td>DL_TIMEOUT</td>
<td>This is an error timeout, the service failed due to timeout</td>
</tr>
<tr>
<td>DL_ERROR</td>
<td>Generic error. Service failed</td>
</tr>
<tr>
<td>DL_OK</td>
<td>Service processed correctly</td>
</tr>
</tbody>
</table>
Appendix A  Join procedure

Figure 5 shows the join procedure diagram.

Figure 5. Join procedure diagram

The actions performed by the master are:
- Sends the allow join
- Waits for the join acknowledge
- When the join request arrives, it sends a join ACK and marks the module as “Joined”.

At the same time the actions performed by the slave are:
- Waits for the allow join request
- When an allow join request arrives, sends a join request
- Waits for the JACK
- When the JACK arrives, the module is able to receive a data.

The message sequences are described in Figure 6:

Figure 6. Message sequences

The join procedure is mandatory for nodes configured as slave; if the joined procedure doesn’t perform correctly, a slave node is not able to communicate with the master.
**Appendix B**  
The STM32Fx family modem configuration parameter description

The following is a description of the modem configuration data struct:

```c
typedef struct{
    u32 m_Ser_H_Freq;      /* Frequency H Channel */
    u32 m_Ser_L_Freq;      /* Frequency L Channel */
    u8 m_Rx_Control;       /* RX Control Parameters */
    u8 m_SerGain;          /* Gain*/
    u16 m_SerZeroCrossing; /*Zero Crossing*/
    SV_PRE_LEN m_SerPSKPreamLen; /*PSK Preamble Length*/
    u8 m_FSK_Config_Byte   /* FSK Configuration Byte */
}MOD_ConfigTypedef;
```

The valid value of the parameters are:

- `m_Ser_H_Freq;` CENELEC A-B or C, compliant frequency integer value.
- `m_SerL_Freq;` CENELEC A-B or C, compliant frequency integer value.
- `m_Rx_Control;` Rx Control byte value (see datasheet).
- `m_SerGain;` integer value from 0 to 31.
- `m_SerZeroCrossing;` Zero Crossing delay value.
- `m_SerPSKPreamLen;` BIT_16, BIT_24, BIT_32,BIT_40.
- `m_FSK_Config_Byte;` FSK Configuration Byte, (see datasheet).
Appendix C  Example to configure the communication protocol

C.1 Configure the STM32Fx family serial layer

Follow the STM32Fx family USART configuration used inside the serial layer to communicate with a PLM device:

Usart ' USART3

USART_RX_GPIO ' GPIO_Pin_11_PORT_B
USART_TX_GPIO ' GPIO_Pin_10_PORT_B

The part of the code to modify is:

/*-----------------------------------------------------------*/
/* Serial Define */
#define USART       USART3
#define USART_PORT  GPIOB
#define USART_TX    GPIO_Pin_10
#define USART_RX    GPIO_Pin_11
#define USART_TR    GPIO_Pin_9
#define USART_IRQ   USART3_IRQn
#define RCC_APBPeriph_USART RCC_APB1Periph_USART3
#define RCC_APBPeriph_USART_ClockCmd(Periph_USART,value)
RCC_APB1PeriphClockCmd(Periph_USART,value)
#define RCC_APBPeriph_GPIO RCC_APB2Periph_GPIOB
#define RCC_APBPeriph_GPIO_ClockCmd(Periph_GPIO,value)
RCC_APB2PeriphClockCmd(Periph_GPIO,value)
/*-----------------------------------------------------------*/

The names "RCC_APB2Periph_GPIOB", "GPIO_Pin_x", "USART3" ecc.

The names are defined in the STM32F10x_StdPeriph_Driver.

Furthermore, it is necessary to call the function “vUARTInterruptHandler()” defined in serial.c in the appropriate InterruptServiceRoutine. The following example shows the ISR of USART3 in the Interrupt file of the STM32Fx family library:

Example:

void PPP_IRQHandler(void)
{
}
void UART4_IRQHandler(void)
{
}
void USART3_IRQHandler(void)
{
    vUARTInterruptHandler();
}
void USART1_IRQHandler(void)
C.2 Configure and run the data link layer

To configure the data link layer, it is necessary to call the function “DL_Init(DLType)” provided in the services.h, only once in the main.c.

To run the data link it is necessary to call, in a loop, the function “DL_FSM()” provided in the dl_mgt20.h.

The following example shows these:

Example:
```c
void main(void)
{
    DL_Init(DL_MASTER);
    while(1)
    {
        DL_FSM();
    }
}
```
Appendix D  Security service use and configuration

D.1  How to use the security service

The MIB modem configuration (MIB table index 00h) must be configured in SS_Layer (value: 2) to use the security service.

If the ST7580 is not configured in SS_Layer, the ST7580 PLM can send a security data but it cannot receive any security data.

D.2  Example of MIB modem configuration to use the security service

In the system configuration phase, the following instruction has to be used:

```c
#define MOD_CONF_SS_LAYER    0x2
DL_MIB_WriteRequest (MIB_M_CONFIG, MOD_CONF_SS_LAYER)
```

5  References

1. ST7580 datasheet.
2. UM0932 user manual.
6 Revision history

Table 3. Document revision history

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>04-Oct-2012</td>
<td>1</td>
<td>Initial release.</td>
</tr>
<tr>
<td>19-Oct-2012</td>
<td>2</td>
<td>Added new details on the cover page about the firmware availability.</td>
</tr>
<tr>
<td>29-Mar-2013</td>
<td>3</td>
<td>Added Section 4.4: Data link security services and Appendix D. Minor text changes.</td>
</tr>
</tbody>
</table>