HÖFT &
WESSEL

Firmware Manual

HW 86010, HW 86020

Firmware Release 2.2.2

Document Version 2.00

Document information

Product:	HW 86010/20, firmware release 2.2.2
Type of document:	Firmware Manual
Version:	2.00
Date of issue:	28.02.2005
Authors:	Dr. Andreas Müller, Höft & Wessel AG Thomas Gerike, Höft & Wessel AG Ralf Schmidt, Höft & Wessel AG
Change history	
1.00	First pre-release version
1.01	New config commands and features added
1.02, 18.01.02	First official release
1.03, 05.03.03	New config commands added
1.04, 19.11.03	New config commands added
1.05, 07.09.04	Config commands restructured
2.00, 28.02.05	CLDPS added, Errors corrected, References added, API of protocol data mode added, Voice mode added

Contents

1.0 Preface 8
1.1 About this document 8
1.2 Contact Höft & Wessel AG 8
1.3 References 8

2.0 Product overview 9
2.1 General description 9
2.2 Summary of features 11
2.3 Typical applications 11
2.4 Principles of operation 12
2.4.1 DECT network entities 12
2.4.2 Connections 13

3.0 Firmware description 14
3.1 Overview 14
3.1.1 Operation modes 14
3.1.2 Mode selection 15
3.1.2.1 Selection by reset sequence 15
3.1.2.2 Selection by software escape commands 15
3.2 System security 16
3.2.1 DECT identities 17
3.2.1.1 FT related identities 17
3.2.1.2 PT related identities 19
3.2.1.3 Subscription identities 20
3.2.2 EasySubs 21
3.2.3 On-Air subscription of Portable Terminals 22
3.2.4 Offline subscription of Portable Terminals 22
3.3 Configuration mode 23
3.3.1 Entering the configuration mode 23
3.3.2 Configuration protocol 23
3.3.3 Leaving the configuration mode 23
3.3.4 Configuration command overview 24
3.3.4.1 Commands 24
3.3.4.2 Return codes 27
3.3.4.3 Argument formats 28
3.3.5 Configuration commands reference 29
3.3.5.1 Hardware commands 29
3.3.5.1.1 All hardware parameters: GHALL 29
3.3.5.1.2 Module type: GHTY 29
3.3.5.1.3 Relative rssi value: GHRSSI 30
3.3.5.1.4 Receive quality: GHQUAL 30
3.3.5.1.5 Flash memory type: GHFL 31
3.3.5.1.6 Antenna: SPANT / GPANT 31
3.3.5.2 Software-ID commands 32
3.3.5.2.1 Software versioning parameters: GSALL 32
3.3.5.2.2 Firmware build: GSNR 32
3.3.5.2.3 Firmware version: GSVER 33
3.3.5.3 Module commands 33

3.3.5.3.1 Module parameters: GMALL 33
3.3.5.3.2 Module frequency: GMF 34
3.3.5.3.3 Module quad tune: GMQ 34
3.3.5.3.4 Module modulation: GMM 35
3.3.5.4 Mode commands 35
3.3.5.4.1 Protocol mode flag: SPPR / GPPR 35
3.3.5.4.2 Value of CTR6 test mode: SPCTR / GPCTR 36
3.3.5.4.3 Value of CLDPS flag: SPCLDPS / GPCLDPS 37
3.3.5.4.4 Multipoint flag: SPMP / GPMP 38
3.3.5.5 Info commands 39
3.3.5.5.1 Serial number parameters: GNALL 39
3.3.5.5.2 European manufacturer: GNEMC 39
3.3.5.5.3 MAC address: GNETH 40
3.3.5.5.4 DECT serial number: GNDNR 40
3.3.5.5.5 Production serial number: GNSER 40
3.3.5.5.6 Unit number: GNUNR 41
3.3.5.6 Identity commands 42
3.3.5.6.1 Air subscription accept: SIAIR / GIAIR 42
3.3.5.6.2 Air subscription identified by PARK: SISUA / SISUB / SISUD / GISUB / DISUB 43
3.3.5.6.3 Subscription key: GISK 47
3.3.5.6.4 Identity PIN: SIPIN 47
3.3.5.6.5 Subscription master key: SISMK 48
3.3.5.6.6 PARK of FT: GIPARK 48
3.3.5.6.7 Access rights identity: SIARI / GIARI / DIARI 49
3.3.5.7 Voice commands 50
3.3.5.7.1 Voice microphone parameters: SPVMIC/ GPVMIC 50
3.3.5.7.2 Voice mode flag: SPVOICE / GPVOICE 51
3.3.5.7.3 Voice speaker: SPVSPE / GPVSPE 52
3.3.5.7.4 Voice sidetone: SPVST / GPVST 53
3.3.5.8 Serial and IO commands 54
3.3.5.8.1 Baud rate SPBD / GPBD / IPBD 54
3.3.5.8.2 Serial communication: SPCOM / GPCOM 55
3.3.5.8.3 Parameter call control: SPCC / GPCC 56
3.3.5.8.4 Enhanced call control: SPECC / GPECC 57
3.3.5.8.5 User interface: SPUI / GPUI 58
3.3.5.9 Other configuration commands 59
3.3.5.9.1 Configurable parameters: GPALL 59
3.3.5.9.2 Type of DECT termination: SPTM / GPTM 60
3.3.5.9.3 Dial string internal: SPDSI / GPDSI / DPDSI 61
3.3.5.9.4 Dial string default: SPDSD / GPDSD / DPDSD 62
3.3.5.9.5 Customer string: SPCUST / GPCUST / DPCUST 63
3.3.5.9.6 Location flag: SPLOC / GPLOC 64
3.3.5.9.7 Parameter sync. windows: SPSYWD / GPSYWD 65
3.3.5.9.8 DPSCFG command 66
3.3.5.9.9 Retry value: SPRETRY / GPRETRY / DPRETRY 67
3.3.5.9.10 Timeout v.: SPTIMEOUT / GPTIMEOUT / DPTIMEOUT 69
3.3.5.10 General commands 71
3.3.5.10.1 All data: GALL 71
3.3.5.10.2 Firmware diagnostics: CRC 72

3.3.5.10.3 Result code Ok: GOK 72
3.3.5.10.4 Exit configuration mode: EXIT 73
3.4 Transparent data mode 74
3.4.1 Usage of RS-232 interface 74
3.4.1.1 Connection of the interface 74
3.4.1.2 Interface parameters 75
3.4.2 Flow Control 75
3.4.3 Interworking of modem lead signals 76
3.4.4 Call control 76
3.4.4.1 Outgoing call, PT interface 77
3.4.4.2 Outgoing call, FT interface 77
3.4.4.3 Incoming call, PT interface 78
3.4.4.4 Incoming call, FT interface 78
3.4.4.5 Call release, PT interface 79
3.4.4.6 Call release, FT interface 79
3.4.5 Data transmission 80
3.4.5.1 Alignment with call control 80
3.4.5.2 Usage of modem lead signals 80
3.4.5.3 Escaping to configuration mode 80
3.4.6 Example: Transparent multipoint mode 81
3.5 Protocol data mode 81
3.5.1 General description 82
3.5.2 Usage of RS-232 interface 84
3.5.2.1 Connection of the interface 84
3.5.2.2 Interface parameters 84
3.5.3 HDLC frame structure 85
3.5.3.1 Flag field (FLAG) 85
3.5.3.2 Address field (ADDR) 85
3.5.3.3 Control field (CTRL) 86
3.5.3.4 Data field (DATA) 86
3.5.3.5 Frame check sequence (FCS) 87
3.5.4 HDLC procedures 88
3.5.4.1 Multiplexing of LAP channels 88
3.5.4.2 Transparency 89
3.5.5 LAP protocol overview 90
3.5.6 LAP information elements 91
3.5.6.1 Information frames 92
3.5.6.2 Supervisory frames RR, RNR and REJ 92
3.5.6.3 Supervisory frames SABM and UA 92
3.5.6.4 Information elements in the ADDR field 93
3.5.7 LAP procedures 94
3.5.7.1 States 94
3.5.7.2 Conditions 94
3.5.7.3 Timers 95
3.5.7.4 Sequence variables 95
3.5.7.5 Sender procedures 96
3.5.7.6 Receiver Procedures 98
3.5.7.7 Establishment 99
3.5.7.8 Termination 99
3.5.7.9 Re-establishment 99

3.5.8 SDL representation of LAP 100
3.5.9 Call control information elements 109
3.5.9.1 General description 109
3.5.9.2 ConnectInd command 110
3.5.9.3 DisconnectInd command 111
3.5.9.4 ConnectReq command 112
3.5.9.5 DisconnectReq command 112
3.5.9.6 LocationInd command 113
3.5.9.7 LocationRes command 114
3.5.10 Call control procedures 115
3.5.10.1 Incoming call 115
3.5.10.2 Outgoing call 115
3.5.10.3 Call release, host initiated 116
3.5.10.4 Call release, PT initiated 116
3.5.11 API of the dectprot.dll 117
3.5.11.1 DECT_CALLBACK_FUNC_T 117
3.5.11.2 DectInit 118
3.5.11.3 DectDestroy 118
3.5.11.4 DectRegisterCallback 118
3.5.11.5 DectOpen 119
3.5.11.6 DectClose 119
3.5.11.7 DectRead 119
3.5.11.8 DectWrite 120
3.5.11.9 DectConnectReq 120
3.5.11.10 DectDisconnectReq 121
3.5.11.11 DectGetConnStatus 121
3.5.11.12 DectGetLineStatus 121
3.5.11.13 DectGetIpui 122
3.5.11.14 DectGetBytesAvail 122
3.5.11.15 DectGetTxFree 122
3.5.11.16 DectGetTxPending 122
3.5.11.17 DectLapStateCfm 123
3.5.11.18 DectLocationRes 123
3.5.11.19 DectSwitchRoaming 123
3.5.11.20 DectSwitchLocation 124
3.5.11.21 DectLapStateGetLen 124
3.5.11.22 DectLapStateGetIpui 124
3.5.11.23 DectLapStateGetCallNr 125
3.5.11.24 DectBuildIpuiTypeN 125
3.5.11.25 DectReadTo 126
3.5.11.26 DectWriteTo 126
3.6 Voice mode 127
3.6.1 Block diagram 127
3.6.2 Advises on voice commands 128
3.6.2.1 Speaker output configuration 128
3.6.2.2 Microphone input configuration 129
3.6.2.3 Sidetone configuration 130

4.0 Abbreviations 131

5.0 Index 133

1.0 Preface

1.1 About this document

HW 86010 and HW 86020 are delivered together with version 2.2.2 of Höft & Wessel DECT firmware. The information contained in this firmware manual relates to firmware version 2.2.2.

For hardware-related information please see the Integration Manual of the respective product.

1.2 Contact Höft & Wessel AG

For immediate assistance please address yourself to the Höft & Wessel service line:

Telephone: +49-1803-232829
Telefax: +49-511-6102-421
Email: info@hoeft-wessel.de

If you have general questions concerning Höft & Wessel communication products you may directly contact the communications department:

Telephone: +49-511-6102-226
Telefax: +49-511-6102-421
Email: tol@hoeft-wessel.de

Latest revisions of all publicly available documentation and firmware downloads are available from our web-site www.hoeft-wessel.de

Höft & Wessel AG
Rotenburger Strasse 20
D-30659 Hannover
GERMANY

1.3 References

Source	Description
Höft & Wessel AG	• User Manual HW 8612, HW 8612 US, HW 8614, HW 8614 US, HW 8615, HW 8617, HW 8617 US, HW 8621, HW 8622, HW 8622 US, HW 8660, HW 8660 US, HW 86916/26 • Data Unwired System Manual

2.0 Product overview

The DECT transceiver module HW 86010 and the Frequency Hopping Spread Spectrum (FHSS) transceiver module HW 86020 are highly versatile and powerful engines for popular and advanced DECT / FHSS applications. They provide both RF and baseband signal processing as well as a complete protocol stack.

2.1 General description

The protocol stack has been implemented as firmware running on the micro controller of the HW 86010/20. It comprises the DECT protocol layers MAC (EN 300 175-3), DLC (EN 300 175-4) and NWK (EN 300 175-5). Data service is provided according to the DSP C.1/C.2 profile based on LU3 connection. It offers payload data rates of four times 26 kBit/s in point-to-multipoint applications.

A new transmission protocol – CLDPS (Connection-Less DECT Packet System) – has been implemented. It offers connection-less, packed based data transmission on DECT with payload data rates of 500 kBit/s (per base station). A base station allocates 12 DECT channels (time / frequency multiplexing) and uses a dedicated slot format. This capacity can dynamically be shared between the subscribed portables according to the actual demand. CLDPS allows 64 simultaneously connected portables and therefore is capable to support even large wireless networks.

Data-Unwired systems usually support both connection-based DECT and packet-based CLDPS, which can be configured by software.

Comparison of both protocols:

Chracteristic	Connection based	Packet based (CLDPS)
networking capability	no[1]	yes, 64 active subscribers per radio cell (= base station)
data rate	2x26 kBit/s for up / downlink, synchronous	500 kBit/s per radio cell, asynchronous
symmetry up/downlink	symmetrical	asymmetrical, dynamical
occupied DECT channels	1 duplex	12 duplex per radio cell
real-time capabilities	data are transferred in fixed 10 ms time frame	allocation of timeslots by base station not collision-based 9 traffic slots per 10ms time frame undetermined behaviour with increasing number of subsribers and amount of data
voice transmission	yes, up to 4 simultaneous links per base station	no[2]

The FHSS protocol stack of the HW 86020 further includes the MAC layer procedures related to frequency hopping.

Moreover the firmware includes full interworking with the RS-232 interface.

Note: Earlier firmware versions supported a high-speed point-to-point mode. Due to the development of CLDPS Höft & Wessel devices will no longer support this mode.

---

[1] presents 1:4 protocol mode further supported but not recommend for new projects.
[2] packet based voice transmission possible

2.2 Summary of features

Feature	Short description
Air interface	Compliant with DECT (EN 300 175) (HW 86010) Compliant with FCC part 15 (HW 86020)
Protocols	C-Plane according GAP (EN 300 444)
Data transmission	Connection Orientated: According DSP C.2 (EN 300 651) Connection-Less: CLDPS
Point-to-multipoint	Connection Orientated: up to 4 simultaneous connections (4x26 kBit/s) (EN 300 651) Connection-Less: up to 64 simultaneous connections (500 kBit/s per radio cell) (CLDPS)
Small footprint	Size: 53 mm x 37 mm
Versatile interfaces	e.g. RS-232, PCM, I/O, I²C, voice, bus
Firmware upgradeable	Firmware can be downloaded
Voice	Voice transmission is possible
Easy configuration	Configuration mode for easy installations
Easy subscription	Through the new EasySubs technique

2.3 Typical applications

- Wireless internet access
- Radio networks
- Mobile data acquisition
- Automation and industrial control

2.4 Principles of operation

2.4.1 DECT network entities

HW 86010 employs radio transmission according to the international DECT standard. It is compliant with the air interface standard EN 300 175. HW 86020 uses a modified version of that standard which is compliant with FCC part 15. The following description applies to both systems.

The DECT standard defines two communication entities: The fixed termination (FT), commonly seen as base station, and the portable termination (PT), usually a handset. Throughout this manual the terms “fixed” and “portable” are used in the DECT sense. This does not preclude that a FT may change its location or a PT may be stationary mounted.

The HW 86010/20 can be configured either as PT or as FT. For the most simple case, a point-to-point connection between two modules, one side must be configured as PT and the other side as FT

The general architecture of any DECT system comprises one FT and a variable number of PTs. This is called a point-to-multipoint network. The number of PTs in a network is not limited by the DECT standard but only by implementation constraints.

Larger DECT networks often include multiple “base stations”. Strictly speaking, the DECT network still has a single FT but this is distributed on multiple cells. Many people get confused about that concept, because they associate “base station” and FT. Within DECT terminology the term “base station” is not used at all, but this entity is called a “radio fixed part” (RFP). So in any DECT system there is one FT which comprises one or multiple RFPs.

All entities are identified by DECT-internal “addresses” (for a more detailed discussion on DECT identifiers see section 3.2.1). When installing a DECT system, every PT must learn the identity of the FT and the FT must learn the identities of each PT. This procedure is called subscription. Subscription defines which PTs belong to a FT. All DECT security features (authentication and encryption) build on that mechanism. The subscription procedure for HW 86010/20 is described in sections 3.2.3 and 3.2.4.

2.4.2 Connections

A connection always involves a pair PT - FT. There are no direct connections between two PTs.

Call control works similar to a telephone system. This means there are the following phases during a communication:

1. A call is set up either by the calling party (can be PT or FT)
2. The call is accepted by the called party (normal case). However the system may be busy or the called party is not ready to answer the call (exceptional case).
3. The communication channel is used for payload data
4. The call is released by any party (normal case) or by the system (exceptional case)

HW 86010 provides efficient methods of call control. These are described in more details in sections 3.4.4 and 3.5.10.

Different types of connections are defined by the DECT standard. E.g. a data connection differs very much from a voice connection. Most available DECT devices only support voice connections. This explains, why it is usually not possible to send data from a HW 86010 to a consumer type of DECT “base station”.

Connection types supported by HW 86010/20 include data connections of type LU3 and voice connections of type LU1. Explanations of LU1, LU3 are given in the DECT DLC layer standard EN 300175-4.

HW 86010/20 supports advanced connection set-up including symmetric multi-bearer connections.

3.0 Firmware description

This section covers a description of Höft & Wessel DECT firmware release 2.2.2 which is the actual firmware of HW 86010/20.

At this time there are two types of firmware:

- single-bearer firmware handles up to 4 connections at a time in point-to-multipoint applications with payload data rates of up to 26 kBit/s per connection
- cldps firmware handles up to 64 connections at a time in point-to-multipoint applications with payload data rates of 250 kBit/s up to 518 kBit/s

It's not possible to transmit data from one firmware type to another.

3.1 Overview

All functions of the HW 86010/20 are enabled by suitable firmware. This includes the processing of the DECT communication protocols, the control of interfaces and other features.

3.1.1 Operation modes

The firmware may run in any of the following operation modes:

Operation mode	Purpose
Configuration mode	set-up module parameters
Data mode	radio data transmission using the RS-232 as interface.
Download mode	load the DECT module with new firmware

The data mode has the following sub-modes

Data sub-modes	Purpose
Transparent data mode	transparent data transmission over RS-232 interface, single connection endpoint
Protocol data mode	multiplexed data transmission over RS-232 interface, multiple connection endpoints

The data sub-mode can be configured by use of the **SPPR** configuration command (see section 3.3.4).

Each of the operation modes has a specific usage of the RS-232 interface. Please refer to the descriptions of the operation modes.

3.1.2 Mode selection

The operation mode can be selected either by an appropriate reset sequence or by software escape commands. The download mode can only be selected by a reset sequence.

3.1.2.1 Selection by reset sequence

The download mode is entered depending on the status of the BOOT0 and BOOT1 signals during reset. If not the download mode is entered, the following procedure applies.

In case of an external reset the firmware checks the values of RTSI and DTRI signals and selects the mode accordingly.

RTSI	DTRI	Reaction
HIGH	HIGH	Enter configuration mode
LOW	HIGH	Enter data mode
x	LOW	Reserved for future use, do not use

When the configuration mode is entered through an external reset it will be executed using a baud rate of 9.600 bd.

In case the data mode is selected, the data sub-mode is determined from the configuration information. The data sub-mode can be configured by use of the **SPPR** configuration command (see section 3.3.4).

3.1.2.2 Selection by software escape commands

A transition from configuration mode to data mode is performed without hardware reset by use of the **EXIT** configuration command (see section 3.3.4).

A transition from transparent data mode to configuration mode is performed without hardware reset by use of the +-+ escape sequence (see section 3.4.5.3). In this case the configuration mode will be executed using the baud rate configured for data mode.

3.2 System security

The DECT standard includes useful security functions that efficiently protect DECT systems from hostile break-in and espionage. For details on the security features please refer to standard EN 300 175-7.

The firmware implements security features in compliance with the GAP standard EN 300 444.

Before a PT is allowed to set-up connections to any FT it must be subscribed at that FT. During the subscription procedure PT and FT mutually exchange their identities.

In compliance with GAP the firmware supports on-air subscription of PTs, meaning that the subscription information is exchanged over the air interface. Through on-air subscription the HW 86010/20 can be subscribed to DECT equipment of other manufacturers.

Offline subscription is an alternative subscription procedure that does not require any information exchange over the air interface. This procedure is only supported by equipment of Höft & Wessel.

Both procedures lead to equivalent results and can be used alternatively.

On each connection set-up, the FT requests an authentication from the PT. This assures that only subscribed PTs connect to a FT.

User data is sent over the air in encrypted format. This provides effective protection from espionage.

3.2.1 DECT identities

The DECT standard defines identities for PTs and FTs that are used for mutual identification and authentication. Standard EN 300 175-6 contains a detailed description of these identities.

The following sub-sections contain a summary of the DECT identities and their usage.

3.2.1.1 FT related identities

A FT is identified by an **ARI** (access rights identity).

According to the DECT standard a FT may own multiple ARIs, which are called **PARI** (primary ARI), SARIs (secondary ARIs) and TARIs (tertiary ARIs). In accordance with the GAP service profile (EN 300 444) HW 86010/20 supports one ARI which is then the PARI. SARIs and TARIs are not supported.

The DECT standard allows different ARI classes. HW 86010/20 (as FT) uses most ARI class A, but ARI class B and C are also supported. However HW 86010/20 (as PT) is interoperable with FTs that use a different ARI class.

The ARI class A is a 36 bits wide, world-wide unique identifier. It is factory-burnt into the module during production and cannot be modified.

However the factory-burnt ARI can be overloaded by a user-defined ARI. The administration of multi-cell networks is simplified, if all RFPs carry the same ARI. Please see the configuration command SIARI.

The structure of the ARI class A is shown below.

0	0	0	EMC			FPN		
b35	b34	b33	b32	...	b17	b16	...	b0

The three leftmost bits are always zero. This identifies ARI class A.

The **EMC** (ETSI manufacturer code) is a 16-bit value that has been assigned by ETSI to a manufacturer. Höft & Wessel has assigned the EMC 322 (decimal).

The **FPN** (DECT fixed part number) is a 17-bit value that is unique in the context of an EMC. It is assigned by the manufacturer.

Höft & Wessel uses an internal code, the **DNR** (DECT serial number) to uniquely identify modules. The DNR is a 20-bit value. The FPN is derived from the DNR through integer division by eight:

FPN = DNR div 8

In a multi-cell environment the FT consists in multiple RFPs. In a single-cell environment there is only one RFP.

Each RFP is identified by a **RFPI** (radio fixed part identity). It consists in the PARI of the FT and the RPN (radio fixed part number). The **RPN** is used in multi-cell networks in order to distinguish between RFPs which have the same ARI.

RPN shall be 0 for standalone RFP (single-cell environment) and 1 to 7 for multi-cell systems.

For more complex installations with more than 7 RFPs please contact Hoeft & Wessel for ARI class B.

In order to identify allowed FTs any PT stores one **PARK** (portable access rights key). A PARK corresponds to a single ARI or to a group of ARIs that only differ in their least significant bits. The **PLI** (PARK length indicator) defines, how many bits of the ARI are relevant. The default is 36, i.e. all bits are relevant.

In multi-cell networks the PARK may be selected such that it covers the ARIs of all RFPs.

When a PARK is manually entered, it is coded according to the GAP standard. The following format is used.

- The PARK starts with two digits representing the PLI in decimal format.
- Then follow up to 12 digits representing <pli> bits of the ARI in octal format. If necessary the bit string is padded with zeros at the right side in order to achieve octal alignment.
- Finally a check digit is entered. The check digit is calculated as the sum of each digit multiplied by its position in the string modulo 11. The check digit lies between 0 and 10 and is represented either as the decimal digit, or as a "*" if equal to 10.

Sometimes it can be necessary to manually calculate a PARK from PLI, EMC and DNR this is illustrated in the following example:

EMC	decimal: 322	binary: 0000 0001 0100 0010
DNR	decimal: 524752	FPN = DNR div 8
FPN	decimal: 65594	binary: 1 0000 0000 0011 1010
PLI	decimal: 23	
ARI	binary: 000 0000 0001 0100 0010 1 0000 0000 0011 1010	
ARI(pli)	binary: 000 000 000 010 100 001 010 00	
	octal: 00024120 (last digit padded with zero)	
check	0*1+0*2+0*3+2*4+4*5+1*6+2*7+0*8= 48 modulo 11 = 4	
PARK	23000241204	

Note: The DNR is always a multiple of 8.

3.2.1.2 PT related identities

A PT is identified by an **IPEI** (international portable equipment identity). This is a 36 bits wide, world-wide unique identifier. It is factory-burnt into the module during production and cannot be modified.

The structure of the IPEI is shown below.

EMC	DNR
b35 ... b20	b19 ... b0

The IPEI is part of the default **IPUI** (international portable user identity) of type N, that is used for identification of a PT in a DECT network. The DECT standard allows other IPUI types and allows multiple IPUIs at a PT. HW 86010/20 (as PT) does not use IPUI types other than N. However HW 86010/20 (as FT) is interoperable with PTs that use a different IPUI type.

3.2.1.3 Subscription identities

During the subscription procedure defined in the DECT standard a **UAK** (user authentication key) is created. This key represents a pair (FT, PT) and must be known by PT and FT since it is used for the authentication procedure. When the FT requests an authentication from a PT it tests the correct UAK.

The UAK is not entered nor transmitted over the air interface but independently computed by FT and PT from public information which is encrypted using a secret **PIN** (personal identity number) code.

This PIN code is stored at the FT and must be entered at the PT as part of the subscription procedure.

The format of the PIN is 1 to 8 decimal digits.

Note: leading zeros in PIN codes are significant, e.g. PIN 007 is different from PIN 7.

The default PIN (factory setting) is: 0

The PIN code is entered at the FT by use of the SIPIN configuration command. System integrators are advised to use different PINs in different installations in order to provide a good level of security. The PIN must be entered at the first installation of a FT and can be modified by the system operator later.

The firmware supports on-air subscription according to GAP and a proprietary offline subscription procedure. In on-air subscription the public information is transmitted by the FT over the air interface, whereas in offline subscription it is read out from the FT as **SK** (subscription key).

The SK is an encrypted format of the UAK.

3.2.2 EasySubs

EasySubs is a powerful technique for handling of subscription information in the FT.

Conventional FT implementations include a table of all subscribed PTs with their UAKs. Since memory is limited, the FT may only support a very limited number of subscriptions.

The EasySubs technique avoids storage of UAKs in the FT but provides an efficient means for on-demand computation of UAKs from other information already available. Due to EasySubs, FTs of Höft & Wessel support an unlimited number of PT subscriptions.

EasySubs if fully compliant with the DECT standard. It is used for both, on-air and offline subscriptions. EasySubs is interoperable with GAP-compliant PTs of other manufacturers.

The security of the DECT system is fully preserved by EasySubs by introducing an additional key, the **SMK** (subscription master key). The SMK is stored in the FT in non-volatile memory. It is used during on-demand computation of UAKs.

Only a single SMK is needed, independent of the number of PTs to be subscribed.

The default SMK (factory setting) is: 00000000

The PT stores subscription information in the conventional way, i.e. EasySubs only affects the FT.

In multi-cell networks all RFPs must be programmed with the same values of PIN and SMK respectively.

A big advantage of EasySubs: Any PT must only be subscribed to a single RFP of a multi-cell network. Then it automatically communicates with all other RFPs of that network.

Note: If the system operator modifies the values of PIN and/or SMK at his FT, all previous PT subscriptions get invalid and must be renewed.

3.2.3 On-Air subscription of Portable Terminals

The firmware supports on-air subscription according to GAP. In on-air subscription the public information is transmitted by the FT over the air interface.

The on-air subscription procedure is described below.

Step 1	FT	Enable on-air subscription by setting SIAIR ON. Leave FT powered on.
Step 2	PT	Initiate on-air subscription by issuing a SISUA command. Result code <ok> signals successful subscription
Step 3	FT	Disable on-air subscription by setting SIAIR OFF or by leaving the configuration mode.

Air subscription also is set OFF on a reset of the FT module.

3.2.4 Offline subscription of Portable Terminals

The firmware supports a proprietary offline subscription procedure that works without transmitting information over the air interface. Therefore this technique is also applicable to situations were PT and FT are physically separated during subscription.

The offline subscription procedure requires the exchange of the SK which is read from the FT and input into the PT.

The offline subscription procedure is described below.

Step 1	FT	Get a SK for the PT by issuing a GISK command
Step 2	PT	Perform offline subscription by issuing a SISUB command. Result code <ok> signals successful command execution. This does not imply that the subscription itself was successful (e.g. PIN could be incorrect)

3.3 Configuration mode

In this mode parameters of the hardware are set and monitored by a controlling unit (e.g. a personal computer) connected to its RS-232 port.

3.3.1 Entering the configuration mode

The configuration mode is entered either by a specific reset sequence or by an appropriate escape command from any of the data sub-modes (see section 3.1.2).

When entered by an external reset, the RS-232 port is operated at 9.600 Bd independent from the parameter setting of the baud rate.

When entered by an escape command, the RS-232 port is operated at the configured baud rate.

3.3.2 Configuration protocol

Configuration mode uses a simple ASCII-based configuration protocol. The controlling unit (host) acts as master. The module acts as slave.

1. The master sends a configuration command. This is a valid command string as described in section 3.3.4. The command is terminated by **<CR><LF>**.
2. Depending on the command given, the module may respond with a response string. This is always terminated by **<CR><LF>**. The response string may contain multiple lines of text. In this case every line is terminated by **<CR><LF>**.
3. The module sends a return code. See section 3.3.4.2. The return code is terminated by **<CR><LF>**. This completes the command.
4. The protocol continues at step 1.

By successful completion of step 3 any modified configuration data has been saved in the non-volatile memory of the HW 86010/20.

3.3.3 Leaving the configuration mode

The configuration mode is terminated by a reset sequence or by the **EXIT** configuration command.

3.3.4 Configuration command overview

3.3.4.1 Commands

The commands provided in configuration mode are given below:

Command	Description
Hardware commands	
GHALL	Get Hardware Data (list)
GHTY	Get Hardware Moduletype
GHRSSI	Get RSSI value
GHQUAL	Get quality values
GHFL	Get Hardware Flashtype
SPANT	Set Antenna
GPANT	Get Antenna
Software-ID commands	
GSALL	Get Software Data (list)
GSNR	Get Software Number
GSVER	Get Software Version
Module commands	
GMALL	Get Module Data (list)
GMF	Get Module Frequency
GMQ	Get Module Quad-Tune
GMM	Get Module Modulation
Mode commands	
SPPR	Set Parameter Protocol Data Submode
GPPR	Get Parameter Protocol Data Submode
SPCTR	Set Parameter CTR 6 Testmode
GPCTR	Get Parameter CTR 6 Testmode
SPCLDPS	Set Parameter CLDPS
GPCLDPS	Get Parameter CLDPS
SPMP	Set Parameter Transparent Multipoint Mode
GPMP	Get Parameter Transparent Multipoint Mode
Info commands	
GNALL	Get Number Data (list)
GNEMC	Get Number EMC
GNETH	Get Ethernet MAC address
GNDNR	Get Number DECT Serial
GNSER	Get Number Production Serial
GNUNR	Get Serial Number of the Host Device

Command	Description
Identity commands	
SIAIR	Set Identity Air Subscription Accept
GIAIR	Get Identity Air Subscription Accept
SISUA	Set Identity Subscription On-Air
SISUB	Set Identity Subscription Offline
SISUD	Set Identity Subscription Offline Direct
GISUB	Get Identity Subscriptions (list)
DISUB	Delete Identity Subscription
GISK	Get Identity Subscription Key
SIPIN	Set Identity PIN
SISMK	Set Identity Subscription Master Key
GIPARK	Get Park of FT
SIARI	Replace ARI of RFP (for multi-cell operation)
GIARI	Get Key for ARI (needed for SIARI)
DIARI	Restores original ARI of FT (factory default ARI)
Voice commands	
SPVMIC	Set Voice Microphone Parameters
GPVMIC	Get Voice Microphone Parameters
SPVOICE	Set the Voice Mode Flag
GPVOICE	Get the Voice Mode Flag
SPVSPE	Set Voice Speaker Parameters
GPVSPE	Get Voice Speaker Parameters
SPVST	Set Voice Sidetone Parameters
GPVST	Get Voice Sidetone Parameters
Serial and IO commands	
SPBD	Set Parameter Baud Rate
GPBD	Get Parameter Baud Rate
IPBD	Information Supported Baud Rates (list)
SPCOM	Set Serial Communication Parameters
GPCOM	Get Serial Communication Parameters
SPCC	Set Parameter Call Control
GPCC	Get Parameter Call Control
SPECC	Set Parameter Enhanced Call Control
GPECC	Get Parameter Enhanced Call Control
SPUI	Set Parameter User Interface
GPUI	Get Parameter User Interface

Command	Description
Other configuration commands	
GPALL	Get Parameter Data (list)
SPTM	Set Parameter Termination
GPTM	Get Parameter Termination
SPDSI	Set Parameter Dial String Internal
GPDSI	Get Parameter Dial String Internal
DPDSI	Delete Parameter Dial String Internal
SPDSD	Set Parameter Dial String Default
GPDSD	Get Parameter Dial String Default
DPDSD	Delete Parameter Dial String Default
SPCUST	Set Customer String
GPCUST	Get Customer String
DPCUST	Delete Customer String
SPLOC	Set Parameter Location Registration
GPLOC	Get Parameter Location Registration
SPSYWD	Set Parameter Synchronisation Window
GPSYWD	Get Parameter Synchronisation Window
DPSCFG	Disable Config Mode Escape Sequence ‘+-+’
SPRETRY	Set Parameter Retry
GPRETRY	Get Parameter Retry
DPRETRY	Delete Parameter Retry
SPTIMEOUT	Set Parameter Timeout
GPTIMEOUT	Get Parameter Timeout
DPTIMEOUT	Delete Parameter Timeout
General commands	
GALL	Get All Data (list)
CRC	Do Firmware Diagnostics
GOK	Get OK (dummy command)
EXIT	Exit configuration mode

3.3.4.2 Return codes

The return codes provided in the configuration mode are given below:

Return Code	Description
OK	Command successful
ERROR 1	Command failed
ERROR 2	Command invalid
ERROR 3	Command parameter invalid
ERROR 4	Subscription table full
ERROR 21	Invalid character at command start
ERROR 22	Unexpected argument (Get or Information command)
ERROR 23	Argument missing (Set or Delete command)
ERROR 24	Wrong argument type
ERROR 25	Wrong number of arguments
ERROR 26	Internal conversion error
ERROR 41	First character of PARK invalid
ERROR 42	Second character of PARK invalid
ERROR 43	PARK length indicator too big
ERROR 44	Wrong character, where octal digit was expected
ERROR 45	Invalid character in checksum
ERROR 46	Wrong checksum
ERROR 47	Checksum too long
ERROR 49	SISUA timeout
ERROR 50	SIARI wrong key
ERROR 51	SISUA wrong PIN
ERROR 52	SISUA other error
ERROR 53	Unsuitable environment
ERROR 60	SPCUST string too long
ERROR 90	Command too long
ERROR 91	Init data corrupted
ERROR 93	Wrong firmware

3.3.4.3 Argument formats

Format	Description
octal	string containing characters ‘0’ to ‘7’
decimal	string containing characters ‘0’ to ‘9’
hexadecimal	string containing characters ‘0’ to ‘9’, ‘A’ to ‘F’
string	string containing characters ‘0’ to ‘9’, ‘A’ to ‘Z’ or special characters ‘.’, ‘*’, ‘#’, ‘-‘, ‘+’, ‘:’, ‘ ‘ (space)
PARK	specific format Pddoooooooooooooc A PARK always starts with a character ‘P’ followed by two decimal digits, followed by up to 12 octal digits and terminated by a check digit. The check digit is a decimal digit or ‘*’

Note: Characters are not case sensitive

3.3.5 Configuration commands reference

3.3.5.1 Hardware commands

3.3.5.1.1 All hardware parameters: GHALL

Syntax	**Command**	GHALL
	Response	list of parameters
Description	Get a list of all hardware parameters.	
Return value(s)		For an explanation of the response string see the respective Get commands of the individual parameters.
Example	Host: Module:	**GHALL** **Moduletype: 86010** **Flashtype: 278000BF SST39LF/VF400A** **OK**
Application	PT, FT	
Notes	• Format may be changed in future versions. • Because of possible changes in future, it is not advisable to implement this command directly in own software. All returned parameters are based on commands given in the manual.	

3.3.5.1.2 Module type: GHTY

Syntax	**Command**	GHTY
	Response	<type>
Description	Get the embedded module type.	
Return value(s)	**type**	Höft & Wessel product code format: string
Example	Host: Module:	**GHTY** **86010** **OK**
Application	PT, FT	
Notes		

3.3.5.1.3 Relative rssi value: GHRSSI

Syntax	**Command**	GHRSSI
	Response	<rssi>
Description	Get the relative rssi value.	
Return value(s)	**rssi**	Radio signal strength indication format: decimal
Example	Host: Module:	**GHRSSI** **129** **OK**
Application	PT	
Notes	• 0 if not synchronised • updated internally every second • The PT has to be subscribed to the FT which shall be measured.	

3.3.5.1.4 Receive quality: GHQUAL

Syntax	**Command**	GHQUAL
	Response	<ok>,<nok>
Description	Get the receive quality.	
Return value(s)	**ok**	Number of correct frames format: decimal
	nok	Number of incorrect frames format: decimal
Example	Host: Module:	**GHQUAL** **445,4** **OK**
Application	PT	
Notes	• The command self resets the counters. • The PT has to be subscribed to the FT which shall be measured.	

3.3.5.1.5 Flash memory type: GHFL

Syntax	**Command**	GHFL
	Response	<flash>
Description	Get the type of flash memory of the module.	
Return value(s)	**flash**	Manufacturer part number of the flash memory IC format: string
Example	Host: Module:	**GHFL** **AM29LV400** **OK**
Application	PT, FT	
Notes	• HW 86010 is delivered with different flash memory types. These are functionally equivalent.	

3.3.5.1.6 Antenna: SPANT / GPANT

Syntax	**Command**	SPANT <antenna>
	Response	none
Description	Select the antenna to be used by the hardware.	
Arguments	**antenna**	antenna **0**: internal antenna right **1**: internal antenna left **2**: external antenna **3**: antenna diversity (connection orientated) **4**: fast antenna diversity (CLDPS) format: decimal
Example	Host: Module:	**SPANT 0** **OK**
Application	PT, FT	
Notes	• Option 3: Antenna diversity is software controlled by the bit error rate (only PT). • Option 4: Antenna diversity is hardware controlled by fast (PT and FT).	

Syntax	**Command**	GPANT
	Response	<antenna>
Description	Get the information, which antenna is selected.	
Return value(s)	**antenna**	See SPANT
Example	Host: Module:	**GPANT** **0** **OK**
Application	PT, FT	
Notes	• See SPANT	

3.3.5.2 Software-ID commands

3.3.5.2.1 Software versioning parameters: GSALL

Syntax	**Command**	GSALL
	Response	list of all software versioning parameters
Description	Get a list of all software versioning parameters.	
Return value(s)		For an explanation of the response string see the respective Get commands of the individual parameters.
Example	Host: Module:	**GSALL** **Software: #26157 Aug 06 2004** **Version: 2.22** **OK**
Application	PT, FT	
Notes	• Format may be changed in future versions. • Because of possible changes in future, it is not advisable to implement this command directly in own software. All returned parameters are based on commands given in the manual.	

3.3.5.2.2 Firmware build: GSNR

Syntax	**Command**	GSNR
	Response	<snr>
Description	Get the firmware build number and date.	
Return value(s)	**snr**	Firmware build number and date. See example. format: string
Example	Host: Module:	**GSNR** **#26157 Aug 06 2004** **OK**
Application	PT, FT	
Notes		

3.3.5.2.3 Firmware version: GSVER

Syntax	**Command**	GSVER
	Response	<ver>
Description	Get the firmware version of the module.	
Return value(s)	**ver**	Firmware version information. See example format: string
Example	Host: Module:	**GSVER** **2.22** **OK**
Application	PT, FT	
Notes		

3.3.5.3 Module commands

3.3.5.3.1 Module parameters: GMALL

Syntax	**Command**	GMALL
	Response	list of all parameters
Description	Get a list of all module-specific parameters.	
Return value(s)		For an explanation of the response string see the respective Get commands of the individual parameters.
Example	Host: Module:	**GMALL** **Frequency: 112** **Quad-Tune: 130** **Modulation: 120,120,0** **OK**
Application	PT, FT	
Notes	• Values of module-specific parameters are adjusted during the production process. They may vary between different modules. • Format may be changed in future versions. • Because of possible changes in future, it is not advisable to implement this command directly in own software. All returned parameters are based on commands given in the manual.	

3.3.5.3.2 Module frequency: GMF

Syntax	**Command**	GMF
	Response	<freq>
Description	Get the module-specific frequency parameter.	
Return value(s)	**Freq**	Frequency parameter format: decimal
Example	Host: Module:	**GMF** **112** **OK**
Application	PT, FT	
Notes	• The frequency parameter is set during production in order to fine-adjust the local oscillator frequency of the module. • The GMF command serves for diagnostic purposes.	

3.3.5.3.3 Module quad tune: GMQ

Syntax	**Command**	GMQ
	Response	<quad>
Description	Get the module-specific quad-tune parameter.	
Return value(s)	**quad**	Quad-tune parameter format: decimal
Example	Host: Module:	**GMQ** **130** **OK**
Application	PT, FT	
Notes	• The quad-tune parameter is set during production in order to fine-adjust the frequency demodulator of the module. • The GMQ command serves for diagnostic purposes.	

3.3.5.3.4 Module modulation: GMM

Syntax	**Command**	GMM
	Response	<mod1>,<mod2>,<mod3>
Description	Get the module-specific modulation parameters.	
Return value(s)	**mod1..3**	Modulation parameter format: decimal
Example	Host: Module:	**GMM** **120,120,0** **OK**
Application	PT, FT	
Notes	• The modulation parameters are set during production in order to fine-adjust the frequency modulation of the module. • The GMM command serves for diagnostic purposes.	

3.3.5.4 Mode commands

3.3.5.4.1 Protocol mode flag: SPPR / GPPR

Syntax	**Command**	SPPR <on/off>
	Response	none
Description	Set the protocol mode flag.	
Arguments	**on**	Protocol data sub-mode selected format: string
	off	Transparent data sub-mode selected format: string
Example	Host: Module:	**SPPR ON** **OK**
Application	FT	
Notes	• Protocol mode normally is used only by FT.	

Syntax	**Command**	GPPR
	Response	<on/off>
Description	Get the protocol mode flag.	
Return value(s)	**on/off**	See SPPR
Example	Host: Module:	**GPPR** **OFF** **OK**
Application	PT, FT	
Notes		

3.3.5.4.2 Value of CTR6 test mode: SPCTR / GPCTR

Syntax	**Command**	SPCTR <on/off>
	Response	none
Description	Set the value of the CTR6 test mode flag.	
Arguments	**on**	Activate CTR6 test mode format: string
	off	Deactivate CTR6 test mode format: string
Example	Host: Module:	**SPCTR ON** **OK**
Application	PT, FT	
Notes	• The CTR6 test mode is a dedicated mode for standardised test of the DECT air interface. In CTR6 test mode the module interacts with CTR6 test equipment.	

Syntax	**Command**	GPCTR
	Response	<on/off>
Description	Get the value of the CTR6 test mode flag.	
Return value(s)	**on/off**	Value ON, if in CTR6 test mode, otherwise OFF format: string
Example	Host: Module:	**GPCTR** **OFF** **OK**
Application	PT, FT	
Notes	• See SPCTR	

3.3.5.4.3 Value of CLDPS flag: SPCLDPS / GPCLDPS

Syntax	**Command**	SPCLDPS <on/off>
	Response	none
Description	Set the value of the CLDPS flag.	
Arguments	**on**	Activate CLDPS format: string
	off	Deactivate CLDPS format: string
Example	Host: Module:	**SPCLDPS ON** **OK**
Application	PT, FT	
Notes		

Syntax	**Command**	GPCLDPS
	Response	<on/off>
Description	Get the value of the CLDPS flag.	
Return value(s)	**on/off**	Value ON, if in CLDPS mode, otherwise OFF format: string
Example	Host: Module:	**GPCLDPS** **ON** **OK**
Application	PT, FT	
Notes		

3.3.5.4.4 Multipoint flag: SPMP / GPMP

Syntax	**Command**	SPMP <on/off>
	Response	none
Description	Set the Multipoint flag.	
Arguments	**on**	Multipoint selected format: string
	off	Multipoint not selected format: string
Example	Host: Module:	**SPMP OFF** **OK**
Application	FT	
Notes	• All characters received at serial RX of PTs (1 ... 4) are transmitted over serial com from FT. • All characters received at serial RX of FT are transmitted over serial com from PTs (1 ... 4).	

Syntax	**Command**	GPMP
	Response	<on/off>
Description	Get the Multipoint flag.	
Return value(s)	**on/off**	See SPMP
Example	Host: Module:	**GPMP** **OFF** **OK**
Application	FT	
Notes	• See SPMP	

3.3.5.5 Info commands

3.3.5.5.1 Serial number parameters: GNALL

Syntax	**Command**	GNALL
	Response	list of all serial number parameters
Description	Get a list of all serial number parameters.	
Return value(s)		For an explanation of the response string see the respective Get commands of the individual parameters.
Example	Host: Module:	**GNALL** **EMC: 322** **DectNo: 524752** **SerNo: 1043756** **OK**
Application	PT, FT	
Notes	• The DECT serial number in combination with the EMC is unique. The production serial number is unique. • Format may be changed in future versions. • Because of possible changes in future, it is not advisable to implement this command directly in own software. All returned parameters are based on commands given in the manual.	

3.3.5.5.2 European manufacturer: GNEMC

Syntax	**Command**	GNEMC
	Response	<emc>
Description	Get the European manufacturer code of the module.	
Return value(s)	**emc**	European manufacturer code format: decimal
Example	Host: Module:	**GNEMC** **322** **OK**
Application	PT, FT	
Notes	• The DECT serial number in combination with the EMC is unique.	

3.3.5.5.3 MAC address: GNETH

Syntax	**Command**	GNETH
	Response	<eth>
Description	Get the ethernet MAC address.	
Return value(s)	**eth**	ethernet address format: hex-special, see example
Example	Host: Module:	**GNETH** **00-30-2e-fb-b0-58** **OK**
Application	PT, FT	
Notes	• Relevant to CLDPS operation.	

3.3.5.5.4 DECT serial number: GNDNR

Syntax	**Command**	GNDNR
	Response	<dnr>
Description	Get the DECT serial number of the module.	
Return value(s)	**dnr**	DECT serial number format: decimal
Example	Host: Module:	**GNDNR** **524752** **OK**
Application	PT, FT	
Notes	• The DECT serial number in combination with the EMC is unique.	

3.3.5.5.5 Production serial number: GNSER

Syntax	**Command**	GNSER
	Response	<ser>
Description	Get the production serial number of the module.	
Return value(s)	**ser**	Production serial number format: decimal
Example	Host: Module:	**GNSER** **1043756** **OK**
Application	PT, FT	
Notes	• The production serial number is unique.	

3.3.5.5.6 Unit number: GNUNR

Syntax	**Command**	GNUNR
	Response	<unitno>
Description	Get the unit number of the module.	
Return value(s)	**unitno**	unit number format: decimal
Example	Host: Module:	**GNUNR** **10723607** **OK**
Application	PT, FT	
Notes	- For hosted applications the <unitno> is the serial number of the Höft & Wessel host device (e. g. HW 8612).	

3.3.5.6 Identity commands

3.3.5.6.1 Air subscription accept: SIAIR / GIAIR

Syntax	**Command**	SIAIR <on/off>
	Response	none
Description	Enable or disable on-air subscriptions.	
Arguments	**on**	Enable on-air subscriptions format: string
	off	Disable on-air subscriptions format: string
Example	Host: Module:	**SIAIR ON** **OK**
Application	FT	
Notes	• System security is improved, if on-air subscriptions are disabled during normal operation and only enabled in case of a particular need to subscribe a PT with the FT. • This flag is also set OFF after each module reset.	

Syntax	**Command**	GIAIR
	Response	<on/off>
Description	Get the value of the AIR flag. This flag determines whether a FT is enabled to accept on-air subscriptions of PTs. During normal operation the flag is OFF. It must be switched to ON before the on-air subscription is invoked.	
Return value(s)	**on/off**	See SIAIR
Example	Host: Module:	**GIAIR** **OFF** **OK**
Application	FT	
Notes	• Refer to SISUA command for on-air subscription.	

3.3.5.6.2 Air subscription identified by PARK: SISUA / SISUB / SISUD / GISUB / DISUB

Syntax	**Command**	SISUA <park>, <pin> SISUA <pin>
	Response	none
Description	Perform an on-air subscription of a PT at the FT identified by its PARK.	
Arguments	**park**	PARK code of the FT (optional) format: PARK
	pin	PIN code, 1 to 8 digits format: decimal
Example 1	Host: Module:	**SISUA P360002410000010, 007** **OK**
Example 2	Host: Module:	**SISUA 007** **OK**
Application	PT	
Notes	• Leading zeros of PIN codes are relevant, e.g. PIN 007 is different from PIN 7. • During on-air subscription the PT communicates with the FT over the air interface. This requires the FT to be operated in configuration or data transmission mode. • The FT must have on-air subscriptions enabled (see SIAIR configuration command). Otherwise it will reject the on-air subscription. • The PIN code must be identical to the PIN code programmed in the FT. Otherwise the FT will reject the on-air subscription. • SISUA only with parameter <pin> but without <park> let the PT subscribe to the first FT it finds, which has set access rights on with SIAIR ON. • In case of many FTs in a multi-cell network possibly the command has to be repeated.	

Syntax	**Command**	SISUB <emc>,<dnr>,<pin>,<sk> SISUB <park>
	Response	none
Description	Perform an offline subscription of a PT at the FT identified by its EMC and DECT serial number or identified by its PARK.	
Arguments	**emc**	EMC code of the FT to which the PT shall be subscribed format: decimal
	dnr	DECT serial number of the FT to which the PT shall be subscribed format: decimal
	pin	PIN code, 1 to 8 digits format: decimal
	sk	SK key for offline subscription of the PT format: decimal
	park	PARK code of the FT format: PARK
Example 1	Host: Module:	**SISUB 322, 847544, 007, 8439554** **OK**
Example 2	Host: Module:	**SISUB 360002413167270** **OK**
Application	PT	
Notes	• Leading zeros of PIN codes are relevant, e.g. PIN 007 is different from PIN 7. • The SK key is obtained from the FT by use of the GISK configuration command. • The PIN code must be identical to the PIN code programmed in the FT. • It's also possible to use <park> instead of <emc> and <dnr>, which is useful for an offline ARI-B subscription..	

Syntax	**Command**	SISUD <emc>,<dnr>,<pin> SISUD <park>
	Response	none
Description	Perform a direct offline subscription of a PT at the FT identified by its EMC and DECT serial number or identified by its PARK. Only useable if SMK is 00000000.	
Arguments	**emc**	EMC code of the FT to which the PT shall be subscribed format: decimal
	dnr	DECT serial number of the FT to which the PT shall be subscribed format: decimal
	pin	PIN code, 1 to 8 digits format: decimal
	park	PARK code of the FT (optional) format: PARK
Example 1	Host: Module:	**SISUD 322, 847544, 007** **OK**
Example 2	Host: Module:	**SISUD P360002413167270** **OK**
Application	PT	
Notes	• Leading zeros of PIN codes are relevant, e.g. PIN 007 is different from PIN 7. • The PIN code must be identical to the PIN code programmed in the FT. • It's also possible to use <park> instead of <emc> and <dnr>, which is useful for an offline ARI-B subscription.	

Syntax	**Command**	GISUB
	Response	List of records of type <emc>, <dnr>, <pli>
Description	Get a list of the PARK table of a PT.	
Return value(s)	**emc**	EMC code of the FT identified in the subscription record format: decimal
	dnr	DECT serial number of the FT identified in the subscription record format: decimal
	pli	PARK length indicator format: decimal
Example	Host: Module:	**GISUB** **P360002413167270# (322,847544,36)** **OK**
Application	PT	
Notes	• See section 3.2.1.1 for an explanation of the PLI parameter	

Syntax	**Command**	DISUB all DISUB <emc>, <dnr>
	Response	none
Description	Delete a subscription entry from the PARK table of the PT. This clears an existing subscription of the PT from the referred FT. The alternative syntax formats allow the FT to be identified either by the pair <emc>, <dnr> or by its <park>. Both formats are equivalent.	
Arguments	**all**	All subscriptions will be deleted from the PARK table.
	emc	EMC code of the FT to be deleted from PARK table format: decimal
	dnr	DECT serial number of the FT to be deleted from PARK table format: decimal
Example 1	Host: Module:	**DISUB all** **OK**
Example 2	Host: Module:	**DISUB 322, 847544** **OK**
Application	PT	
Notes	• For easiest unsubscribtion use the argument ALL. • If the FT referred in the DISUB command was not contained in the PARK table, an error code is returned.	

3.3.5.6.3 Subscription key: GISK

Syntax	**Command**	GISK <emc>, <dnr>
	Response	<sk>
Description	Get a Subscription Key (SK) for the PT identified by EMC and DECT serial number. The SK is required as an argument for offline subscription.	
Return value(s)	**emc**	EMC code of the PT that shall be subscribed format: decimal
	dnr	DECT serial number of the PT that shall be subscribed format: decimal
	sk	SK key for offline subscription of the PT format: decimal
Example	Host: Module:	**GISK 322, 847544** **8439554** **OK**
Application	FT	
Notes	• For offline subscription see SISUB command.	

3.3.5.6.4 Identity PIN: SIPIN

Syntax	**Command**	SIPIN <pin>
	Response	none
Description	Program a new PIN code into the FT. The PIN code is needed by PTs that want to subscribe to the FT.	
Arguments	**pin**	PIN code, 1 to 8 digits format: decimal
Example	Host: Module:	**SIPIN 007** **OK**
Application	FT	
Notes	• Leading zeros of PIN codes are relevant, e.g. PIN 007 is different from PIN 7	

3.3.5.6.5 Subscription master key: SISMK

Syntax	**Command**	SISMK <smk>
	Response	none
Description	Program a new SMK code into the FT. The SMK code improves the system security (see section 3.2.2).	
Arguments	**smk**	SMK code, 1 to 8 digits format: decimal
Example	Host: Module:	**SISMK 0815** **OK**
Application	FT	
Notes	• Leading zeros of SMK codes are relevant, e.g. SMK 0815 is different from SMK 815	

3.3.5.6.6 PARK of FT: GIPARK

Syntax	**Command**	GIPARK
	Response	<park>
Description	Get the Park of a FT.	
Return value(s)	**park**	PARK code of the FT format: PARK
Example	Host: Module:	**GIPARK** **P360002413167270#** **OK**
Application	FT	
Notes	• The result can be used for all PT subscription commands like <SISUA>, <SISUD>	

3.3.5.6.7 Access rights identity: SIARI / GIARI / DIARI

Syntax	**Command**	SIARI <park>, <key>, <rpn>
	Response	none
Description	Overwrites the ARI of a RFP with the ARI of another RFP (for multi-cell networks).	
Arguments	**park**	PARK code of the FT format: PARK
	key	key for setting ARI (get it with GIARI) format: decimal
	rpn	radio fixed part number format: decimal
Example	Host: Module:	**SIARI P360002410000010#,327,3** **OK**
Application	FT	
Notes	• rpn is 0 for standalone RFP • rpn is 1 ... 7 for multi-cell networks	

Syntax	**Command**	GIARI
	Response	<park>, <key>, <rpn>
Description	Get information to set other RFP to same ARI (for multi-cell networks).	
Return value(s)	**park,** **key,** **rpn**	See SIARI
Example	Host: Module:	**GIARI** **P360002410000010#,327,3** **OK**
Application	FT	
Notes	• See SIARI	

Syntax	**Command**	DIARI
	Response	none
Description	Resets the ARI of a RFP to its factory-programmed value (corresponding to emc and dectnr of RFP).	
Arguments		none
Example	Host: Module:	**DIARI** **OK**
Application	FT	
Notes		

3.3.5.7 Voice commands

3.3.5.7.1 Voice microphone parameters: SPVMIC/ GPVMIC

Syntax	**Command**	SPVMIC <micvol>,<micatt>,<micref>, <mode>
	Response	none
Description	Set Voice MICrophone Parameters.	
Arguments	**micvol**	Preamp gain: amplification or attenuation depending on <mode> in amplification mode 0 ... 15 in attenuation mode 0 ... 7 (highest bit not used) format: decimal
	micatt	Attenuation level of digital post filter in 0.5 dB steps (0 --> 0 dB, 2 --> -1 dB, 5...7 --> -2.5 dB) format: decimal (0 ... 7)
	micref	microphone reference trimming in 0.17 dB steps format: decimal (0 ... 7)
	mode	0: amplification mode in 3 dB steps 1: attenuation mode in 2 dB steps format: decimal (0 ... 1)
Example	Host: Module:	**SPVMIC 0,0,2,0** **OK**
Application	PT, FT	
Notes	• Please use amplification mode for better voice quality. • Voice mode has to be enabled to use this command. Check voice enabled using GPVOICE command.	

Syntax	**Command**	GPVMIC
	Response	<micvol>,<micatt>,<micref>, <mode>
Description	Get Voice MICrophone Parameters.	
Return value(s)	**micvol, micatt, micref, mode**	See SPVMIC
Example	Host: Module:	**GPVMIC** **0,0,2,0** **OK**
Application	PT, FT	
Notes	• See SPVMIC	

3.3.5.7.2 Voice mode flag: SPVOICE / GPVOICE

Syntax	**Command**	SPVOICE <on/off>
	Response	none
Description	Set the VOICE mode flag.	
Arguments	**on**	Voice on format: string
	off	Voice off, data transmission on format: string
Example	Host: Module:	**SPVOICE ON** **OK**
Application	PT, FT	
Notes	• SPVOICE ON enables all other voice commands.	

Syntax	**Command**	GPVOICE
	Response	<on/off>
Description	Get the VOICE mode flag.	
Return value(s)	**on/off**	See SPVOICE
Example	Host: Module:	**GPVOICE** **ON** **OK**
Application	PT, FT	
Notes	• See SPVOICE	

3.3.5.7.3 Voice speaker: SPVSPE / GPVSPE

Syntax	**Command**	SPVSPE <volafe>,<volpcm>
	Response	none
Description	Set Voice Speaker Parameters.	
Arguments	**volafe**	VBAFE output volume speaker in 2 dB steps format: decimal (0 ... 15)
	volpcm	receive volume format: decimal (0 ... 255)
Example	Host: Module:	**SPVSPE 8,128** **OK**
Application	PT, FT	
Notes	• Voice mode has to be enabled to use this command. Check voice enabled using GPVOICE command.	

Syntax	**Command**	GPVSPE
	Response	<volafe>,<volpcm>
Description	Get Voice Speaker Parameters.	
Return value(s)	**volafe, volpcm**	See SPVSPE
Example	Host: Module:	**GPVSPE** **8,128** **OK**
Application	PT, FT	
Notes	• Voice mode has to be enabled to use this command. Check voice enabled using GPVOICE command.	

3.3.5.7.4 Voice sidetone: SPVST / GPVST

Syntax	**Command**	SPVST <stena>,<stgai>,<stsca>
	Response	none
Description	Set Voice SideTone Parameters.	
Arguments	**stena**	sidetone enable 0: sidetone off 1: sidetone on format: decimal (0 ... 1)
	stgai	sidetone gain format: decimal (0 ... 31)
	stsca	global tone scale factor format: decimal (0 ... 3)
Example	Host: Module:	**SPVST 1,16,0** **OK**
Application	PT, FT	
Notes	• Voice mode has to be enabled to use this command. Check voice enabled using GPVOICE command.	

Syntax	**Command**	GPVST
	Response	<stena>,<stgai>,<stsca>
Description	Get Voice SideTone Parameters	
Return value(s)	**stena, stgai, stsca**	See SPVST
Example	Host: Module:	**GPVST** **1,16,0** **OK**
Application	PT, FT	
Notes	• Voice mode has to be enabled to use this command. Check voice enabled using GPVOICE command.	

3.3.5.8 Serial and IO commands

3.3.5.8.1 Baud rate SPBD / GPBD / IPBD

Syntax	**Command**	SPBD <baud>
	Response	none
Description	Set the value of the baud rate of the RS-232 interface.	
Arguments	**baud**	Baud rate in Bd. format: decimal
Example	Host: Module:	**SPBD 115200** **OK**
Application	PT, FT	
Notes	• In configuration mode and download mode the actually used baud rate may be different from the configured baud rate. • Change of baud rate becomes effective when the configuration mode is terminated. • <baud> must be a valid baud rate. The IPBD command returns a list of all allowed values.	

Syntax	**Command**	GPBD
	Response	<baud>
Description	Get the value of the baud rate of the RS-232 interface.	
Return value(s)	**baud**	See SPBD
Example	Host: Module:	**GPBD** **115200** **OK**
Application	PT, FT	
Notes	• In configuration mode and download mode the actually used baud rate may be different from the configured baud rate.	

Syntax	**Command**	IPBD
	Response	list of <baud>
Description	Information about selectable baud rates.	
Return value(s)	**baud**	See SPBD
Example	Host: Module:	**IPBD** **600,4800,9600,19200,38400,57600,115200** **OK**
Application	PT, FT	
Notes		

3.3.5.8.2 Serial communication: SPCOM / GPCOM

Syntax	**Command**	SPCOM <databit>,<parity>,<stopbit>, <handshake>
	Response	none
Description	Set Serial Communication Parameter.	
Arguments	**databit**	number of databits format: decimal (5 ... 8)
	parity	**N**: None **E**: Even **O**: Odd **M**: Mark **S**: Space format: char
	stopbit	**1**: one stopbit **2**: two stopbits format: decimal
	handshake	**RTSCTS** **NONE** format: string
Example	Host: Module:	**SPCOM 8, N, 1, RTSCTS** **OK**
Application	PT, FT	
Notes		

Syntax	**Command**	GPCOM
	Response	<databit>,<parity>,<stopbit>, <handshake>
Description	Get Serial Communication Parameter.	
Return value(s)	**databit,** **parity,** **stopbit,** **handshake**	See SPCOM
Example	Host: Module:	**GPCOM** **8, N, 1, RTSCTS** **OK**
Application	PT, FT	
Notes		

3.3.5.8.3 Parameter call control: SPCC / GPCC

Syntax	**Command**	SPCC <callctrl>
	Response	none
Description	Set parameter call control.	
Arguments	**callctrl**	**0**: connection only with DTRI **1**: connection without DTRI format: decimal
Example	Host: Module:	**SPCC 0** **OK**
Application	PT, FT	
Notes	• Normally connection is established only if DTRI is active. • If call control is set to 1, connection is also established without activation of DTRI.	

Syntax	**Command**	GPCC
	Response	<callctrl>
Description	Get parameter call control.	
Return value(s)	**callctrl**	See SPCC
Example	Host: Module:	**GPCC** **0** **OK**
Application	PT, FT	
Notes	• See SPCC	

3.3.5.8.4 Enhanced call control: SPECC / GPECC

Syntax	**Command**	SPECC <on/off>
	Response	none
Description	Set Enhanced Call Control.	
Arguments	**on**	Activate Enhanced Call Control format: string
	off	Deactivate Enhanced Call Control format: string
Example	Host: Module:	**SPECC ON** **OK**
Application	PT	
Notes	• If ON, DCD is active if PT has synchronisation. • If the PT receives a call from FT, it asserts RIIO. To accept the call the host must activate DTRI. After DTRI is active, RIIO goes inactive. • IF OFF DCD and RIIO are configured as normal outputs on PTs and follow the corresponding inputs at FT side.	

Syntax	**Command**	GPECC
	Response	<on/off>
Description	Get Enhanced Call Control flag.	
Return value(s)	**on/off**	See SPECC
Example	Host: Module:	**GPECC** **ON** **OK**
Application	PT	
Notes	• See SPECC	

3.3.5.8.5 User interface: SPUI / GPUI

Syntax	**Command**	SPUI <led>,<key>
	Response	none
Description	Set user interface.	
Arguments	**led**	**0:** led interface off, GPIO1 and GPIO2 are normal GPIOs **1:** led interface on, GPIO1 controls the connection led (normally green) GPIO2 controls the configuration led (normally red) format: decimal (0 ... 1)
	key	**0:** key interface off, ADR19 has no special function **1:** key interface on, ADR19 is used for key interface (subscription functionality) format: decimal (0 ... 1)
Example	Host: Module:	**SPUI 1,1** **OK**
Application	PT, FT	
Notes	- GPIOs are handled automatically in some Höft&Wessel devices.	

Syntax	**Command**	GPUI
	Response	<led>,<key>
Description	Get user interface.	
Return value(s)	**led,** **key**	See SPUI
Example	Host: Module:	**GPUI** **1,1** **OK**
Application	PT, FT	
Notes		

3.3.5.9 Other configuration commands

3.3.5.9.1 Configurable parameters: GPALL

Syntax	**Command**	GPALL
	Response	list of all configurable parameters
Description	Get a list of all configurable parameters.	
Return value(s)		For an explanation of the response string see the respective Get commands of the individual parameters.
Example	Host: Module:	**GPALL** **Baudrate : 115200** **Com : 8,N,1,RTSCTS** **Antenna : 0** **ChannelCnt : 1** **Termination : PT** **Equipment : DCE (var)** **CLDPS : OFF** **Protocol : OFF** **CTR6 : OFF** **Enhanced CC: OFF** **OK**
Application	PT, FT	
Notes	• Format may be changed in future versions. • Because of possible changes in future, it is not advisable to implement this command directly in own software. All returned parameters are based on commands given in the manual.	

3.3.5.9.2 Type of DECT termination: SPTM / GPTM

Syntax	**Command**	SPTM <PT/FT>
	Response	none
Description	Set the type of DECT termination.	
Arguments	**PT**	Module operates as PT format: string
	FT	Module operates as FT format: string
Example	Host: Module:	**SPTM PT** **OK**
Application	PT, FT	
Notes	• After a successful SPTM command the module automatically performs a reset into config mode.	

Syntax	**Command**	GPTM
	Response	<PT/FT>
Description	Get the type of DECT termination.	
Return value(s)	**term**	See SPTM
Example	Host: Module:	**GPTM** **PT** **OK**
Application	PT, FT	
Notes		

3.3.5.9.3 Dial string internal: SPDSI / GPDSI / DPDSI

Syntax	**Command**	SPDSI <emc>,<dnr>
	Response	none
Description	Set Dial String for Internal call in transparent mode. This defines the -PT to which the FT will establish a connection upon activation of DTRI.	
Arguments	**emc**	EMC code of the PT which shall be called from FT. format: decimal
	dnr	DECT serial number of the PT which shall be called from FT. format: decimal
Example	Host: Module:	**SPDSI 322, 695432** **OK**
Application	FT	
Notes	• The dial string stored in internal RAM and deleted with hard reset. • Enter config mode with '+-+' and leave it after SPDSI with command EXIT. • Leaving the config mode with hard reset will delete the dial string. PT must be subscribed to FT. • For permanently storing the dial string use SPDSD.	

Syntax	**Command**	GPDSI
	Response	<emc>,<dnr>
Description	Get Dial String for Internal call in transparent mode	
Return value(s)	**emc,** **dnr**	See SPDSI
Example	Host: Module:	**GPDSI** **322,695432** **OK**
Application	FT	
Notes		

Syntax	**Command**	DPDSI
	Response	none
Description	Delete Dial String for Internal call in transparent mode.	
Arguments		none
Example	Host: Module:	**DPDSI** **OK**
Application	FT	
Notes		

3.3.5.9.4 Dial string default: SPDSD / GPDSD / DPDSD

Syntax	**Command**	SPDSD <emc>,<dnr>
	Response	none
Description	Set Dial String for Default call in transparent mode. This defines the PT to which the FT will establish a connection upon activation of DTRI. The dial string is stored in non-volatile memory, i.e. it remains stored even after power off.	
Arguments	**emc**	EMC code of the PT which shall be called from FT. format: decimal
	dnr	DECT serial number of the PT which shall be called from FT. format: decimal
Example	Host: Module:	**SPDSD 322, 695432** **OK**
Application	FT	
Notes	• PT must be subscribed to FT. • SPDSI has priority over SPDSD. • Frequent write operations to non-volatile memory reduce the life time of the hardware and should be avoided. Use SPDSI, if you do not need to store the dial string.	

Syntax	**Command**	GPDSD
	Response	<emc>,<dnr>
Description	Get Dial String for Default call in transparent mode.	
Return value(s)	**emc,** **dnr**	See SPDSD
Example	Host: Module:	**GPDSD** **322,695432** **OK**
Application	FT	
Notes		

Syntax	**Command**	DPDSD
	Response	none
Description	Delete Dial String for Default call in transparent mode.	
Arguments		none
Example	Host: Module:	**DPDSD** **OK**
Application	FT	
Notes		

3.3.5.9.5 Customer string: SPCUST / GPCUST / DPCUST

Syntax	**Command**	SPCUST <custstr>
	Response	none
Description	Set the customer string.	
Arguments	**custstr**	Customer string, max. 20 characters format: string
Example	Host: Module:	**SPCUST production test** **OK**
Application	PT, FT	
Notes		

Syntax	**Command**	GPCUST
	Response	<custstr>
Description	Get the customer string.	
Return value(s)	**custstr**	See SPCUST
Example	Host: Module:	**GPCUST** **production test** **OK**
Application	PT, FT	
Notes		

Syntax	**Command**	DPCUST
	Response	none
Description	Delete the customer string.	
Arguments		none
Example	Host: Module:	**DPCUST** **OK**
Application	PT, FT	
Notes		

3.3.5.9.6 Location flag: SPLOC / GPLOC

Syntax	**Command**	SPLOC <on/off>
	Response	none
Description	Set the Location registration flag.	
Arguments	**on**	Location selected format: string
	off	Location not selected format: string
Example	Host: Module:	**SPLOC ON** **OK**
Application	FT	
Notes	• Only useable in protocol mode.	

Syntax	**Command**	GPLOC
	Response	<on/off>
Description	Get the Location flag.	
Return value(s)	**on/off**	See SPLOC
Example	Host: Module:	**GPLOC** **ON** **OK**
Application	FT	
Notes	• See SPLOC	

3.3.5.9.7 Parameter sync. windows: SPSYWD / GPSYWD

Syntax	**Command**	SPSYWD <syncwnd>
	Response	none
Description	Set additional Synchronisation Window.	
Arguments	**syncwnd**	Additional bits for Synchronisation Window format: decimal (0 ... 56)
Example	Host: Module:	**SPSYWD 0** **OK**
Application	FT	
Notes	• Useful only with special wide range antennas. • The normal size of the FT sync window is +/- 3 Bit, so the maximum distance between FT and PT is approximately 350 m. • For greater distances you have to increase the site of the sync window with the command <SPSYWD>. Exemplary distance calculations:	

Distance in meter	<syncwnd>
1000	5
2000	13
3000	21
4000	28
5000	36
6000	44
7000	51
7500	55

Syntax	**Command**	GPSYWD
	Response	<syncwnd>
Description	Get additional Synchronisation Window	
Return value(s)	**syncwnd**	See SPSYWD
Example	Host: Module:	**GPSYWD** **0** **OK**
Application	FT	
Notes	• See SPSYWD	

3.3.5.9.8 DPSCFG command

Syntax	**Command**	DPSCFG
	Response	none
Description	Disable Soft Way To Config Mode. Sets a variable in volatile memory, if this is set, switching from data mode to config mode with '+-+' sequence is impossible.	
Arguments		none
Example	Host: Module:	**DPSCFG** **OK**
Application	PT, FT	
Notes	• usage: enter config mode with '+-+', send this command, exit with EXIT. • The command is helpful for cascaded lines of DECT modules. • After a hardware reset '+-+' is always enabled.	

3.3.5.9.9 Retry value: SPRETRY / GPRETRY / DPRETRY

Syntax	**Command**	SPRETRY <no> SPRETRY <no>,<retry>
	Response	none
Description	Set the retry value.	
Arguments	**no**	**1:** DECT IP layer retry. After expiration of retries packet is discarded. Value 255 means endless retry. default: 6, connection orientated only **2:** LAP layer disconnect retry. After expiration of retries LAP signals a disconnect to application (please see SPTIMEOUT 2). Value 255 means endless retry. default: 6 **5:** CLDL1 layer unicast packet retry. default: 4, CLDPS only **6:** CLDL1 layer force modus unicast packet retry. default: 4, CLDPS only **7:** CLDL1 layer broadcast packet retry. For highest speed broadcast transmission from FT to PT side, use SPRETRY 7,0 (don't retry broadcast packets). default: 2, CLDPS only, FT only format: decimal 1,2,5,6,7
	retry	Retry value (optional). format: decimal 0 ... 253, 255
Example 1	Host: **SPRETRY 2** Module: **OK**	
Example 2	Host: **SPRETRY 2,7** Module: **OK**	
Application	Restrictions according to parameter <no>.	
Notes	- Per default use only retry 7.	

Syntax	**Command**	GPRETRY<no>
	Response	<no>,<retry>
Description	Get the retry value.	
Return value(s)	**no, retry**	See SPRETRY
Example	Host: Module:	**GPRETRY** **2,6** **OK**
Application	Restrictions according to parameter <no>.	
Notes		

Syntax	**Command**	DPRETRY
	Response	none
Description	Set all retry values to default.	
Arguments		none
Example	Host: Module:	**DPRETRY** **OK**
Application	PT, FT	
Notes		

3.3.5.9.10 Timeout value: SPTIMEOUT / GPTIMEOUT / DPTIMEOUT

Syntax	**Command**	SPTIMEOUT <no>,<time>
	Response	none
Description	Set the timeout value.	
Arguments	**no**	**2:** LAP layer repeat timeout. After lapse of time without packet confirmation packet transmission is repeated and retry counter is incremented (please see SPRETRY 2). With default combination of 6 retries and 500 ms LAP disconnect time is 3 seconds. default: 500 **3:** Connection termination disconnect timeout. After deactivation of DTRI, normally a timeout of 5000 milliseconds starts, then the module disconnects. So it is possible to transmit short deactivations of DTRI. With this timeout it is possible to change the time. default: 5000 **4:** Connection establishment reconnect timeout. When DTRI of PT is active and DTRI of FT is inactive, FT rejects connection request. After rejection the PT waits some time before next connection request. default: 1000, PT only **5:** SWAP keepalive timeout. If greater than 0 special probe packets are expected to be received from the SWAP server to check if the connection is still alive. If no packet is received within the timeout, the current connection is terminated and a new connection established. It is vital to configure the same keepalive timeout at the SWAP service. If the timeout value is 0 this option is disabled. default: 0, PT only format: decimal 0 ... 5
	time	in milliseconds, must be divisible by 100 format: decimal 0 ... 25400
Example	Host:	**SPTIMEOUT 3, 5000**
	Module:	**OK**
Application	PT, FT	
Notes	• Per default use only timeout 3. • Timeout value 4 concerns PT only.	

Syntax	**Command**	GPTIMEOUT <no>
	Response	<time>
Description	Get timeout value.	
Return value(s)	**no, time**	See SPTIMEOUT
Example	Host: Module:	**GPTIMEOUT 3** **5000** **OK**
Application	PT, FT	
Notes	• Timeout value 4 concerns PT only.	

Syntax	**Command**	DPTIMEOUT
	Response	none
Description	Set all timeout values to default.	
Arguments		none
Example	Host: Module:	**DPTIMEOUT** **OK**
Application	PT, FT	
Notes		

3.3.5.10 General commands

3.3.5.10.1 All data: GALL

Syntax	Command	GALL
	Response	list of selected parameters
Description	Get a list of all parameters.	
Return value(s)		For an explanation of the response string see the respective Get commands of the individual parameters.

Example

```
Host:      GALL
Module:    Modultype : 86010
           Flashtype  : 278000BF SST39LF/VF400A
           Software   : #26157 Aug 06 2004
           Version        : 2.22
           Frequency      : 65
           Quad-Tune      : 177
           Modulation: 111,111,0
           EMC            : 322
           DectNo         : 735312
           SerNo          : 10642194
           Baudrate       : 115200
           Com            : 8,N,1,RTSCTS
           Antenna        : 1
           ChannelCnt     : 1
           Termination    : FT
           Equipment      : DCE (var)
           CLDPS          : ON
           Protocol       : OFF
           CTR6           : OFF
           Location       : OFF
           Multipoint     : OFF
           Park: P36000241263412*#
           ARI : P36000241263412*#,35808,0
           OK
```

Application	PT, FT
Notes	• Format may be changed in future versions. • Because of possible changes in future, it is not advisable to implement this command directly in own software. All returned parameters are based on commands given in the manual.

3.3.5.10.2 Firmware diagnostics: CRC

Syntax	**Command**	CRC
	Response	<crc>
Description	Perform a self check of the firmware by computing a cyclic redundancy checksum over the content of the Flash EPROM.	
Return value(s)	**crc**	Computed 16 Bit cyclic redundancy checksum format: hexadecimal
Example	Host: Module:	**CRC** **EF12** **OK**
Application	PT, FT	
Notes		

3.3.5.10.3 Result code Ok: GOK

Syntax	**Command**	GOK
	Response	none
Description	Get a result code OK from the module.	
Return value(s)		none
Example	Host: Module:	**GOK** **OK**
Application	PT, FT	
Notes	• The GOK command is typically used as an alive request. GOK always returns OK and does not cause any changes to the module configuration.	

3.3.5.10.4 Exit configuration mode: EXIT

Syntax	**Command**	EXIT
	Response	none, module performs a reset
Description	Leave the configuration mode and enter the data mode. The entered data sub-mode depends on the setting of the PPR parameter (see SPPR). When the command is successful, a reset is performed immediately and no return code is given by the module.	
Return value(s)		none
Example	Host: Module:	**EXIT** performs a reset to data mode
Application	PT, FT	
Notes	• Parameter changes made by certain configuration commands only become effective at the exit command. This is described in the Notes section of the respective commands.	

3.4 Transparent data mode

The transparent data mode is a sub-mode of the data mode. It allows transparent data transmission using the RS-232 interface. The transparent data mode is selected by issuing the configuration command SPPR OFF.

Both, PT and FT may operate in transparent data mode. Moreover PT and FT may be operated in different data modes, e.g. a PT in transparent data mode may connect to a FT in protocol data mode.

In the transparent data mode, all data is treated as a stream. No specific framing is required.

Call control is provided by the modem lead lines.

This mode is restricted to a single connection (point-to-point). This means that even a FT only supports a single connection when operated in transparent data mode.

3.4.1 Usage of RS-232 interface

3.4.1.1 Connection of the interface

The RS-232 interface can be switched between two modes: DTE (data terminal equipment) and DCE (data communication equipment). Switching is controlled by firmware using the SPEQ configuration command.

In DTE mode the interface behaves like the COM port of a PC, i.e. DCDIO and RIIO are inputs of HW 86010/20.

In DCE mode the interface behaves like the RS-232 port of a modem, i.e. DCDIO and RIIO are outputs of HW 86010/20.

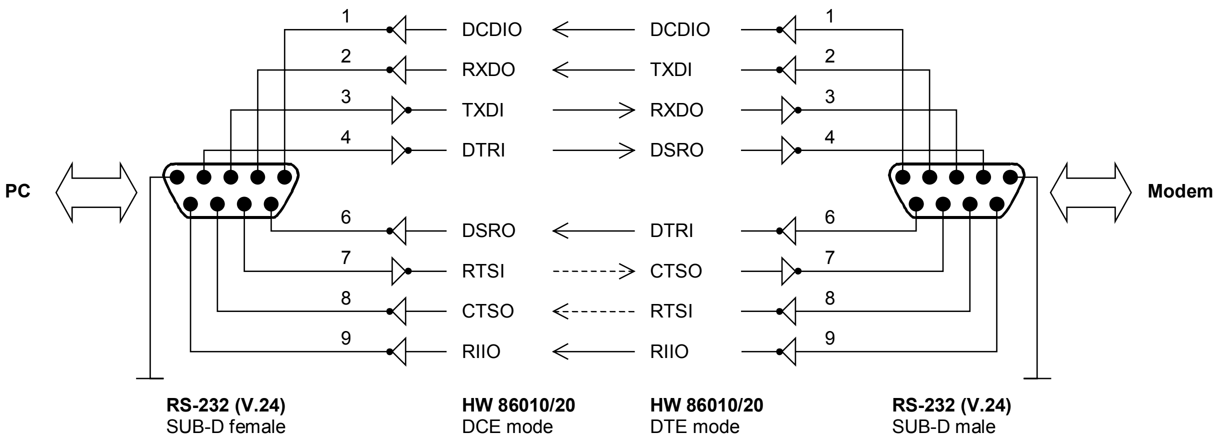

PC
1
DCDIO
DCDIO
1
2
RXDO
TXDI
2
3
TXDI
RXDO
3
4
DTRI
DSRO
4
6
DSRO
DTRI
6
7
RTSI
CTSO
7
8
CTSO
RTSI
8
9
RIIO
RIIO
9
Modem
RS-232 (V.24)
SUB-D female
HW 86010/20
DCE mode
HW 86010/20
DTE mode
RS-232 (V.24)
SUB-D male

Figure 1: RS-232 port set-up in DCE mode (can be connected to a PC) and DTE mode (can be connected to a Modem)

3.4.1.2 Interface parameters

The baud rate of the RS-232 interface is selected using the SPBD configuration command. The actual baud rate can be retrieved with the GPBD command. A list of available baud rates is shown in response to the IPBD command.

The baud rate setting is a local matter, i.e. the two peers of a connection may use different baud rates at their ends.

The direction of the DCDIO and RIIO signals is set using the SPEQ command. The actual setting can be retrieved with the GPEQ command.

3.4.2 Flow Control

For flow control on the RS-232 interface the HW 86010/20 uses hardware handshake (RTS/CTS). The hardware handshake signals are active low (usual polarisation in TTL level RS-232 interfaces).

The following description applies to hardware handshake.

Whenever the host deactivates RTSI (RTSI goes high), the HW 86010/20 will stop output of data after the current data byte. Due to pipelining it may happen that some additional bytes are output before the module stops. Data output is resumed as soon as the module senses an active RTSI again.

Whenever HW 86010/20 deactivates CTSO (CTSO goes high), the host shall stop output of data. HW 86010/20 tolerates up to 16 bytes being output by the host after deactivation of CTSO has occurred. The module activates CTSO again as soon as it is ready to accept more data from the host.

RTS/CTS handshake is used for local flow control between the module and the connected host and not directly inter-worked through the DECT link.

In case the host is not ready to accept data from the module and has deactivated RTSI, the module continues to accept data from its peer until its internal data buffers are filled. Then it will apply DECT flow control which stops data transmission from the peer.

Hardware flow control can be switched by using the command <SPCOM>.

The peer module continues to accept data from its host (the peer host), until its internal data buffers are filled. Finally the peer module deactivates CTSO. This signals the peer host to stop data transmission.

When the host gets ready to accept data and has activated RTSI, the internal data buffers of the modules are emptied before the peer module activates CTSO. This signals the peer host to resume data transmission.

3.4.3 Interworking of modem lead signals

In transparent data mode the modem lead signals are available on the DTRI, DSRO, DCDIO and RIIO pins.

DTRI, DCDIO and RIIO signals are interworked to the peer module. DTRI is always interworked to DSRO. DCDIO and RIIO are interworked to DCDIO and RIIO respectively.

Interworking RIIO and DCDIO requires that one module is configured in DTE mode and the peer module in DCE mode. In case both peers are configured in DCE mode, RIIO and DCDIO outputs remain inactive. In case both peers are configured in DTE mode the RIIO and DCDIO input signals are ignored.

Note: DTRI, DSRO and RIIO are also used for call control purpose. This function may overload the normal functions of these signals in certain situations. See section 3.4.4 for details.

The DECT protocol transmits modem lead signals such that only changes of these signals are signalled. When the module detects a change at any of its modem lead inputs it will transmit a dedicated message to its peer.

The maximum transmission rate is one message every 10ms. Hence at the receiving side the lines are updated in 10ms intervals. This effect causes certain changes to the signal timing. Moreover, due to internal pipelining the timing between data bytes sent over the RS-232 interface and modem lead signal changes is not preserved. This must be taken into account in certain applications.

3.4.4 Call control

Call control uses the modem lead signals DTRI, DSRO and RIIO. The call control function is multiplexed with the regular usage of these signals.

An outgoing call is a call that originates from the PT.

An incoming call is a call that originates from the FT.

Please note the command <SPECC>.

3.4.4.1 Outgoing call, PT interface

In order to request a call, the PT-side host shall activate the DTRI signal (i.e. pull it to low level).

An established call is indicated to the host through an activation of the DSRO signal. The DSRO signal remains active for at least 10ms.

There might be several reasons why a call request may not be accepted by the peer:

- Busy condition
- Out of coverage range
- Invalid subscription
- Application-specific reasons

The interface does not provide information about the actual reason.

If DSRO remains deactivate the host may continue the call request by retaining DTRI active.

The host may cancel a call request by deactivating DTRI before DSRO has become active.

3.4.4.2 Outgoing call, FT interface

To accept outgoing calls the FT-side host shall leave the DTRI signal permanently active. In this state the FT accepts any outgoing call immediately.

The host shall reject the call by retaining DTRI deactivated.

3.4.4.3 Incoming call, PT interface

A call request from a FT is signalled to the PT-side host by an activation of the DSRO signal. If the PT is in DCE mode, the call request is also signalled by an activation of the RIIO output signal (see SPECC command).

The host shall accept the call by activating the DTRI signal for at least 10ms. As soon as the call is accepted, the RIIO output signal is deactivated for at least 10ms (DCE mode only, see SPECC command).

3.4.4.4 Incoming call, FT interface

For incoming calls please use config mode commands SPDSI and SPDSD. If there are both entries with SPDSI and SPDSD, the SPDSI entry is used.

On the activation of DTRI or after reset (dependent of DTRI and SPCC) to data mode the RFP establishes a connection to the PT. At this time DTRI of the PT must be inactive, in order to prevent a concurrent connection establishment initiated by the PT.

When the connection has been established DSRO of the PT goes to active state and the host must respond by activating DTRI.

Example for calling PTs with SPDSI with FT as active part:

Step		**Action**
1	FT	enter config mode with '+-+'
2	FT	SPDSI EMC, DNR
3	FT	EXIT, DTRI is active
4	PT	DTRI is inactive
5	PT	when DSRO goes active activate DTRI
6	FT	when DSRO goes active connection is established
7	FT/PT	transmit data
8	FT PT	go to step 1 for next connection when DSRO goes inactive deactivate DTRI
9	FT PT	after last PT deactivate DTRI when DSRO goes inactive deactivate DTRI

3.4.4.5 Call release, PT interface

The PT-side host shall initiate a call release by pulling DTRI inactive for at least 5 seconds. If after that time also the DSRO signal from the HW 86010/20 is inactive the call has been released.

The HW 86010/20 shall indicate a call release from the FT or the network to its host by deactivating DSRO for at least 5 seconds. After this time has expired the host must deactivate DTRI during the following second unless a new call shall requested.

3.4.4.6 Call release, FT interface

The FT-side host shall initiate a call release by pulling DTRI inactive for at least 5 seconds. If after that time also the DSRO signal from the HW 86010/20 is inactive the call has been released.

The HW 86010/20 shall indicate a call release from the PT or the network to its host by deactivating DSRO for at least 5 seconds. The host may retain DTRI activated, while waiting for new calls.

3.4.5 Data transmission

This section describes the operational rules for data transmission in transparent data mode.

3.4.5.1 Alignment with call control

Any data received from the host through the RS-232 interface while there is no call established will be discarded by the module.

When requesting a call the host must wait until the call is established before starting sending data. Otherwise this data may be lost.

When a call has been established, data is transferred in full-duplex mode between the hosts through the RS-232 interface.

Any data sent by the host to the module through the RS-232 interface after the call has been released is discarded by the module

3.4.5.2 Usage of modem lead signals

A host may use the modem lead signals to control and monitor equipment connected to the RS-232 interface of the peer.

When using modem lead signals attention must be paid to the influence of interworking (see section 3.4.3).

A host must not pull DTRI inactive for more than 4 seconds for any other purpose than releasing the call. However the host may pull DTRI inactive for shorter intervals, e.g. to control a modem connected to the RS-232 interface of the peer.

3.4.5.3 Escaping to configuration mode

The configuration mode may be entered from transparent data mode by sending the escape sequence +-+ to the module. If the module detects the escape sequence while a call is established, it will immediately release the call.

The following timing requirements apply:

- Before the first + character and after the last + character there must be a pause of at least 200ms
- Between two characters of the escape sequence the maximum allowed pause is 500ms.

3.4.6 Example: Transparent multipoint mode

In a configuration with 1 FT and 4 PTs the command SPMP ON on FT-side leads to the transparent multipoint mode (see SPMP command):
FT to PT: FT sends 26-byte data packets to each connected PT.
PT to FT: Each PT sends 26-byte data packets to FT.
FT sends these packets first-in-first-out over its RS-232 interface.
In this mode there is no flow control.

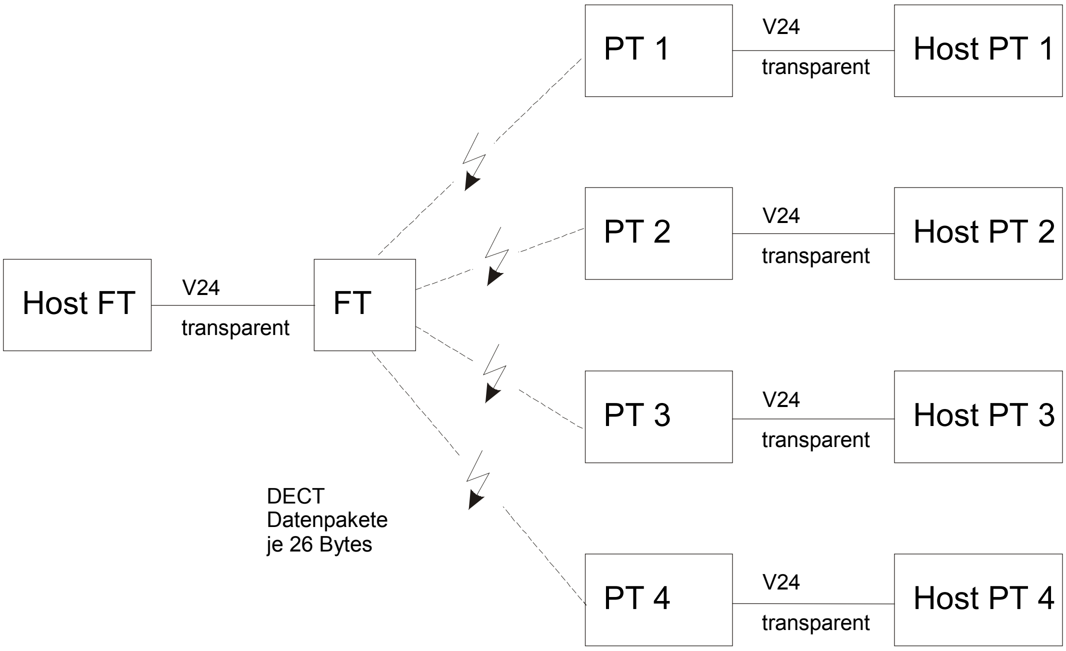

Host FT
V24
transparent
FT
DECT
Datenpakete
je 26 Bytes
PT 1
V24
transparent
Host PT 1
PT 2
V24
transparent
Host PT 2
PT 3
V24
transparent
Host PT 3
PT 4
V24
transparent
Host PT 4

Figure 2: Transparent multipoint mode

3.5 Protocol data mode

The protocol data mode is a sub-mode of the data mode. It allows multiplexed data transmission using the RS-232 interface. The protocol data mode is selected by issuing the configuration command SPPR ON.

Only the single bearer FT may operate in protocol data mode.

The protocol data mode allows the simultaneous handling of multiple connections at the FT. Moreover it provides more flexibility than the transparent data mode.

Call control is handled by software through a dedicated command channel.

In order to use the protocol mode the host must format its data according to a specific protocol which is described below.

3.5.1 General description

The HW 86010/20 implements data transmission according to the DECT standard DSP C.2 (EN 300 651). This standard uses the LU3 protocol, which consists in an error-protected frame relay (LU2) together with a LAP (link access protocol).

LU2 provides an asynchronous frame-oriented data service.

The purposes of LAP are end-to-end link control, error correction and flow control. LAP provides an asynchronous, stream-oriented interface to the upper layer. Data is transmitted as an asynchronous sequence of frames.

In transparent data mode both layers of LU3, i.e. LU2 and LAP, are executed on the module. The host interfaces to the upper interface of LAP. This is illustrated in Figure 3.

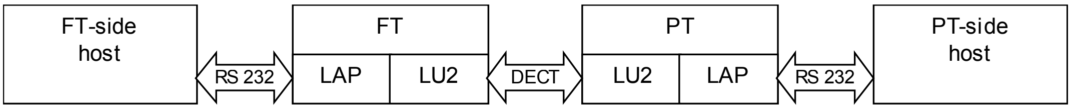


Figure 3: Distribution of LU3 in transparent data mode

In protocol data mode the FT may serve multiple connections. Each of them requires end-to-end link control, error correction and flow control. Therefore in protocol mode the LU3 layer is distributed between host and module. LAP is executed on the host and LU2 on the module. This is illustrated in Figure 4. Please observe that in this example PT1 to PT4 operate in transparent data mode.

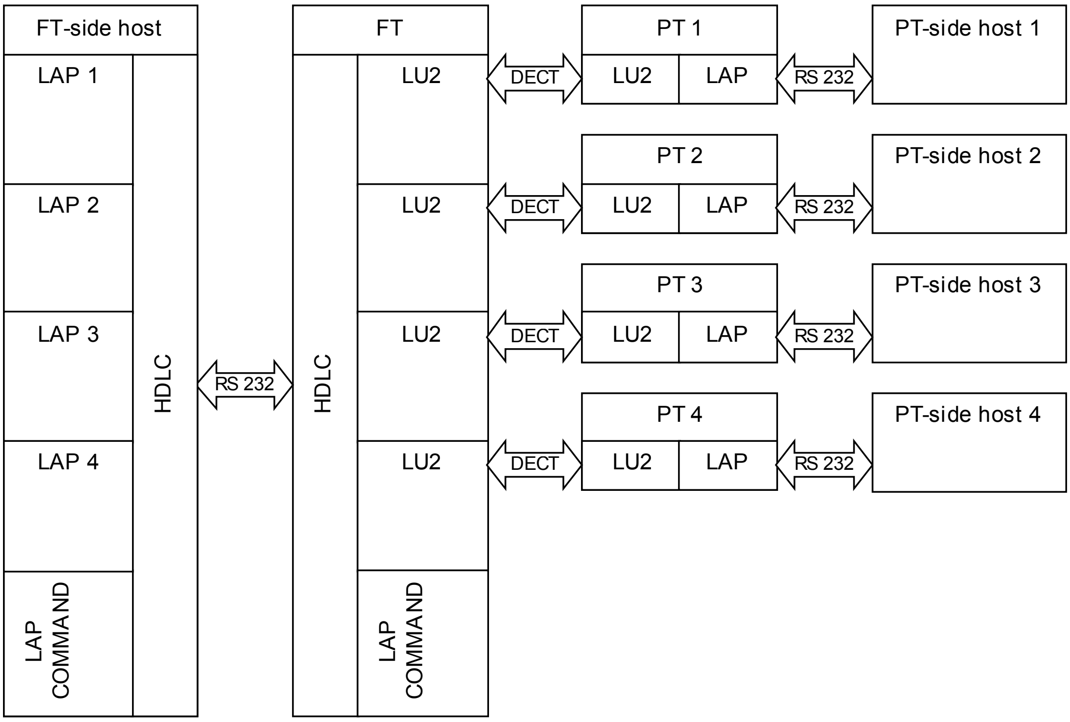

FT-side host
LAP 1
LAP 2
LAP 3
LAP 4
LAP COMMAND
HDLC
RS 232
FT
HDLC
LU2
LU2
LU2
LU2
LAP COMMAND
DECT
DECT
DECT
DECT
PT 1
PT 2
PT 3
PT 4
LU2
LAP
LU2
LAP
LU2
LAP
LU2
LAP
RS 232
RS 232
RS 232
RS 232
PT-side host 1
PT-side host 2
PT-side host 3
PT-side host 4

Figure 4: Distribution of LU3 in protocol data mode

For the transport of LAP frames over the RS-232 interface a HDLC style framing is used. The HDLC layer also serves the purpose of multiplexing and de-multiplexing LAP channels.

The protocol data mode includes a command channel. It is used for call control and provides mechanisms for future extended control functions.

3.5.2 Usage of RS-232 interface

3.5.2.1 Connection of the interface

In protocol data mode only the interface signals TXDI and RXDO are used.

The modem lead signals DTRI, DSRO, DCDIO, and RIIO are not used. However the status of the lead signals can be signalled over the protocol data channel.

3.5.2.2 Interface parameters

The baud rate of the RS-232 interface is selected using the SPBD configuration command. The actual baud rate can be retrieved with the GPBD command. A list of available baud rates is shown in response to the IPBD command.

The baud rate setting is a local matter, i.e. the two peers of a connection may use different baud rates at their ends.

In protocol data mode a baud rate of 115.200 Bd is generally recommended.

The SPEQ and GPEQ commands are not relevant in protocol data mode.

3.5.3 HDLC frame structure

A byte oriented (asynchronous) HDLC framing is implemented.

HDLC frame					
FLAG	ADDR	CTRL	DATA	FCS	FLAG

A HDLC frame consists in a flag field (start byte), a variable length address, a control field, payload data and a 16-bit frame check sequence.

3.5.3.1 Flag field (FLAG)

The frame is started by a 1-byte wide flag field. The flag field has the value 0x7E. All other bytes of a frame must be different from that value. This is achieved through a transparency algorithm (see section 3.5.4.2)

The frame is terminated by another flag field. In a continuous sequence of frames only one flag byte is required as delimiter between frames. The flag field is also used as inter-frame padding. So there may be multiple flag fields between two frames.

3.5.3.2 Address field (ADDR)

The address field of the HDLC frame serves the purpose of identifying the LAP channel for that frame.

HDLC defines a variable-length address mechanism: The LSB (bit 0) of the last address byte is 1, the LSB of all other address bytes is 0. Hence the receiver is able to scan all address bytes until it detects a byte with the LSB set.

A) Data channel address field

For data channels, a 2-byte wide address field is used.

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADDR1	Call Handle							0
ADDR2	NLF	0	0	M	SAPI		C/R	1

Call Handle: This 7-bit wide binary field is a unique identifier of a connection. Call handles are allocated by the HW 86010/20 during call establishment and are used throughout the call for addressing a specific connection.

SAPI: This 2-bit wide field contains higher layer information. It identifies the service access point on the upper interface of the LAP protocol. The following values are assigned:

00	User Data
11	Signalling Data (i.e. status of lead lines)

The remaining values are reserved for future use.

All remaining fields of the ADDR2 byte carry information elements of the LAP protocol (see section 3.5.6).

B) Command channel address field

For the command channel a 1-byte wide address field is used.

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADDR	NLF	0			SAPI		C/R	1

PROT: The 3-bit wide Protocol field defines the type of protocol to be used.

PROT	Meaning
0	Call control protocol (see section 3.5.9)
1	Configuration
2	Ethernet
3	Download
4	Debug
5 to 7	Reserved for future use

The other fields of the ADDR byte are information elements of the LAP protocol (see section 3.5.6).

3.5.3.3 Control field (CTRL)

Refer to 3.5.6.

3.5.3.4 Data field (DATA)

The variable length Data field is used to carry a LAP frame. The HW 86010 supports Data field of up to 26 bytes length.

The length of the Data field is not explicitly signalled but derived at the receiver from the frame boundaries detected by flag fields.

3.5.3.5 Frame check sequence (FCS)

The 16-bit wide FCS is a cyclic redundancy checksum. It provides a mechanism for detecting erroneous frames at the receiver by comparing the computed and the received FCS.

The FCS uses the generator polynomial $x^0+x^5+x^{12}+x^{16}$.

The FCS is calculated over the address and data fields. It does not include the flag field.

An efficient software implementation of the FCS algorithm is included in document RFC 1662.

3.5.4 HDLC procedures

3.5.4.1 Multiplexing of LAP channels

Multiple channels, i.e. data channels and a command channel, are multiplexed on the RS-232 interface. Separate instances of LAP are required for each channel. The address field in the HDLC frame identifies the channel and the associated LAP instance.

A) Transmitting side

The LAP frames are processed by the HDLC layer in the sequence of their arrival.

B) Receiving side

The FCS is checked. Frames with bad FCS are discarded.

The HDLC layer then sends the frame to the LAP associated with the address field of the frame.

3.5.4.2 Transparency

The byte value 0x7E is reserved for the flag field of the HDLC frame (see section 3.5.2). If any of the address, data or FCS fields contains a byte with value 0x7E, it must be removed prior to transmission in order to avoid misinterpretation as flag field by the receiver.

The following transparency algorithm is used:

A) Transmitting side

A complete HDLC frame is assembled (including FCS).

The byte sequence between the two flag fields is checked for any occurrence of values 0x7D or 0x7E. Any byte 0x7D is replaced with the 2-byte sequence 0x7D 0x5D. Any byte 0x7E is replaced with the 2-byte sequence 0x7D 0x5E.

The resulting byte sequence is transmitted. Depending on the data content of the frame, the frame length has been enlarged by the transparency algorithm.

B) Receiving side

The received byte sequence is scanned for the first occurrence of a flag field (0x7E). This marks the start of the frame.

The following bytes are assembled into the HDLC frame. Any byte sequence 0x7D 0x5D results in a single byte 0x7D to be assembled into the HDLC frame. Any byte sequence 0x7D 0x5E results in the single byte 0x7E to be assembled into the HDLC frame.

The next occurrence of a flag field in the received byte sequence marks the end of the frame. It may also indicate the start of a following frame.

The completely received HDLC frame is then further processed (e.g. the FCS is verified).

3.5.5 LAP protocol overview

LAP (link access protocol) is a widely spread protocol for safe data transmission. It provides end-to-end error correction and flow control for a data link. Each data link requires its own instance of LAP.

There exist multiple variants of LAP which differ in some details of their frame structures and their ways to establish a connection. E.g. the LAP variant LAP-D is used in the ISDN D-channel. ITU recommendation Q.921 includes a detailed description.

The LAP variant used in LU3 data transmission is more precisely referred to as LAP-U. It is described in EN 300 651. The firmware uses LAP-U over the air interface.

The LAP protocol used in protocol mode is very similar to LAP-U. The only differences result from embedding the LAP packet into a HDLC frame. This implies using the HDLC ADDR field for transport of LAP information elements and using the HDLC FCS, whereas LAP-U uses a different type of checksum.

LAP is a peer-to-peer protocol. The peers correspond by exchanging LAP frames. These frames may have variable length.

The LAP protocol is full duplex. The two directions are independent, therefor we only regard one direction for this description. Due to this approach we distinguish in this text between sender and receiver. The sender transmits information to the receiver. The receiver returns acknowledgements to the sender. Each LAP instance comprises a sender and a receiver.

LAP support multiple frame operation. The sender may send k I-frames, before it needs an acknowledgement from the receiver. The firmware uses k=4.

LAP provides an efficient flow control mechanism using supervisory frames.

3.5.6 LAP information elements

A LAP frame consists in a control byte and an optional information field.

LAP frame	
CONTR	Information

There are different frame types, information frames (I-frames) and supervisory frames (RR-, RNR-, REJ-, SABM- and UA-frames).

CONTR: The 1-Byte wide control field defines the frame type and, depending on the frame type, other information element. The detailed definition of its structure is shown below.

Frame Type	b7	b6	b5	b4	b3	b2	b1	b0
I Command	N(R)			P	N(S)			0
RR Command / Response	N(R)			P/F	0	0	0	1
RNR Command / Response	N(R)			P/F	0	1	0	1
REJ Command / Response	N(R)			P/F	1	0	0	1
SABM Command	0	0	1	P	1	1	1	1
UA Response	0	1	1	F	0	0	1	1

N(S): The 3-bit send sequence number is used by the sender to identify an I-frame.

N(R): The 3-bit receive sequence number is used by the receiver to acknowledge I-frames.

P: The poll bit is set by the sender in order to request flow control information from the receiver.

F: The final bit is set by the receiver in response to a command with P bit set.

Information: This variable length field (0 to 26 bytes) contains higher layer information to be transported through a LAP channel. It is only present in I-frames.

3.5.6.1 Information frames

When data is transmitted over LAP, it is segmented into I-frames. Each I-frame is identified by a sequence number. This number is incremented for every new I-frame. When an I-frame is repeated, e.g. due to a detected transmission error, it retains its original sequence number. The sequence number preserves integrity of the sequence of I-frames.

Sequence numbers may have values 0 to 7. All operations on sequence numbers are modulo 8. E.g. if the sequence number had value 7, incrementing will result in value 0.

Each I-frame carries also an acknowledgement for the reverse direction. So the information flow in one direction is multiplexed with the acknowledgement flow in the reverse direction.

3.5.6.2 Supervisory frames RR, RNR and REJ

The receiver informs the transmitter about its conditions (see section 3.5.7.2) by using RR-, RNR- and REJ-frames.

A RR-frame indicates that the self busy condition is cleared.

A RNR-frame indicates that the self busy condition is set.

A REJ-frame indicates that the reject exception condition changes from cleared to set. It also indicates that the self busy condition is cleared.

3.5.6.3 Supervisory frames SABM and UA

The SABM command is used for the sole purpose of link re-establishment in certain error recovery situations.

The UA response is used for the sole purpose of responding to a SABM command.

3.5.6.4 Information elements in the ADDR field

The following LAP information elements are not carried in the LAP frame but in the ADDR field of the associated HDLC frame:

NLF: The new link flag indicates a new link. It is set for SABM commands and UA responses.

M: The more bit indicates that a LAP frame is split on multiple HDLC frames and that another segment follows. HW 86010 does not support LAP frame split. The more bit is always 0.

C/R: The command / response bit allows the distinction between command and response frames. The logic of this flag depends the direction of the communication:

Direction	C/R bit	Frame type
Module -> Host	0	Command
Module -> Host	1	Response
Host -> Module	0	Response
Host -> Module	1	Command

3.5.7 LAP procedures

This section describes the mechanisms behind and the operational procedures of the LAP protocol. The purpose is to provide additional illustration to the detailed protocol implementation as described in section 3.5.8.

3.5.7.1 States

The LAP state machine is made-up by the following states:

State	Description
Idle	A LAP link has been initiated (i.e. a LAP instance exists) but there has been no request for data transmission from the higher layer
WaitEstablish	The higher layer has requested the first data transmission. The LAP instance establishes a connection to its peer.
Active	The LAP connection has been established. Depending on the conditions described below, data may be transmitted over the LAP link.

3.5.7.2 Conditions

While in active state, the LAP operation is determined by conditions (flags).

The **receiver** is attributed by the following conditions:

Self Busy: The condition is set or cleared locally by the higher layer (flow control). If the condition is set, the receiver will discard any I-frames from its peer. If the condition is cleared the receiver is ready to receive I-frames from its peer.

Reject Exception: The condition is set, when the receiver receives an I-frame with an unexpected sequence number (N(S) sequence error). It is cleared, when the receiver receives an I-frame with correct sequence number. All I-frames with wrong sequence numbers are discarded by the receiver.

Ack Pending: This condition indicates that an acknowledgement is pending, i.e. the receiver has received successfully at least one I-frame which it has not yet acknowledged. The condition is set, when the receiver receives a valid I-frame. It is cleared when all received I-frames have been acknowledged.

The **sender** is attributed by the following conditions:

Peer Busy: This condition indicates that the peer is busy. It is set when a RNR-frame is received. It is cleared when either a RR- or a REJ-frame is received.

Timer Recovery: This condition is set in case of timeout, while the sender is waiting for an acknowledgement of a previously transmitted I-frame. While in Timer Recovery condition the sender requests an acknowledgement by polling the receiver. The condition is cleared, when a response on the polling command has been received.

3.5.7.3 Timers

LAP requires two timers:

- the LAP-establish timer DLU.02 with timeout period of 2.0 seconds
- the retransmission timer DLU.04 with timeout period of 1.0 seconds

Since only one timer is active at a time, an implementation may use a single instance of a timer object to realise both timers.

The following procedures apply to timers:

- When a timer is started, it begins running until it is stopped or it expires.
- When a running timer is again started, this has no effect, i.e. the timer carries on running.
- When a timer is stopped it is automatically reset. When it is then started again, the full timeout period applies.
- Restarting a timer is equivalent to stopping it and immediately starting it again.

When a timer expires it generates an event to the LAP state machine.

3.5.7.4 Sequence variables

While in active state and while both the Peer Busy and the Timer Recovery conditions are cleared, the sender performs multiple frame operation.

LAP labels each I-frame with a sequence number (see section 3.5.6.1). The sequence number is also used in acknowledgements.

The sender maintains two sequence variables V(S) and V(A). The receiver maintains a sequence variable V(R).

- V(S) contains the sequence number of the next frame to be transmitted.
- V(A) contains the sequence number of the next frame to be acknowledged.
- V(R) contains the sequence number of the next frame to be received.

All sequence variables lie within the range 0 to 7. All operations on sequence variables are modulo 8.

The sequence of I-frames can be separated into three sections:

- Frames which have been sent and an acknowledgement was received
- Frames which have been sent but no acknowledgement has bee received yet (pending frames).
- Frames which shall be sent

A sender may transmit $8 \geq k \geq 1$ I-frames before it requires an acknowledgement from the receiver.

The maximum number of pending I-frames k is an implementation constant of the sender. The firmware uses k=4. It is recommended that the LAP implementation on the host also uses k=4.

3.5.7.5 Sender procedures

A) Sending I-frames

When an I-frame is sent, its sequence number (i.e. the value of V(S)) is transmitted in the N(S) field of the control byte. V(S) is incremented directly after the transmission. Timer DLU.04 is started.

B) Receiving acknowledgements

Acknowledgements are contained in the N(R) field, which is included in any received I-, RR-, RNR- or REJ-frame. When an acknowledgement is received it is first checked whether N(R) is valid, i.e. $V(A) \leq N(R) \leq V(S)$ modulo 8. Reception of an invalid N(R) is a severe error which can only be recovered through a link re-establishment. The following procedure applies to valid N(R) only.

If N(R) = V(A) then N(R) does not acknowledge any new frames and nothing happens.

If N(R) > V(A) then N(R) acknowledges all frames with sequence numbers up to but not including N(R). V(A) is set to N(R) and timer DLU.04 is stopped.

If N(R) < V(S) then there are still pending frames left. In this case DLU.04 is started again.

C) Reacting on flow control

A RNR-frame from the receiver indicates to the sender that it must stop transmitting I-frames. The sender sets the peer busy condition and stops transmitting I-frames. However it may still send supervisory frames. Upon entering the peer busy condition the sender starts timer DLU.04 regardless the processing of the acknowledgement.

A RR- or REJ-frame from the receiver indicates to the sender that it may resume sending I-frames. The peer busy condition is cleared. The timer DLU.04 is treated according the processing of the acknowledgement.

D) Repeating I-frames

A REJ-frame from the receiver indicates to the sender that it must repeat all I-frames with sequence numbers from and including the received value N(R). The sender sets its value V(S) to N(R) and repeats the I-frames accordingly.

E) Timer Expiry

When timer DLU.04 expires, the timer recovery condition is set. The sender sends an appropriate supervisory frame (RR- or RNR-frame depending on the self busy condition of its own receiver) with the P bit set. It then restarts timer DLU.04.

While in timer recovery condition the sender does not transmit I-frames, but waits for a supervisory response frame with the F bit set. Any acknowledgement with F bit cleared will only be used to update V(A) but does not effect the timer.

When a supervisory frame with F bit set is received, this clears the timer recovery condition. V(A) is set to N(R). Also V(S) is set to N(R) which may imply repetition of I-frames.

When timer DLU.04 expires again while the sender is already in timer recovery condition, a repetition count variable RC is incremented and the timer is restarted. If RC exceeds N250 (value: 3), this is treated as a severe error situation and the link is re-established.

3.5.7.6 Receiver Procedures

A) Receiving I-frames

When an I-frame is received the following procedure applies:

If the self busy condition is set, the frame is discarded. Otherwise the receiver checks if N(S) is equal to V(R). In this case the frame is accepted and its content is passed to the higher layer. The reject exception condition is cleared and V(R) is incremented.

If N(S) is different from V(R) (N(S) sequence error) the frame is discarded and the reject exception condition is set.

B) Sending acknowledgements

When the last I-frame was accepted the receiver need not acknowledge it immediately. If there is data to be sent in the reverse direction the acknowledgement may be carried in an I-frame. Otherwise the acknowledgement is sent either as RR-response (if the self busy condition is cleared) or as RNR-response (if it is set).

When the last I-frame was discarded due to N(S) sequence error the receiver must acknowledge it immediately. If the reject exception condition is newly set, it sends a REJ-response. If the reject exception condition was already set before, it sends a RR-response.

In any acknowledgement the receiver sends its actual value of V(R), after possible incrementing, in the N(R) field.

C) Setting and clearing the self busy condition

If the higher layer is ready to accept data from the LAP receiver, it clears the self busy condition. The receiver informs the sender by sending a RR-command frame.

If the higher layer is busy, i.e. does not accept data from the LAP receiver, it sets the self busy condition. The receiver informs the sender by sending a RNR-command frame.

D) Responding to polling requests

When the receiver receives a RR- or RNR-command frame with P bit set, it immediately respond to it with a RR-response (if the self busy condition is cleared) or a RNR-response (if it is set) which has the F bit set.

3.5.7.7 Establishment

A LAP link is established under the following conditions:

- The LAP for the command channel is established immediately when the protocol data mode is entered.
- The LAP for a data channel is established when a call has been set-up using call control procedures in the command channel (see section 3.5.9).

A newly created LAP protocol instance starts its operation in the Idle state. V(S), V(A) and V(R) are set to 0, all conditions are cleared.

The host-side LAP starts establishment by sending an I-frame with P-bit set. This frame does not carry any user data.

The module-side LAP answers by sending a RR-frame with F-bit set.

Now both peers are in Active state and may start exchanging I-frames as described in sections 3.5.7.5 and 3.5.7.6.

The details of the link establishment are included in the SDL representation (see section 3.5.8).

3.5.7.8 Termination

The LAP operation for a data channel is terminated implicitly, when the associated call is terminated using the call control procedures in the command channel.

The LAP operation for the command channel is terminated implicitly, when the protocol data mode is quit.

3.5.7.9 Re-establishment

In error recovery situations any of the peer instance may invoke a link re-establishment by sending a SABM-command with the P bit set and the NLF flag set.

It then sets V(S), V(A) and V(R) to 0 and clears all conditions.

Upon reception of the SABM-command the peer sets its V(S), V(A) and V(R) to 0 and clears all conditions. It then responds with an UA-response with the F bit set and the NLF flag set.

If no UA response is received for a SABM command, the originator repeats the SABM command. For details please refer to section 3.5.8.

3.5.8 SDL representation of LAP

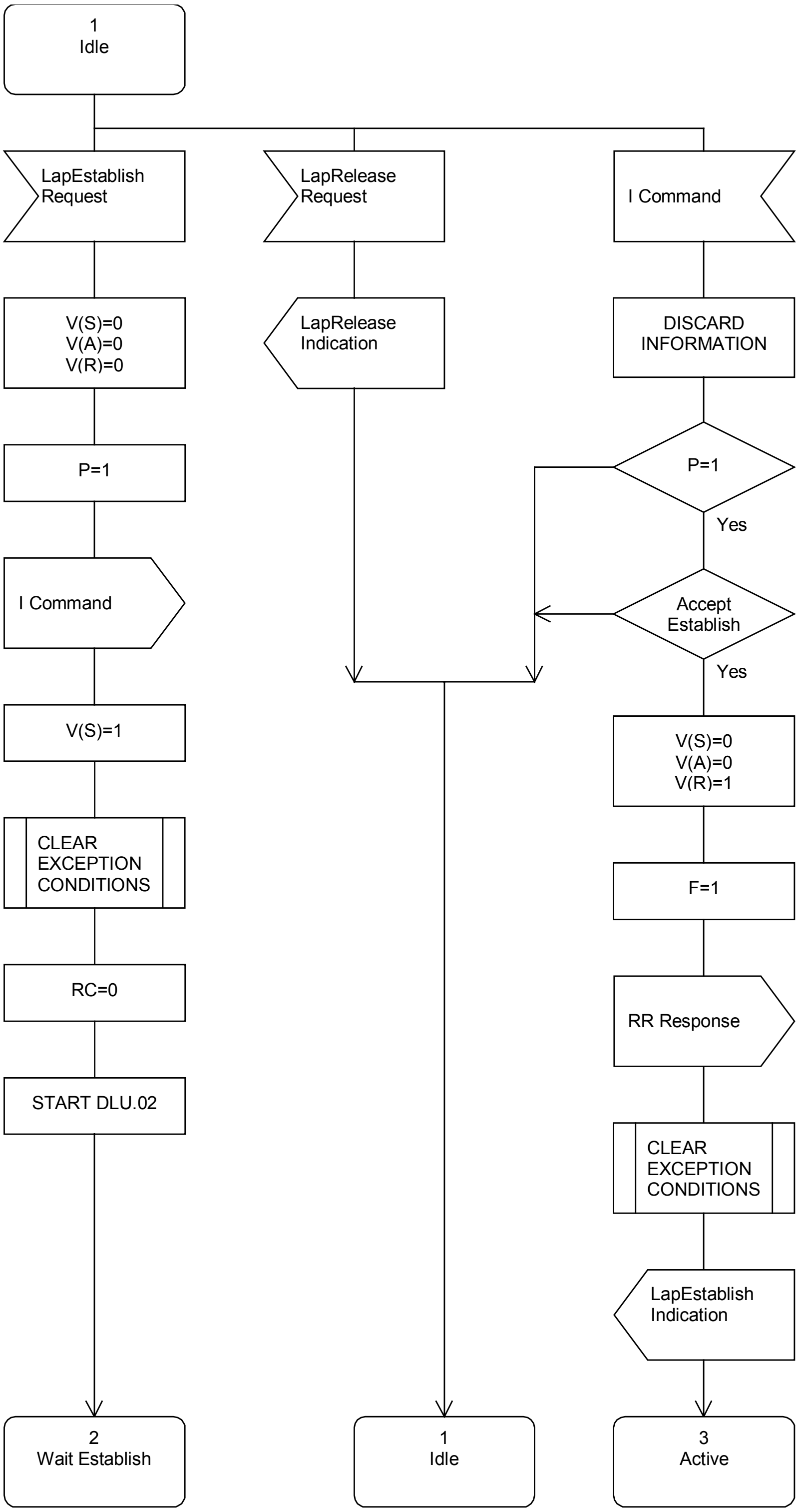

1
Idle
LapEstablish Request
LapRelease Request
I Command
V(S)=0
V(A)=0
V(R)=0
LapRelease Indication
DISCARD INFORMATION
P=1
P=1
Yes
I Command
Accept Establish
Yes
V(S)=1
V(S)=0
V(A)=0
V(R)=1
CLEAR EXCEPTION CONDITIONS
F=1
RC=0
RR Response
START DLU.02
CLEAR EXCEPTION CONDITIONS
LapEstablish Indication
2
Wait Establish
1
Idle
3
Active

Figure 5: SDL representation of LAP, part 1

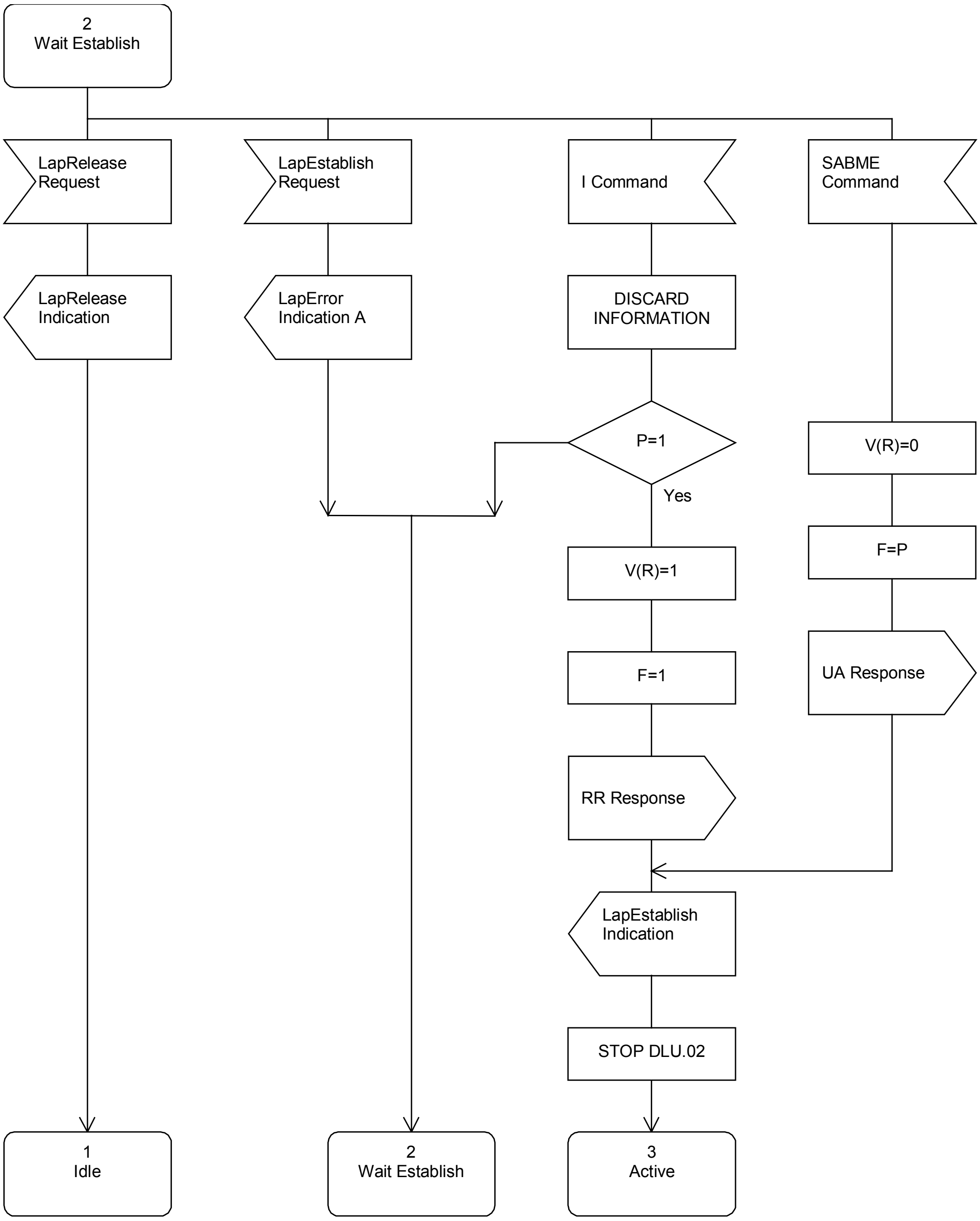

2
Wait Establish
LapRelease Request
LapEstablish Request
I Command
SABME Command
LapRelease Indication
LapError Indication A
DISCARD INFORMATION
P=1
Yes
V(R)=0
V(R)=1
F=P
F=1
UA Response
RR Response
LapEstablish Indication
STOP DLU.02
1
Idle
2
Wait Establish
3
Active

Figure 6: SDL representation of LAP, part 2

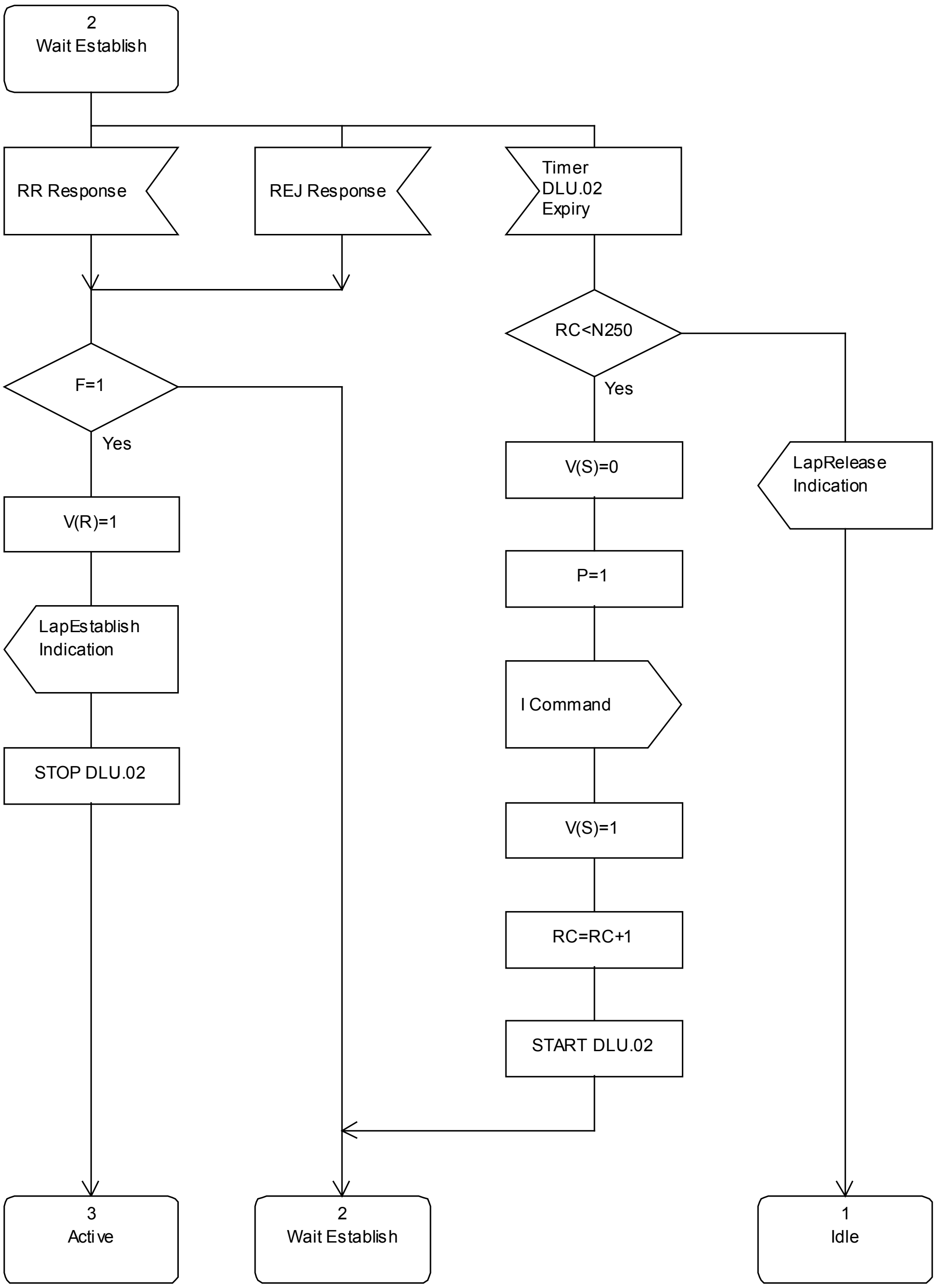

2
Wait Establish
RR Response
REJ Response
Timer
DLU.02
Expiry
RC<N250
Yes
F=1
Yes
V(S)=0
LapRelease
Indication
V(R)=1
P=1
LapEstablish
Indication
I Command
STOP DLU.02
V(S)=1
RC=RC+1
START DLU.02
3
Active
2
Wait Establish
1
Idle

Figure 7: SDL representation of LAP, part 3

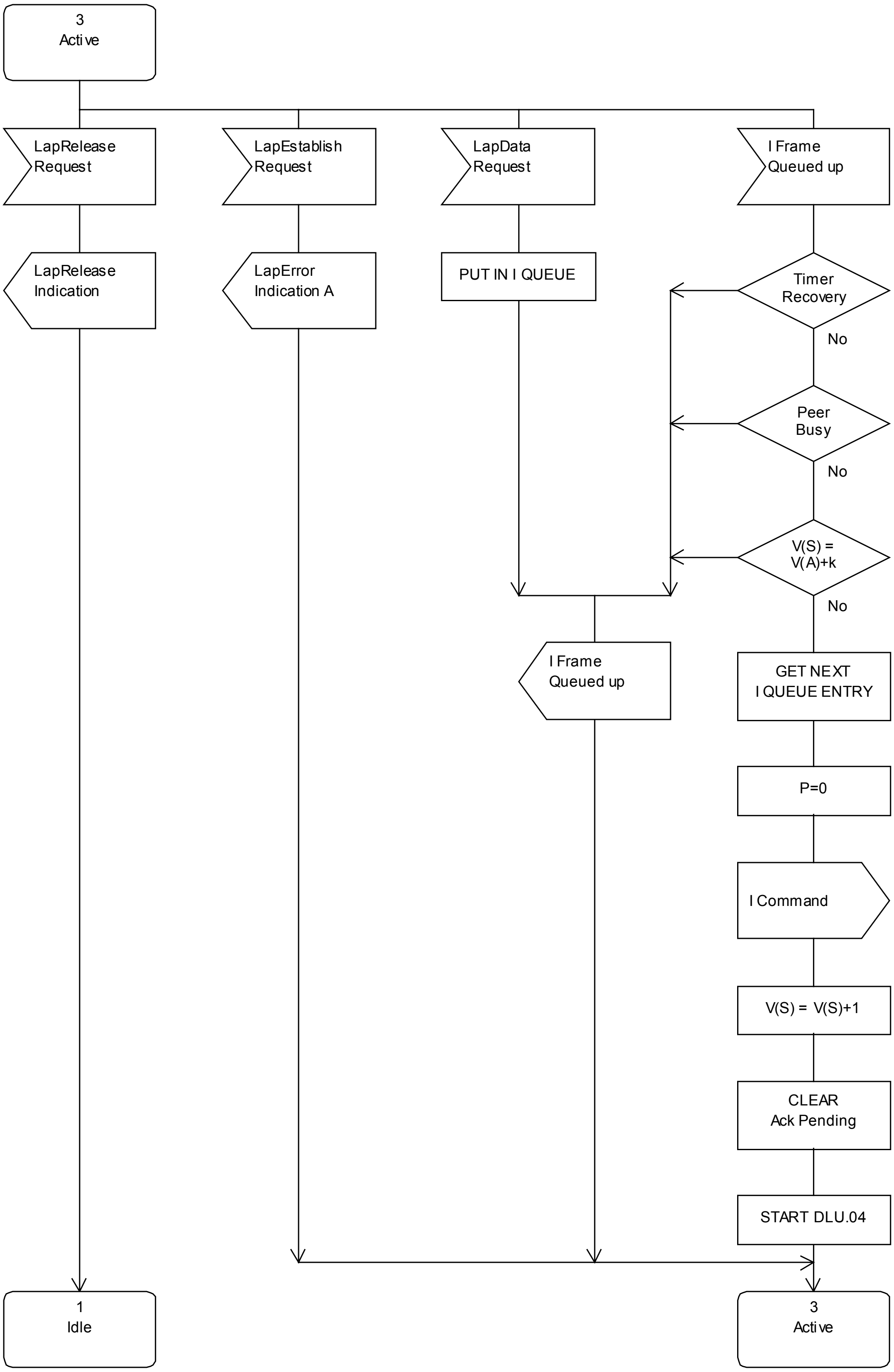

3
Active
LapRelease Request
LapEstablish Request
LapData Request
I Frame Queued up
LapRelease Indication
LapError Indication A
PUT IN I QUEUE
Timer Recovery
No
Peer Busy
No
V(S) = V(A)+k
No
I Frame Queued up
GET NEXT I QUEUE ENTRY
P=0
I Command
V(S) = V(S)+1
CLEAR Ack Pending
START DLU.04
1
Idle
3
Active

Figure 8: SDL representation of LAP, part 4

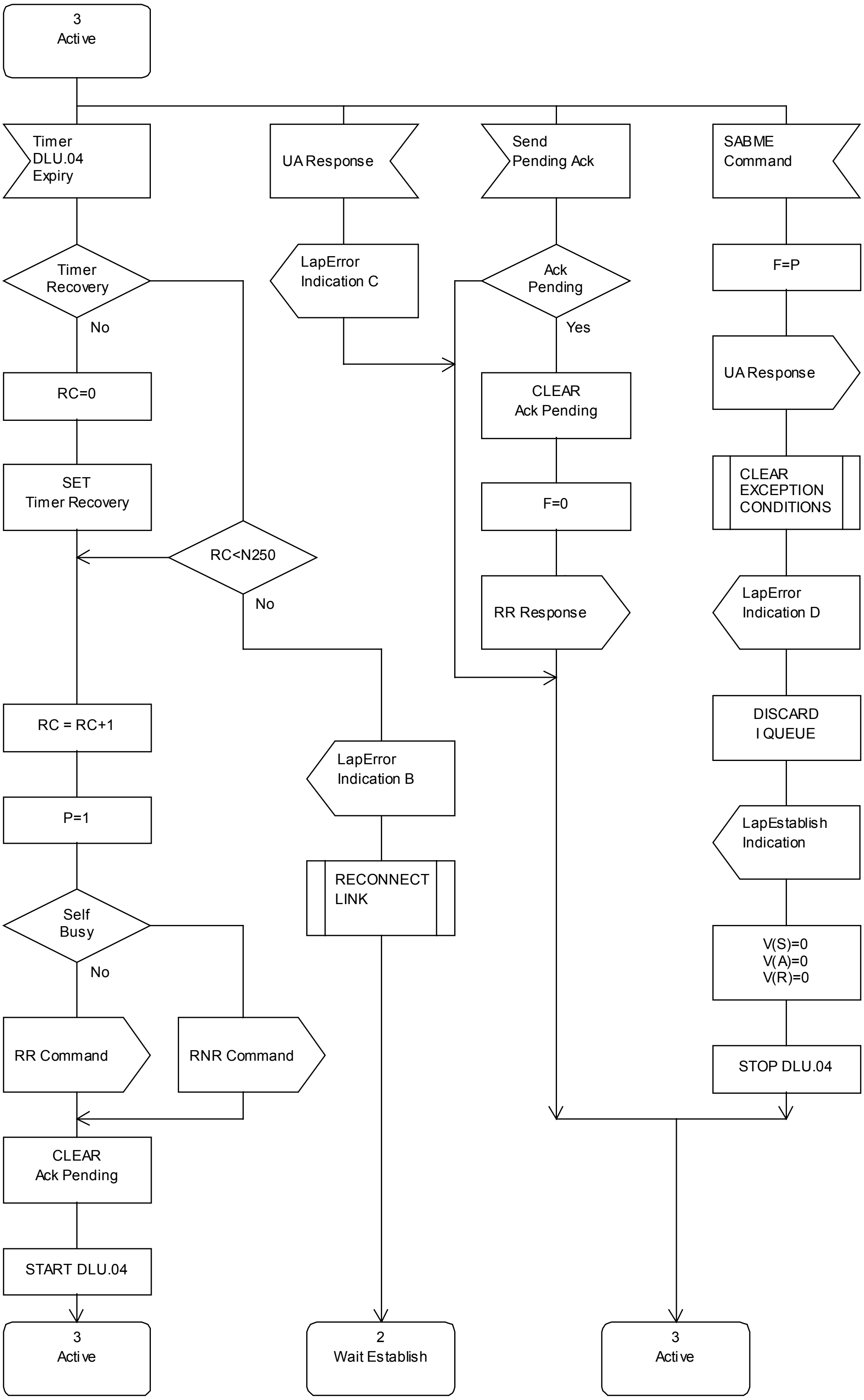

3
Active
Timer DLU.04 Expiry
UA Response
Send Pending Ack
SABME Command
Timer Recovery
No
LapError Indication C
Ack Pending
Yes
F=P
RC=0
UA Response
CLEAR Ack Pending
SET Timer Recovery
CLEAR EXCEPTION CONDITIONS
F=0
RC<N250
No
RR Response
LapError Indication D
RC = RC+1
LapError Indication B
DISCARD I QUEUE
P=1
RECONNECT LINK
LapEstablish Indication
Self Busy
No
V(S)=0
V(A)=0
V(R)=0
RR Command
RNR Command
STOP DLU.04
CLEAR Ack Pending
START DLU.04
3
Active
2
Wait Establish
3
Active

Figure 9: SDL representation of LAP, part 5

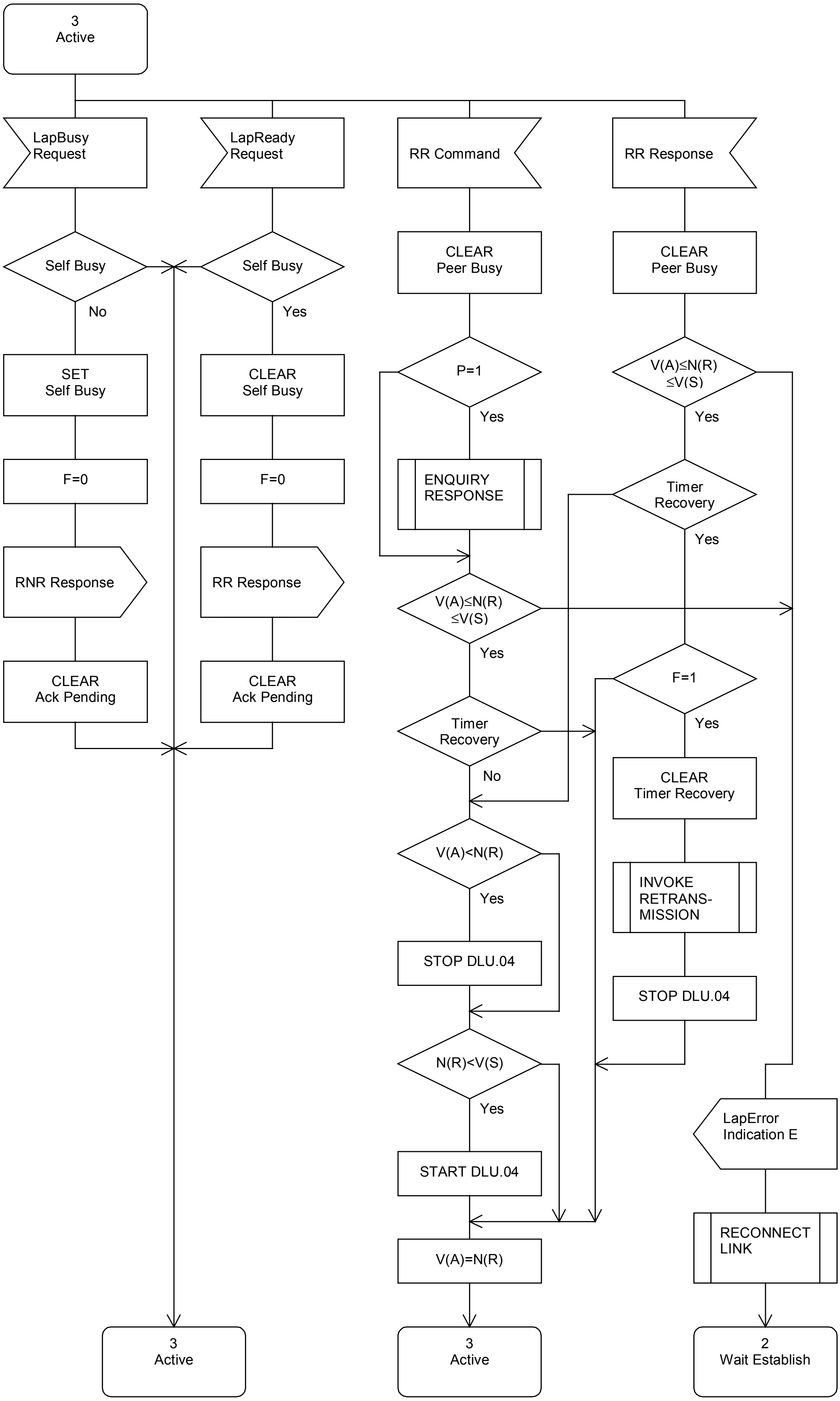

3
Active
LapBusy Request
LapReady Request
RR Command
RR Response
Self Busy
Self Busy
CLEAR Peer Busy
CLEAR Peer Busy
No
Yes
SET Self Busy
CLEAR Self Busy
P=1
V(A)≤N(R) ≤V(S)
Yes
Yes
F=0
F=0
ENQUIRY RESPONSE
Timer Recovery
Yes
RNR Response
RR Response
V(A)≤N(R) ≤V(S)
Yes
F=1
CLEAR Ack Pending
CLEAR Ack Pending
Timer Recovery
Yes
No
CLEAR Timer Recovery
V(A)<N(R)
Yes
INVOKE RETRANS-MISSION
STOP DLU.04
STOP DLU.04
N(R)<V(S)
Yes
LapError Indication E
START DLU.04
RECONNECT LINK
V(A)=N(R)
3
Active
3
Active
2
Wait Establish

Figure 10: SDL representation of LAP, part 6

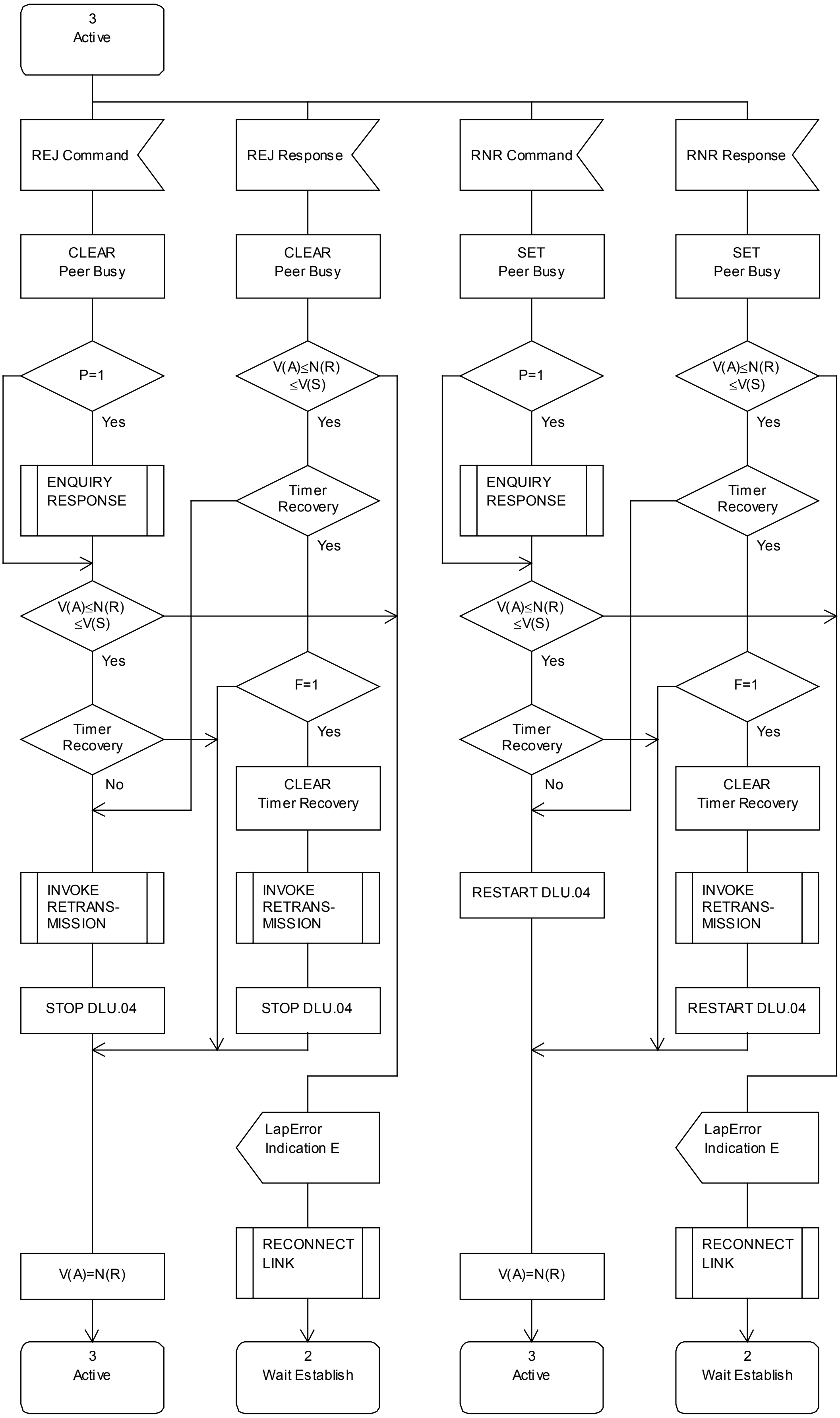

3
Active
REJ Command
REJ Response
RNR Command
RNR Response
CLEAR
Peer Busy
CLEAR
Peer Busy
SET
Peer Busy
SET
Peer Busy
P=1
V(A)≤N(R)
≤V(S)
P=1
V(A)≤N(R)
≤V(S)
Yes
ENQUIRY
RESPONSE
Timer
Recovery
ENQUIRY
RESPONSE
Timer
Recovery
V(A)≤N(R)
≤V(S)
V(A)≤N(R)
≤V(S)
F=1
F=1
Timer
Recovery
Timer
Recovery
No
CLEAR
Timer Recovery
CLEAR
Timer Recovery
INVOKE
RETRANS-
MISSION
RESTART DLU.04
STOP DLU.04
RESTART DLU.04
LapError
Indication E
RECONNECT
LINK
V(A)=N(R)
3
Active
2
Wait Establish

Figure 11: SDL representation of LAP, part 7

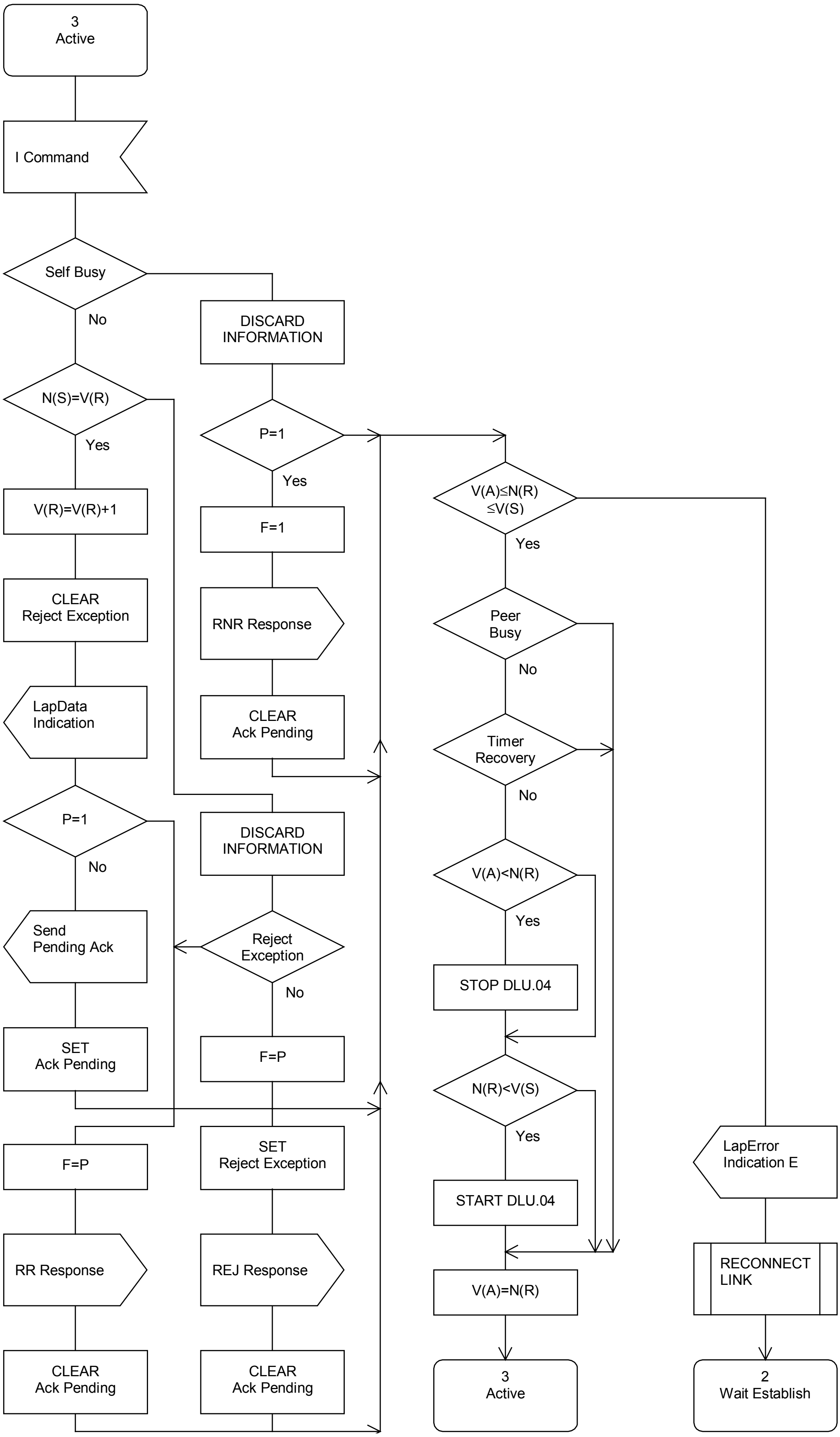

3
Active
I Command
Self Busy
No
DISCARD
INFORMATION
N(S)=V(R)
Yes
P=1
Yes
V(R)=V(R)+1
F=1
V(A)≤N(R)
≤V(S)
Yes
CLEAR
Reject Exception
RNR Response
Peer
Busy
No
LapData
Indication
CLEAR
Ack Pending
Timer
Recovery
No
P=1
No
DISCARD
INFORMATION
V(A)<N(R)
Yes
Send
Pending Ack
Reject
Exception
No
STOP DLU.04
SET
Ack Pending
F=P
N(R)<V(S)
Yes
F=P
SET
Reject Exception
LapError
Indication E
START DLU.04
RR Response
REJ Response
RECONNECT
LINK
V(A)=N(R)
CLEAR
Ack Pending
CLEAR
Ack Pending
3
Active
2
Wait Establish

Figure 12: SDL representation of LAP, part 8

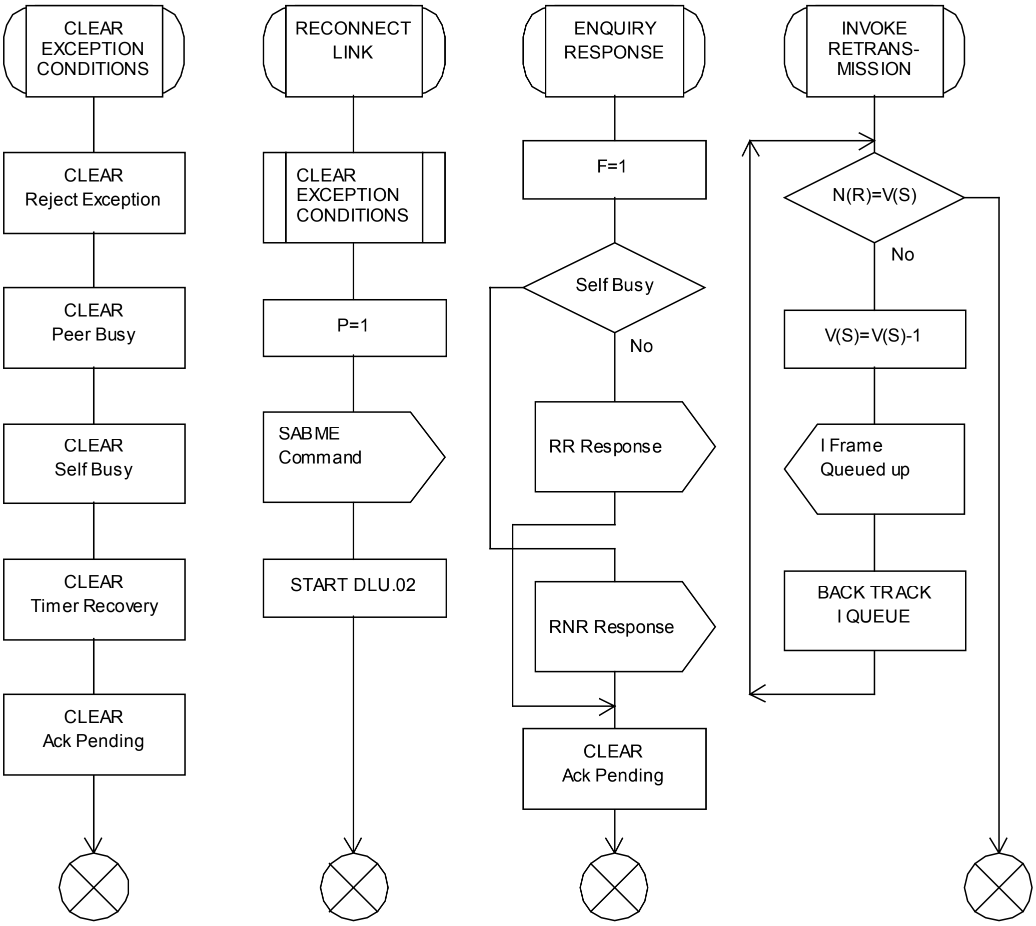

CLEAR EXCEPTION CONDITIONS
CLEAR Reject Exception
CLEAR Peer Busy
CLEAR Self Busy
CLEAR Timer Recovery
CLEAR Ack Pending
RECONNECT LINK
CLEAR EXCEPTION CONDITIONS
P=1
SABME Command
START DLU.02
ENQUIRY RESPONSE
F=1
Self Busy
No
RR Response
RNR Response
CLEAR Ack Pending
INVOKE RETRANS-MISSION
N(R)=V(S)
No
V(S)=V(S)-1
I Frame Queued up
BACK TRACK I QUEUE

Figure 13: SDL representation of LAP, part 9

3.5.9 Call control information elements

3.5.9.1 General description

The call control protocol is an application in the command channel. It is identified by the value prot=0 in the HDLC ADDR byte (see section 3.5.3.2).

Through the call control protocol host and module exchange all necessary signalling information for establishing and releasing calls.

Establishing a call implies setting-up a DECT connection, an associated LAP protocol instance and binding both to a free HDLC data channel. This is referred to as call entity which is identified by a call handle. The call handle is allocated by the module as part of the call establishment procedure.

The call handle is used in the header of each HDLC frame in order to identify the call entity the data belongs to (see section 3.5.3.2).

Releasing a call implies closing-down the associated DECT connection and the LAP protocol instance. When a call is released the call handle is freed.

Each call control command is sent in the information field (see section 3.5.6) of an I-frame.

3.5.9.2 ConnectInd command

Direction: Module -> Host

Inform the host that the HW 86010/20 has established a call to a PT with the indicated IPUI.

The information field has a size of 13 bytes. It is structured as follows:

LAP Information field					
CMD	Handle	IPUI-Type	IPUI-Len	IPUI-Data	Call-Nr

CMD: This 1-byte data field identifies the type of call control command. For a ConnectInd command CMD = 0x01.

Handle: Bit 0 to 6 of this 1-byte data field contain the call handle of the newly established call. Bit 7 is always 0.

IPUI-Type: This 1-byte data field indicates the type of IPUI-Data. The value 0x00 indicates IPUI type N (see EN 300 175 part 6). This is the only IPUI type supported by the firmware. All other values are reserved for future use.

IPUI-Len: This 1-byte data field contains the number of valid bits in the IPUI-Data field. In the current implementation always set to 0x28=40.

IPUI-Data: This 8-bytes data field contains the IPUI. Bit 7 of the first byte is the first bit of the IPUI. Unused bits are set to 0.

Call-Nr: Bit 0 to 6 of this 1-byte data field contain the call number of PT. Bit 7 = 1 means a new call, bit 7 = 0 means a recall after having lost the connection.

3.5.9.3 DisconnectInd command

Direction: Module -> Host

Inform the host that the HW 86010/20 has released the call to a PT with the indicated IPUI.

The information field has a size of 12 bytes. It is structured as follows:

LAP Information field				
CMD	Handle	IPUI-Type	IPUI-Len	IPUI-Data

CMD: This 1-byte data field identifies the type of call control command. For a DisconnectInd command CMD = 0x02.

Handle: Bit 0 to 6 of this 1-byte data field contain the call handle of the released call. Bit 7 is always 0.

IPUI-Type: This 1-byte data field indicates the type of IPUI-Data. The value 0x00 indicates IPUI type N (see EN 300 175 part 6). This is the only IPUI type supported by the firmware. All other values are reserved for future use.

IPUI-Len: This 1-byte data field contains the number of valid bits in the IPUI-Data field. In the current implementation always set to 0x28=40.

IPUI-Data: This 8-bytes data field contains the IPUI. Bit 7 of the first byte is the first bit of the IPUI. Unused bits are set to 0.

3.5.9.4 ConnectReq command

Direction: Host -> Module

Requests the HW 86010/20 to establish a call to a PT with the indicated IPUI.

The information field has a size of 12 bytes. It is structured as follows:

LAP Information field				
CMD	Handle	IPUI-Type	IPUI-Len	IPUI-Data

CMD: This 1-byte data field identifies the type of call control command. For a ConnectReq command CMD = 0x03.

Handle: This 1-byte field is always set to 0. The call handle is assigned by the HW 86010.

IPUI-Type: This 1-byte data field indicates the type of IPUI-Data. The value 0x00 indicates IPUI type N (see EN 300 175 part 6). This is the only IPUI type supported by the firmware. All other values are reserved for future use.

IPUI-Len: This 1-byte data field contains the number of valid bits in the IPUI-Data field. In the current implementation always set to 0x28=40.

IPUI-Data: This 8-bytes data field contains the IPUI. Bit 7 of the first byte is the first bit of the IPUI. Unused bits are set to 0.

3.5.9.5 DisconnectReq command

Direction: Host -> Module

Requests the HW 86010/20 to release the call with the indicated cll handle

The information field has a size of 2 bytes. It is structured as follows:

LAP Information field	
CMD	Handle

CMD: This 1-byte data field identifies the type of call control command. For a DisconnectReq command CMD = 0x04.

Handle: Bit 0 to 6 of this 1-byte data field contain the call handle of the released call. Bit 7 is always 0.

3.5.9.6 LocationInd command

Direction: Module -> Host

Request from FT to host, if a given PT is allowed to synchronise with the FT.

The information field has a size of 12 bytes. It is structured as follows:

LAP Information field				
CMD	Handle	IPUI-Type	IPUI-Len	IPUI-Data

CMD: This 1-byte data field identifies the type of call control command. For a LocationInd command CMD = 0x05.

Handle: The value of this 1-byte data field is always 0x00 (here the call handle is not used).

IPUI-Type: This 1-byte data field indicates the type of IPUI-Data. The value 0x00 indicates IPUI type N (see EN 300 175 part 6). This is the only IPUI type supported by the firmware. All other values are reserved for future use.

IPUI-Len: This 1-byte data field contains the number of valid bits in the IPUI-Data field. In the current implementation always set to 0x28=40.

IPUI-Data: This 8-bytes data field contains the IPUI. Bit 7 of the first byte is the first bit of the IPUI. Unused bits are set to 0.

3.5.9.7 LocationRes command

Direction: Host -> Module

Answer from host to FT concerning LocationInd; result value in parameter Loc-Result:

RESULT_ACCEPT	PT synchronises to FT
RESULT_REJECT	PT puts FT to a temporary blacklist and looks for another FT for synchronisation.

The information field has a size of 13 bytes. It is structured as follows:

LAP Information field					
CMD	Handle	IPUI-Type	IPUI-Len	IPUI-Data	Loc-Result

CMD: This 1-byte data field identifies the type of call control command. For a LocationRes command CMD = 0x06.

Handle: The value of this 1-byte data field is always 0x00 (here the call handle is not used).

IPUI-Type: This 1-byte data field indicates the type of IPUI-Data. The value 0x00 indicates IPUI type N (see EN 300 175 part 6). This is the only IPUI type supported by the firmware. All other values are reserved for future use.

IPUI-Len: This 1-byte data field contains the number of valid bits in the IPUI-Data field. In the current implementation always set to 0x28=40.

IPUI-Data: This 8-bytes data field contains the IPUI. Bit 7 of the first byte is the first bit of the IPUI. Unused bits are set to 0.

Loc-Result: This 1-byte data field contains the result value:

0x00	for RESULT_ACCEPT
0x01	for RESULT_REJECT

3.5.10 Call control procedures

3.5.10.1 Incoming call

An incoming call is always initiated by the host.

The host issues a ConnectReq command, identifying the PT by its IPUI.

The HW 86010/20 (FT) sends a paging message which commands the requested PT to establish a DECT connection. If the PT receives this paging message, it connects to the FT.

When the connection has been established the HW 86010/20 issues a ConnectInd command, and thereby returns a call handle to the host.

Upon reception of the ConnectInd command, the host shall check the IPUI. In case it is different from the requested IPUI, the host may assume that an outgoing call is overlapping the incoming call.

The host may immediately send data using the data channel with that call handle. It shall process all data from that data channel.

Note: This works only after having started LAP.

3.5.10.2 Outgoing call

An outgoing call is always initiated by the PT.

The PT establishes a DECT connection with the FT.

When the connection has been established the HW 86010/20 issues a ConnectInd command, and thereby returns a call handle to the host.

The host may immediately send data using the data channel with that call handle. It shall process all data from that data channel.

Note: This works only after having started LAP.

3.5.10.3 Call release, host initiated

A call is released by the host, when it issues DisconnectReq command.

The HW 86010/20 (FT) immediately releases the call.

After call release, the HW 86010/20 informs the host, by sending a DisconnectInd command. Upon reception of this command the host shall check the value of the received call handle. If different from the call handle in the DisconnectReq command, the host may assume an overlapping PT initiated call release.

By reception of the DisconnectInd command, the call handle is no longer valid and must not be used by the host.

3.5.10.4 Call release, PT initiated

When a PT requests a call release, the HW 86010/20 (FT) immediately releases the call.

After call release, the HW 86010/20 informs the host, by sending a DisconnectInd command.

By reception of the DisconnectInd command, the call handle is no longer valid and must not be used by the host.

3.5.11 API of the dectprot.dll

The dectprot.dll is a library containing functions for using the HW 86010/20 module. In the following the API of this .dll will be explained.

3.5.11.1 DECT_CALLBACK_FUNC_T

Typedef

```
typedef int(*DECT_CALLBACK_FUNC_T)
                    (       int nChannel,
                            DECT_EVENT_T event,
                            void *pArg1,
                            void *pArg2 );
```

Description	Call-back-function for notification of events; for registering use DectRegisterCallback().
Note	

Parameters pArg1 and pArg2 depend from the event-type of the addicted data shown in the table below.

event	parg1	parg2	event description
dectEvNone	NULL	NULL	for internal administration
dectEvDataInd	int = number of bytes	NULL	data available for reading
dectEvConnStatusInd	int = 1 establishment = 0 termination	structure DECT_IPUI_T	connection establishment / - termination
dectEvLineStatusInd	int = line-status-register	NULL	line-status-register (current only DSR)
dectEvLapStateInd	structure DECT_IPUI_T	LAP-state datablock	LAP-state of an aborted connection. Pointer to LAP-state valid only during callback!
dectEvLapStateReq	structure DECT_IPUI_T	int = call-number	connection-reestablishment: requirement of the LAP-state (confirm by using DectLapStateCfm())
dectEvLocationInd	int = modul-number	structure DECT_IPUI_T	location-registration: allow the device to synchronise (reply by using DectLocationRes())

3.5.11.2 DectInit

Function	int DectInit(int nNumPorts, char *apszPortNames[]);	
Description	Channel-layers initialisation and opening of the serial-devices of the DECT-modules.	
Parameter	nNumPorts	Number of the delivered COM-port-names.
	apszPortNames	Pointer to array of char-pointers with serial-device-names (e.g. “COM1”). Zero-pointers in the array will be ignored.
Return	0	Ok
	< 0	Error
Note		

3.5.11.3 DectDestroy

Function	int DectDestroy(void);	
Description	Channel-layers de-initialisation and opening the serial-devices of the DECT-modules.	
Parameter	none	
Return	0	Ok
	< 0	Error
Note		

3.5.11.4 DectRegisterCallback

Function	int DectRegisterCallback(DECT_CALLBACK_FUNC_T pfCallbackFunc);	
Description	Registration of a call-back function. Status changes are signalled by this function.	
Parameter	pfCallbackFunc	Pointer to the function from the type DECT_CALLBACK_FUNC_T. The hand over of NULL will result in deactivating of the call-back.
Return	0	Ok
	< 0	Error
Note	• The possible events result from the enum DECT_EVENT_T.	

3.5.11.5 DectOpen

Function	int DectOpen(int nChannel);	
Description	Open a special channel for activating data-receive.	
Parameter	nChannel	Channel-number
Return	0	Ok
	< 0	Error
Note	• A closed channel will ignore received data.	

3.5.11.6 DectClose

Function	int DectClose(int nChannel);	
Description	Close a special channel for activating data-receive.	
Parameter	nChannel	Channel-number
Return	0	Ok
	< 0	Error
Note	• A closed channel will ignore received data.	

3.5.11.7 DectRead

Function	int DectRead(int nChannel, void *pBuf, int nMaxLen);	
Description	Read the data from a special channel.	
Parameter	nChannel	Channel-number to read from
	pBuf	Pointer to buffer for returning data
	nMaxLen	Max. number of bytes to read
Return	>= 0	Number of actually read bytes
	< 0	Error
Note	• The function directly returns in case of no data for this channel. • See DectReadTo()	

3.5.11.8 DectWrite

Function	int DectWrite(int nChannel, void *pBuf, int nLength);	
Description	Write the data to a special channel.	
Parameter	nChannel	Channel-number to write to
	pBuf	Pointer to buffer holding the data to write
	nLength	Number of bytes to write
Return	>= 0	Number of actually written bytes
	< 0	Error
Note	• To avoid data loss, the returned value should be considered. If required, the remaining data should be written by another function call (e.g. by using DectWriteTo()). If the function returns with less bytes than instructed this can be caused by the flow control. • After returning from this function it is possibly that the buffer is still filled with 4*26 bytes unconfirmed data. • See DectWriteTo(), DectGetTxPending()	

3.5.11.9 DectConnectReq

Function	int DectConnectReq(int nModule, DECT_IPUI_T *pIpui, int nReason);	
Description	Prompt the device to establish connection.	
Parameter	nModule	DECT-module where the connection request shall be send to: 0 Module 1 1 Module 2 -1 all modules
	pIpui	IPUI of the device which shall be called
	nReason	Cause of the request. At this point of time always set to REASON_NORMAL
Return	0	Ok
	< 0	Error
Note	• There will be only a request for a connection establishment to the device. If the device can be reached it registers on a free channel (event dectEvConnStatusInd).	

3.5.11.10 DectDisconnectReq

Function	int DectDisconnectReq (int nChannel, int nReason);	
Description	Prompt the device on a special channel to clear connection. The Confirmation will be returned through an event.	
Parameter	nChannel	Channel-number
	nReason	Cause of the request. At this point of time always set to REASON_NORMAL
Return	0	Ok
	< 0	Error
Note	• There will be only a request for a connection establishment to the device. If the device can be reached it registers on a free channel (event dectEvConnStatusInd).	

3.5.11.11 DectGetConnStatus

Function	int DectGetConnStatus(int nChannel);	
Description	Prompt the device on a special channel for an established connection.	
Parameter	nChannel	Channel-number
Return	>= 0	Connection-state: 0 not connected 1 connected
	< 0	Error
Note		

3.5.11.12 DectGetLineStatus

Function	int DectGetLineStatus(int nChannel);	
Description	Prompt the device on a special channel for the line-state.	
Parameter	nChannel	Channel-number
Return	>= 0	Line-state (bitmapped, at this point of time always set to LINE_STATE_DSR)
	< 0	Error
Note		

3.5.11.13 DectGetIpui

Function	int DectGetIpui(int nChannel, DECT_IPUI_T *pIpui);	
Description	Acquires the IPUI of the connected device on a special channel.	
Parameter	nChannel	Channel-number
	pIpui	Pointer to a structure to return the IPUI
Return	0	Ok
	< 0	Error
Note		

3.5.11.14 DectGetBytesAvail

Function	int DectGetBytesAvail(int nChannel);	
Description	Prompt to get the number of available bytes to be read on a special channel.	
Parameter	nChannel	Channel-number
Return	>= 0	Number of bytes available to be read
	< 0	Error
Note		

3.5.11.15 DectGetTxFree

Function	int DectGetTxFree(int nChannel);	
Description	Prompt to get the number of available free space in the send-buffer.	
Parameter	nChannel	Channel-number
Return	>= 0	Number of free bytes available in the send-buffer
	-1	Error
Note		

3.5.11.16 DectGetTxPending

Function	int DectGetTxPending(int nChannel);	
Description	Prompt to get the number of bytes to be send (unverified data).	
Parameter	nChannel	Channel-number
Return	>= 0	Number of bytes to be send
	-1	Error
Note		

3.5.11.17 DectLapStateCfm

Function	int DectLapStateCfm(int nChannel, unsigned char *pLapState);	
Description	Answer to a lap-state request: reactivate a secure lap-state.	
Parameter	nChannel	Channel-number
	pLapState	Lap-state data-block or NULL when not available
Return	0	Ok
	!= 0	Error
Note		

3.5.11.18 DectLocationRes

Function	int DectLocationRes(int nModule, DECT_IPUI_T *pIpui, int nResult);	
Description	Answer to a location-indication: Is the device allowed to synchronise to the FP or must it search for another FP.	
Parameter	nModule	DECT-module-number: 0 Module 1 1 Module 2
	pIpui	IPUI of the device
	nResult	Connection established? 0 yes 1 no
Return	0	Ok
	!= 0	Error
Note		

3.5.11.19 DectSwitchRoaming

Function	int DectSwitchRoaming(int nChannel, int bOn);	
Description	Activate / deactivate roaming-support for a special channel.	
Parameter	nChannel	Channel-number
	bOn	!= 0 Activate roaming – the events LapStateInd and LapStateReq will be raised 0 Deactivate roaming (connections will always be new established)
Return	0	Ok
	1	Error
Note		

3.5.11.20 DectSwitchLocation

Function	int DectSwitchLocation(int nModule, int bOn);	
Description	Activate / deactivate the location registration.	
Parameter	nModule	DECT-module-number: 0 Module 1 1 Module 2
	bOn	1 Before a synchronisation the PT will send a request if synchronisation is allowed (event LocationInd) 0 Allow synchronisation directly
Return	0	Ok
	1	Error
Note	• Location-indication is related to a module – not a channel. Therefore the notification will be through an event on the first channel of the module (0 or 4).	

3.5.11.21 DectLapStateGetLen

Function	int DectLapStateGetLen(unsigned char *pLapState);	
Description	Acquire the length of a lap-state.	
Parameter	pLapState	Secured lap-state at connection termination
Return	Length of the lap-state in bytes.	
Note		

3.5.11.22 DectLapStateGetIpui

Function	int DectLapStateGetIpui(unsigned char *pLapState, DECT_IPUI_T *pIpui);	
Description	Acquire the IPUI of a lap-state.	
Parameter	pLapState	Secured lap-state at connection termination
	pIpui	Destination-buffer for the IPUI
Return	0	Ok
	-1	Error
Note		

3.5.11.23 DectLapStateGetCallNr

Function	int DectLapStateGetCallNr(unsigned char *pLapState);	
Description	Acquire the connection-number of a lap state.	
Parameter	pLapState	Secured lap-state at connection termination
	pIpui	Destination-buffer for the IPUI
Return	>= 0	Ok
	-1	Error
Note		

3.5.11.24 DectBuildIpuiTypeN

Function	int DectBuildIpuiTypeN(DECT_IPUI_T *pIpui, unsigned int emc, unsigned long dectno);	
Description	Generates from the EMC and DECT serial number of a PT the associated IPUI type N.	
Parameter	pIpui	Destination-buffer for the IPUI
	emc	The equipment manufacturer code for Höft & Wessel modules at this point of time is always 322
	dectno	DECT serial number of the PT
Return	Always 0	
Note	• Only the first 5 bytes of pIpui->data are used at an IPUI type N. • Format: TE EE EN NN NN T = 4 bit type E = 16 bit EMC N = 20 bit DECT serial number	

3.5.11.25 DectReadTo

Function	int DectReadTo(int nChannel, void *pBuf, int nMaxLen, unsigned long timeoutMs);	
Description	Read the data from a channel with a timeout.	
Parameter	nChannel	Channel-number to read from
	pBuf	Pointer to a buffer for returning the data
	nMaxLen	Max. number of bytes to read
	timeoutMs	Timeout in milliseconds. The time to wait for an available data-block
Return	>= 0	Number of actually read bytes
	< 0	Error
Note	• The timeout is not the overall time for reading nMaxLen bytes, but the time to wait between the individual data-blocks to read. • See DectRead()	

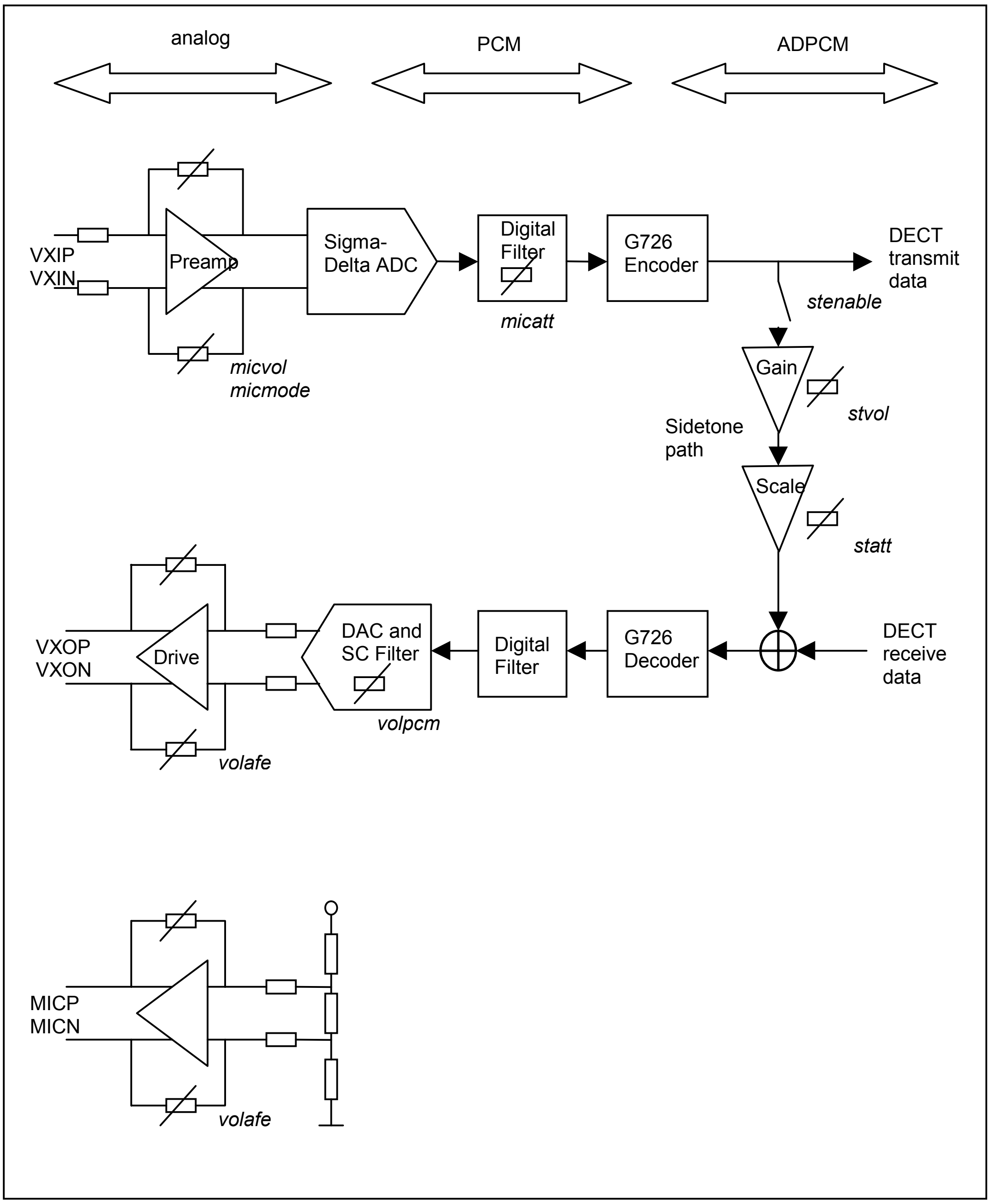
3.5.11.26 DectWriteTo

Function	int DectWriteTo(int nChannel, void *pBuf, int nLength, unsigned long timeoutMs);	
Description	Write the data to a channel with a timeout.	
Parameter	nChannel	Channel-number to write to
	pBuf	Pointer to a buffer holding the data to write
	nLength	Number of bytes to write
	timeoutMs	Timeout in milliseconds. The time to wait for sending a single data-block
Return	>= 0	Number of actually written bytes
	< 0	Error
Note	• The timeout is not the overall time for sending nLength bytes, but the time to wait between the individual send data-blocks. • After returning from this function it is possibly that the buffer is still filled with 4*26 bytes unconfirmed data (query with DectGetTxPending()). • See DectWrite(), DectGetTxPending()	

3.6 Voice mode

3.6.1 Block diagram

The following diagram describes the audio path on HW 86010 and HW 86020 modules:


analog
PCM
ADPCM
VXIP
VXIN
Preamp
micvol
micmode
Sigma-Delta ADC
Digital Filter
micatt
G726 Encoder
DECT transmit data
stenable
Gain
stvol
Sidetone path
Scale
statt
DECT receive data
G726 Decoder
Digital Filter
DAC and SC Filter
volpcm
Drive
VXOP
VXON
volafe
MICP
MICN
volafe

3.6.2 Advises on voice commands

To enter commands, the module must be set into configuration mode. With entering the command <SPVOICE on> the voice mode is enabled while simultaneously the data mode is disabled.

3.6.2.1 Speaker output configuration

```
SPVSPE <volafe> <volpcm>↵
```

volafe - Analog front-end speaker output attenuation:

volafe	max output level		Output level for 0 dBm0 PCM		Gain
	VRMS	Vpeak	VRMS	Vpeak	dBVrms / dBmo
0	2.192	3.100	1.527	2.160	3.7
1	1.741	2.462	1.213	1.715	1.7
2	1.383	1.956	0.963	1.363	-0.3
3	1.099	1.554	0.765	1.082	-2.3
...	...	...	...	...	...
15	0.069	0.098	0.048	0.068	-26.3

Gain values in dBVrms / dBm0: a digital 1dBmO signal with volafe = 0 comes out as an analog signal with 4.7dBVrms (1+3.7). The units for analog sine wave signals are Vrms and dBVrms. The usual unit for digital code corresponding to a sine wave is dBm0 where the maximum sine wave at PCM level is 3.14d Bm0. Regarding the analog input, 0dBm0 can be considered like a “safety” working level to prevent unwanted overshoot on the external signal. The analog front-end characteristic remains linear until 3.14 dBm0. 0 dbm0 = +/- 5706 and max. PCM code at 3.14 dBm0 = +/- 8192 (= 14 bit linear coding).

volpcm - Digital path receive volume:

The decoded PCM signal is multiplied by this value to control PCM output volume.

Example:

volpcm	Gain(dB)
255	+5.99
254	+5.95
253	+5.92
192	+3.52
128	0.00
64	-6.02
32	-12.04
16	-18.06

volpcm	Gain(dB)
8	-24.08
7	-25.24
6	-26.58
5	-28.16
4	-30.10
3	-32.60
1	-42.14
0	- oo

3.6.2.2 Microphone input configuration

SPVMIC <micvol> <micatt> <micref> <micmod>↵

micvol - Analog front-end preamplifier gain:

micvol	Amplification mode			Attenuation mode		
	Max. Input Level	Input Level for 0dBm0 PCM code	Transmit Gain	Max. Input Level	Input Level for 0dBm0 PCM code	Transmit Gain
	mVrms	mVrms	dBm0 / dBVrms	Vrms	Vrms	dBm0 / dBVrms
0	348.0	242.4	12.3	0.454	0.316	10
1	246.3	171.6	15.3	0.571	0.398	8
2	174.4	121.5	18.3	0.719	0.501	6
3	123.4	86.00	21.3	0.905	0.631	4
4	87.39	60.88	24.3	1.139	0.794	2
5	61.87	43.10	27.3	1.434	1.000	0
6	43.80	30.51	30.3	1.806	1.259	-2
7	31.01	21.60	33.3	2.274	1.585	-4
8	21.95	15.29	36.3	micvol[3] ignored!		
9	15.54	10.83	39.3			
10	11.00	7.66	42.3			
11	7.79	5.43	45.3			
12	5.51	3.84	48.3			
>12	3.9	2.72	51.3			

CCITT recommendations are only guaranteed at 7 lowest micvol values.

dBm0/dBVrms means: an analog 2 dBVrms signal with micvol = 0 becomes a digital signal with 14.3 dBm0 (2 + 12.3) in amplification mode or 12 dBVrms in attenuation mode.

micatt - Attenuation level of digital post filter:

micatt	Gain [B]
0	0
1	-0,5
2	-1.0
3	-1.5
4	-2.0
>4	-2.5

Digtal post filter realises a band pass filter to ensure CCITT requirements and to remove DC- offsets in the signals.

micref - Microphone reference trimming:

micref	Gain
0	0 dB
1	+0.17 dB
...	...
7	+1.19 dB

micod - Controls preamplifier mode:

micmod	Preamplifier mode
0	Amplification
1	Attenuation

3.6.2.3 Sidetone configuration

SPVST <stenable> <stvol> <statt>↵

stenable - Enables (1) or disables (0) side-tone generation.

stvol - Value to be multiplied with the Sidetone:

The transmit signal is multiplied by stvol/31 and scaled by the statt parameter, then added to the receive signal.

statt - Post-scale factor for side tone after the individual gain control:

statt	Gain [dB]
0	0
1	-6
2	-12
3	-18

4.0 Abbreviations

ARI	access rights identity
CLDPS	connection-less DECT packet system
CRC	cyclic redundancy checksum
DCE	data communication equipment
DECT	digital enhanced cordless telecommunications
DLC	data link control layer
DNR	DECT serial number
DSP	data service profile
DTE	data terminal equipment
EMC	ETSI manufacturer code
FCS	frame check sequence
FHSS	frequency hopping spread spectrum
FPN	fixed part number
FT	fixed termination
GAP	generic access profile
IPEI	international portable equipment identity
IPUI	international portable user identity
ISDN	integrated services digital network
I/O	input / output
LAP	link access protocol
LSB	least significant bit
MAC	medium access control layer
MSB	most significant bit
NLF	new link flag
NWK	network layer
PARI	primary ARI
PARK	portable access rights key
PCM	pulse code modulation
PIN	personal identity number
PLI	PARK length indicator
PT	portable termination
RF	radio frequency
RFP	radio fixed part
RFPI	radio fixed part identity
RPN	radio fixed part number

SAPI	service access point identity
SARI	secondary ARI
SDL	specification description language
SK	subscription key
SMK	subscription master key
UAK	user authentication key

5.0 Index

ack pending condition 94
acknowledgement 92, 96, 98
Active state 94, 99
address field 85
API 117
architecture 12
ARI 17
authentication 16
base station 12
baud rate 23, 75, 84
BOOT0 signal 15
BOOT1 signal 15
call control 13, 76
call control protocol 86, 109
call handle 85, 109
call release 79, 116
command / response bit 93
command channel 83
configuration command 23
configuration mode 14, 15, 23, 80
configuration protocol 23
ConnectInd command 110, 115
ConnectReq command 112, 115
contact 8
control byte 91
CRC command 72
CTSO signal 75
data mode 14, 15
data transmission 80
DCDIO signal 74, 75, 76
DCE mode 74, 76, 78
decimal 28
DECT protocol stack 9
DECT_CALLBACK_FUNC_T 117
DectBuildIpuiTypeN 125
DectClose 119
DectConnectReq 120
DectDestroy 118
DectDisconnectReq 121
DectGetBytesAvail 122
DectGetConnStatus 121
DectGetIpui 122
DectGetLineStatus 121
DectGetTxFree 122
DectGetTxPending 122
DectInit 118
DectLapStateCfm 123
DectLapStateGetCallNr 125
DectLapStateGetIpui 124
DectLapStateGetLen 124
DectLocationRes 123
DectOpen 119
dectprot.dll 117
DectRead 119
DectReadTo 126
DectRegisterCallback 118
DectSwitchLocation 124
DectSwitchRoaming 123
DectWrite 120
DectWriteTo 126
DIARI command 49
DisconnectInd command 111, 116
DisconnectReq command 112, 116
DISUB command 46
DNR 17
download mode 14
DPCUST command 63
DPDSD command 62
DPDSI command 61
DPRETRY command 68
DPSCFG command 66
DPTIMEOUT command 70
DSRO signal 76, 77
DTE mode 74, 76
DTRI signal 15, 76, 77, 80
EasySubs 11, 21
EMC 17
error correction 90
escape sequence 80
EXIT command 15, 23, 73
final bit 91, 97
firmware 9, 11, 14
fixed termination 12
flag field 85
flow control 75, 90, 97
FPN 17
frame check sequence 87
GALL command 71
GAP 11
GHALL command 29
GHFL command 31
GHQUAL command 30
GHRSSI command 30
GHTY command 29
GIAIR command 42
GIARI command 49
GIPARK command 48
GISK command 47
GISUB command 45
GMALL command 33
GMF command 34
GMM command 35
GMQ command 34
GNALL command 39
GNDNR command 40
GNEMC command 39
GNETH command 40
GNSER command 40
GNUNR command 41
GOK command 72

GPALL command 59
GPANT command 31
GPBD command 54, 75, 84
GPCC command 56
GPCLDPS command 37
GPCOM command 55
GPCTR command 36
GPCUST command.....................................63
GPDSD command.......................................62
GPDSI command 61
GPECC command.......................................57
GPEQ command 75, 84
GPLOC command 64
GPMP command...38
GPPR command 35
GPRETRY command..................................68
GPSYWD command...................................65
GPTIMEOUT command 70
GPTM command 60
GPUI command...58
GPVMIC command 50
GPVOICE command 51
GPVSPE command.....................................52
GPVST command 53
GSALL command..32
GSNR command 32
GSVER command 33
handset..12
hardware handshake...................................75
HDLC 83
HDLC framing..85
hexadecimal 28
identities 17
Idle state 94, 99
I-frame 91, 92, 96, 98, 109
incoming call..................................76, 78, 115
information field 91
information frame 91
interworking 10
IPBD command 54, 75, 84
IPEI 19
IPUI..................... 19, 110, 111, 112, 113, 114
LAP..82, 90
LAP establishment 99
LAP frame 91
LAP-D 90
LAP-U 90
lead signals...76, 80
LocationInd command 113
LocationRes command..............................114
LU2 82
LU3 .... 9, 13, 82, 90
mode selection 15
multi-bearer 13
multiple frame operation.........................90, 95
new link flag...93
octal 28
offline subscription.................................16, 22
on-air subscription 16, 22
outgoing call 76, 115
overview 9
padding 85
PARI 17
PARK 18, 28
peer busy condition 95, 97
pending frame 96
PIN code 20
pipelining 75
PLI 18
point-to-multipoint 12
point-to-point 12
poll bit 91, 97
portable termination 12
protocol data mode 14, 81, 83
protocol field 86
radio fixed part 12
re-establishment 99
reject exception condition 92, 94, 98
REJ-frame 91, 92, 97
response string 23
return code 23, 27
RFP 12
RFPI 18
RIIO signal 74, 75, 76
RNR-frame 91, 92, 97
RPN 18
RR-frame 91, 92, 97
RS-232 ....23, 74
RTSI signal 15, 75
RXDO signal 84
SABM-frame 91, 92, 99
SDL 100
security..16
self busy condition 92, 94, 98
sequence number 91, 92, 95
sequence variable 95
service line 8
SIAIR command 42
SIARI command...49
SIPIN command....................................20, 47
SISMK command 48
SISUA command 43
SISUB command 44
SISUD command 45
SK 20
SMK 21
SPANT command 31
SPBD command 54, 75, 84
SPCC command 56
SPCLDPS command 37
SPCOM command 55
SPCTR command..36
SPCUST command......................................63
SPDSD command..62
SPDSI command 61
SPECC command..57
SPEQ command 74, 75, 84
SPLOC command 64
SPMP command 38
SPPR command 14, 15, 35, 74, 81

SPRETRY command 67
SPSYWD command.................................... 65
SPTIMEOUT command 69
SPTM command 60
SPUI command... 58
SPVMIC command 50
SPVOICE command 51
SPVSPE command..................................... 52
SPVST command 53
string 28
subscription.......................... 11, 12, 16, 20, 21
supervisory frame 91
timeout 95
timer.. 95
timer recovery condition 95, 97
transparency.. 89
transparent data mode 14, 74, 82
TXDI signal 84
UA-frame 91, 92, 99
UAK 20
voice mode 127
WaitEstablish state 94
web-site 8