

# The EPG2 Electropalatograph Device

Werner Thon



Institute of Phonetics and Digital Speech Processing  
University of Kiel, Germany

# Contents

	page
<b>1. Introduction</b>	
1.1 Preface .....	3
1.2 Purpose of this Document .....	3
1.3 Disclaimer .....	4
1.4 Copyright .....	4
<b>2. EPG2 User's Guide</b>	
2.1 EPG2 Operating Principles .....	5
2.2 EPG2 Usage .....	6
<b>3. Hardware Description</b>	
3.1 Overview .....	7
3.1.1 Front End Domain .....	7
3.1.2 Back End Domain .....	7
3.2 Main Board .....	7
3.2.1 Power Supply and Battery Monitor .....	7
3.2.2 Injection Voltage Generator .....	7
3.2.3 Detectors and Address Buffers .....	8
3.2.4 Opto Couplers and Bus Driver .....	8
3.2.5 RS232 Circuitry .....	8
3.3 MUX .....	9
3.4 Coder.....	9
3.4.1 Coder Processor Module .....	9
3.4.2 Codec Module .....	9
3.4.3 Auxiliary Circuitry .....	10
3.5 Display .....	10
3.6 PCB Layout and Production .....	10
3.7 Case .....	11
<b>4. Software Description</b>	
4.1 General Software Structure .....	12
4.2 Display Processor Software .....	12
4.3 Coder Processor Software .....	13
4.3.1 Initialization .....	13
4.3.2 Interrupt subroutine .....	13
4.3.3 Main Program: Mode ‚Display‘ .....	14
4.3.4 Main Program: Mode ‚Coding‘ .....	14
4.3.5 Main Program: Mode ‚Calibration‘ .....	15
<b>5. Device Programming</b>	
5.1 Display .....	16
5.2 Coder .....	17
<b>Appendices</b>	
A Hardware	
B Software	
C Photos	
D Data Disk	

## 1. Introduction

### 1.1 Preface

The development of electropalatograph devices at the Kiel Institute of Phonetics and Digital Speech Processing (*ipds*) dates back to the mid-seventies, when *Heinz Janssen* developed the first Kiel Palatograph (which is still in use for demonstration purposes). The device contained a CMOS-Multiplexer and a LED display. It was a stand-alone device just to display artificial palate contact information. Several attempts have been made during the following years to connect the device to the different computers in use over the years. As the EPG data rate is low (7kbit/sec at a frame duration of 10 ms and 70 contacts on the pseudo palate) it was easy to set up an EPG interface, but it had to be designed for a specific type of computer and the main problem was to synchronize the EPG data stream with the data stream of a microphone signal recorded in parallel. Without the exact synchronization to the audio signal EPG data are of little use.

A new attempt was started in 1997 based on an idea of *Michel Scheffers*:

- \* As a stereo audio input is available for any type of computer (PC, Mac, workstations) the EPG signal should be coded into an audio signal of the same bandwidth and dynamic properties as the microphone signal.
- \* Both signals should be recorded as stereo signal, thus implicitly providing synchronicity.
- \* Decoding of the EPG signal should be done offline.

The first hardware device for this purpose was developed by *Heinz Janssen* and the author. *Michel Scheffers* wrote the decoding software. A combined frequency and amplitude coding scheme was used and the device accomplished that with a set of 11 precision hardware sine generators controlled by a processor module programmed in BASIC. The coding scheme and the device have been presented in a poster at ICPhS 1999 in San Francisco.

The next step was to replace the hardware sine generators by a completely software based solution running on a sufficiently fast microcontroller. A prototype has been built, with *Heinz Janssen* developing the palatography circuitry and the author an 80C165 microcontroller based coder programmed in C. This device, ready in 2001, used the same decoding software as its predecessor.

As the prototype was quite experimental an easy duplication was not possible. The author therefore redesigned the device maintaining its operation principle, but with the aim of enabling other scientific institutions to duplicate it. For the user's convenience a touch screen display has been added, resulting into the current EPG2 device.

### 1.2 Purpose of this Document

This document is intended to supply any information needed to successfully duplicate the EPG2 device. This work requires a skilled electronics technician and at least a small electronics workshop.

The person doing this work should have the following skills:

- experience in developing analog circuitry (audio frequency range),
- experience in developing digital circuitry (TTL, CMOS),
- some experience with embedded microcontrollers,
- some experience in programming, especially in C (to understand the coder program).

If modification of the coder program is intended, the person should have a profound knowledge of C.

The electronics workshop should provide the following facilities:

for testing and hardware debugging:

- a 100MHz dual trace oscilloscope (preferably a digital one, like Tektronix TDS 3012)
- an audio frequency sine generator,
- a TTL pulse generator,

for device programming:

- a DOS or WINDOWS based PC with serial port,
- a lab power supply,

and, for mechanical works:

- a basic mechanical work bench.

### 1.3 Disclaimer

Any information in this document and the attached data disk is supplied ,as is'. It has been compiled with care but is delivered with absolutely no warranty. Neither the author nor the Kiel Institute of Phonetics and Digital Speech Processing shall be held liable for any damage resulting from the use of this information. The use of this information is completely on your own risk.

### 1.4 Copyright

This document and the attached data disk are copyrighted. The information is provided for scientific purposes only, any commercial use is strictly prohibited.

© ipds 1997-2003

## 2. EPG2 User's Guide

### 2.1 EPG2 Operating Principles

As with any other electropalatograph a small high frequency voltage is injected into the test person's body by a hand-held electrode. The contacts of an artificial palate are periodically scanned for the presence of the injection voltage indicating that the contact is touched by the tongue. The ipds artificial palate has 70 contacts, thus a data frame of 70bit of contact information is acquired for each complete scan. The frame rate is set to 100Hz, i.e. the frame duration is 10ms.

The front end circuitry physically connected to the test person is battery operated and completely electrically isolated from the rest of the EPG2 device, thus ensuring the electrical safety of the test person. Data transmission is done via opto couplers.

The frame data stream is input to a coder implemented as software running on an embedded microcontroller. For each frame the coder generates (via digital-to-analog conversion) an analog signal of 8ms duration followed by a pause of 2ms. The analog signal is the sum of 11 sine waves, each with specific frequencies and amplitudes. The data frame is divided into 10 'chunks' of 7bit, each chunk controlling one sine wave. A different set of four frequencies is assigned to each chunk. The two lower bits select one of the four frequencies, the upper 5 bit select one of 32 amplitudes. The eleventh sine wave has its own fixed frequency and maximum amplitude. It is used as reference by the decoding software. Thus the spectrum of the coded EPG signal continuously reflects the artificial palate contact status. The EPG signal is recorded together with the microphone signal as stereo signal via the host computer's sound input, thus implicitly providing synchronicity between the two signals.

To make use of the information in the coded EPG signal it has to be decoded into the frame data stream synchronized to the microphone signal. This is done offline by the decoding software. It may run on the computer where the recording has been made or on any other computer the recorded data files have been transferred to. The decoding software has to detect the frequencies and the amplitudes of the coded EPG signal components. The amplitudes are affected by the transfer function of the coder output and by the transfer function of the sound input of the recording computer. For an accurate reconstruction of the data frames the decoding software has to account for the overall transfer function 'coder output to sound input'. This information is provided by recording a special calibration signal. It consists of 64 frames of the coded EPG signal with sine waves at selected fixed frequency combinations and maximum amplitude. The calibration signal is repeated cyclically. A recording of some seconds is sufficient for the decoding software to determine the overall transfer function which is then used within the decoding process.

*Note: The details of the decoding software are not covered by this document.*

The sound input channel of the computer used for recording (sound card or on-board sound chip) should at least meet the following requirements:

- \* signal-to-noise ratio (including total harmonic distortion): 60dB or better,
- \* upper -3dB frequency band limit: 7kHz or higher.

To provide a convenient user interface a touch screen display has been incorporated into the EPG2 device. It enables the experimenter to control the device operation modes. Furthermore, it provides a visual display of the current contact status (palatogram).

Thus the EPG2 device has three major operation modes (see 2.2):

- \* Display - the current contact status is displayed continuously  
- no coded EPG signal is output
- \* Calibration - the fixed calibration signal is output cyclically repeated
- \* Coding - a coded EPG signal reflecting the current contact status is output continuously

Additionally, an analog microphone amplifier channel is implemented. Both analog output signals (coded EPG signal and microphone signal) may be adjusted in their amplitude on the touch screen.

## 2.2 EPG2 Usage

EPG2 usage is straightforward and simple:

- \* Connect an artificial palate (directly or with an appropriate adapter) to the 50pole jacks R and L.
- \* Connect the hand-held electrode to the jack Vinj.
- \* Connect an electret condenser microphone to the jack MICIN.
- \* Connect a stereo cable with two cynch plugs to the jacks PALOUT (left stereo channel) and MICOUT (right stereo channel).
- \* Connect the other end with a 3.5mm stereo plug to the line input jack of a computer's sound card.
- \* Power up the computer.
- \* Connect the wall plug power supply to the jack 8..10VDC.
- \* Plug the power supply into a mains receptacle, thus powering up the EPG2 device.
  - The display shows the ipds logo and then an opening screen reporting the initialization of the device.
  - After initialization the device enters the mode ,Display' (see appendix C-4.1).
  - On the right three touch buttons (Display, Calibration, Coding) provide mode selection capabilities.
  - The button ,Display' is blinking to show the current mode.
  - A field at the lower right corner shows the battery status:
    - o.k. means more then 10% of battery capacity left,
    - <10% (blinking) means less than 10% of battery capacity left.(With an estimated operation time of a new battery pack exceeding 50 hours, more than 5h of operation time is left in this case, more than enough to finish the current experiment.  
*Note: the battery status field is updated only when the mode is changed.*)
  - The left part of the display shows a palatogram with the current contact status updated continuously. (small circle: tongue does not touch a contact; big circle: tongue does touch a contact)
  - This mode is intended for demonstration purposes and to fine adjust the injection voltage amplitude with a potentiometer just besides the jack Vinj (use a small blade screwdriver!).
- \* Touch the button ,Calibration'.
  - The mode ,Calibration' is entered; the button ,Calibration' is blinking (see appendix C-4.2).
  - A fixed cyclic calibration signal (independent of the current contact status) is output at jack PALOUT.
- \* Invoke a recording program providing an input level meter (like *Cool Edit* for WINDOWS).
- \* Set recording parameters to stereo, 16 bit, 16kHz (or higher).
- \* With the touch slider EPG on the display and the input level slider of the recording program adjust the recording level to 0...2dB below clipping level.
  - This ensures the optimum signal-to-noise ratio.
- \* Record some 10 seconds of this calibration signal.
  - These data are used by the decoding software to account for the overall frequency response of the EPG2 device and the sound card.
  - These data have to be acquired only once per environment (sound card + host computer).
- \* Save the recording into a \*.wav file.

*Note: Though it should be sufficient to do this once per environment it is recommended to record a calibration file before each major recording session, or when errors are reported during decoding; in this case the calibration file may even be recorded afterwards.*

- \* Touch the button ,Coding'.
  - The mode ,Coding' is entered; the button ,Coding' is blinking (see appendix C-4.3).
  - The coded EPG Signal continuously reflecting the contact status is output at jack PALOUT .
  - The amplitude of the coded signal will never exceed the amplitude of the calibration signal.
- \* With the slider MIC adjust the recording level of the microphone signal to about 10dB below clipping level.
  - Note, that the slider EPG is locked now!
  - Do not readjust the input level slider of the recording program, as this will influence both stereo channels!
- \* You are now ready to do an EPG recording!
- \* You can redo the previous steps by touching the appropriate mode button and then doing adjustments as required.
- \* When finished, save the recorded signal into a \*.wav-file.
- \* The files have to be processed further by the decoding software.

*Note: The details of the decoding software are not covered by this document.*

### 3. Hardware Description

#### 3.1 Overview

The block diagram (Appendix A-1.1) shows the main components of the EPG2 device. It is divided into two distinct electrically isolated domains. The front end domain (upper part) is battery operated for safety reasons. It contains the circuitry connected to the pseudo palate, i.e. to the test person. The back end domain is powered by 8..10V DC supplied by a plug power supply (not shown). It contains data processing and display circuitry. Data between the two domains are conveyed via optocouplers and a relay.

##### 3.1.1 Front End Domain

The injection voltage generator supplies a 50kHz square wave  $V_{inj}$  of  $4V_{max}$  fed into the test person's body via a hand-held electrode. The pseudo palate with 70 contacts max. is connected to two multiplexer circuits (MUX left, MUX right, each for one half of the pseudo palate), which scan the contacts for the 50kHz signal. The MUXes are controlled by a 4bit address supplied by the coder in the back end via the address buffers. The 7bit MUX output is fed to 7 detectors which discriminate the presence of a 50kHz signal at their inputs. The front end is powered by a 4x1.5V battery pack switched on by a relay contact. The battery voltage is monitored and regulated to 4.8V. All signals to/from the back end are conveyed through opto couplers to provide electrical insulation between the two domains.

##### 3.1.2 Back End Domain

The coder is the central component of the back end. It essentially consists of a processor (with RAM and Flash-ROM) and a sound chip to generate the analog output signals. The coder generates the 4bit MUX address and receives the 7bit detector data via a bus driver. The control signals PALPU and PALGON as well as the monitor signal POK are processed by the coder. The digital EPG data are coded into an analog signal and output by the soundchip (PALOUT). The soundchip also provides a digitally controlled analog path for the microphone signal (MICIN / MICOUT).

The display is an off-the-shelf touch screen display with its own processor and flash-ROM. It is loosely coupled to the coder via a serial RS232 connection. Alternatively, this coder port is used for programming purposes. The power supply converts 8..10VDC input into regulated 5VDC for coder, display and other back end circuitry.

#### 3.2 Main Board

The main PCB contains all circuitry except MUXes, coder and display.

##### 3.2.1 Power Supply and Battery Monitor (Appendix A-1.2.1)

Any 8..10VDC voltage @2A may be used for supply. The diodes CR1..CR4 provide polarity protection. Two 7805s regulate the supply voltages VCCB (display) and VCCA (coder and other circuits of back end). If VCCA is up, 7705A switches VBAT on via Rel. LM2931 regulates the battery voltage to VCCM (4.8VDC, adjustable) for use by the front end, if /PALPU (from the coder) is asserted. ICL7665 monitors the battery voltage and asserts POK, as long as  $V_{BAT} \geq 5.25V$  (approx. more than 10 % of battery capacity left.)

*Notes: Front end and back end power supplies are completely isolated from each other!!  
Do not connect DIGGND (back end) and ANAGND (front end) in normal use.  
This will degrade electrical safety and may degrade the pseudo palate signal quality.*

*Usefull battery life of a pack of four type AA alkaline batteries is estimated to approx. 50 hours.*

##### 3.2.2 Injection voltage generator (Appendix A-1.2.2)

A free running 200kHz oscillator (4093) feeds two divider stages (4013) controlled by /PALGON (from coder). If /PALGON is asserted, the impedance converter T2 supplies the amplitude-adjustable 50kHz signal  $V_{inj}$  to the hand-held electrode.

### 3.2.3 Detectors and Address Buffers (Appendix A-1.2.3)

The main PCB provides space for two 16pin connectors for two MUX PCBs, as well as (alternatively) for one 2x17pin connector for a 34wire flat ribbon cable, if it is desired to place the MUXes in a separate case.

*Note: The alternative placement of the MUX PCBs in a separate case has not been tested yet!  
A twisted pair ribbon cable of not more than 1m length should be used.*

All connectors share a common address (A...D,/D) and data (M0...M6) bus. The addresses are buffered by U12 (4009). The selection of D or /D is done by a jumper on the MUX PCB (see 5.3). The MUX outputs (zero or 50kHz square wave) are fed into 7 detectors (U8...U11, 4528). The retriggerable monoflops have a hold time of > 21us, thus, a constant HIGH signal is output as long as a 50kHz signal is present at the input.

*Note: The hold time of the monoflops is critical!! 21us <= thold <= 25us  
It seems, that 4528s of different manufacturers and even of different production lots yield different timing parameters!  
Use devices of the same manufacturer and if possible, of the same production lot!  
Test one device, choose Rx and Cx for 23us hold time, then the other devices wired with the same components will be likely to yield a hold time within the limits.*

### 3.2.4 Opto Couplers and Bus Driver (Appendix A-1.2.4)

Communication between the (electrically isolated) domains ‚front end‘ and ‚back end‘ is achieved by 14 opto couplers (U13...U19, 2730):

front end to back end: MD0...MD6 -> /CD0.../CD6  
POK ->/PALPOK

back end to front end: PALPU ->/PALPU  
A0...A3 ->/BA.../BD  
PALGON ->/PALGON

*Note: Opto couplers (especially for the back-to-front direction) should be selected for equal rise and fall times to minimize phase differences!*

The data bus signals /CD0.../CD6 are strobed on the coder’s processor bus D0...D6 by the inverting bus driver U20 (LS640) with /ISTRB . All signals to/from the coder run via the 2x20pin connector for the coder.

### 3.2.5 RS232 Circuitry (Appendix A-1.2.5)

This circuitry connects

- the 16pin connector for the display,
- the 3pin external RS232 connector,
- the 3pin coder RS232 connector (see also 3.4.3),
- pin a9 of the 2x20 coder connector.

To conveniently use the coder RS232 port to connect either to the display or to the external RS232 connector to download programs into the coder processor’s flash ROM, the serial signals are routed via a DPDS switch. In position ‚Run‘ (R on the main PCB) the coder is connected to the display, in position ‚Program‘ (P on the PCB) the coder is connected to the external RS232 connector.

The 16pin display connector also provides power supply to the display and some flow control wiring. Additionally, the display’s RTS signal is converted to TTL (CR7) and routed to pin a9 (DISRTS) of the coder main connector.

### 3.3 MUX (Appendix A-1.3)

The pseudo palates used at *ipds* have 70 contacts, 35 on each half of the palate. The 2x35 wires from the two halves are bundled and connected to two 50pin SubD-type plugs. The EPG2 device contains two identical MUX PCBs, one for each half of the pseudo palate.

Each MUX is equipped with a 50pole SubD-type jack, 35 pins are used, the remaining 15 pins are grounded. Multiplexing is done by seven analog CMOS multiplexer chips (U1...U7, Motorola MC14512CP). Their outputs are connected to the 7bit data bus. Five of the eighth inputs are used (X1...X5) the others are grounded (X0,X6,X7). Thus, 35 signals are acquired in 5 steps. Addressing is done by the three address signals A,B,C. Address signal D and its complement are used to select right or left half. A jumper connecting the disable pin to D or /D is used to define the MUX as ‚left‘ or ‚right‘. As disabled outputs are ‚high Z‘ both MUX PCBs are wired to the same data bus (M0...M6) on the main PCB.

Connection to the main PCB may be achieved either by fitting the MUX PCBs with 1x16pin connectors mating those on the main PCB (for direct mounting on the main board) or by fitting them with 2x17pin connectors, putting them on top of each other and connecting them to a 34pole twisted-pair flat ribbon cable (for connecting to the appropriate 2x17 connector on the main PCB). (see also 3.2.3)

*Note: Due to extremely high impedances of the MUX chip inputs crosstalk may occur between the input lines. To prevent this, connect 3M3 resistors between each line and ground. 35 mounting holes are provided on the MUX PCB adjacent to the SubD jack to solder in the resistors in upright position (see appendix C-1.2). This seems to be necessary when using MUX chips manufactured from the nineties on. Chips from earlier manufacturing lots may well work without these resistors.*

### 3.4 Coder (Appendix A-1.4)

The coder PCB contains

- the coder processor module,
- the codec module with a sound chip,
- some auxiliary circuitry, like LEDs with drivers, maintenance inputs and outputs, and a MIC preamplifier.

The coder controls the functionality of the main PCB, reads digital EPG data acquired by the MUXes, codes these data into a series of samples which is converted to an analog output signal by the soundchip on the codec module.

#### 3.4.1 Coder Processor Module

The coder processor is an off-the-shelf module (PHYTEC microMODUL-165) containing an Infineon 80C165 microprocessor, 256kB RAM and 256kB flash-ROM and some ‚glue‘ circuitry. The processor module is plugged on top of the coder PCB, it acts as the CPU core of the coder, with the codec module and the connected components of the main PCB as peripheral devices.

Details of the PHYTEC microMODUL-165 are available from

<http://www.phytec.de>

#### 3.4.2 Codec Module (Appendix A-1.5)

The codec module is a small PCB plugged on top of the coder PCB. It contains an AD1845 sound chip, a highly programmable device for analog signal I/O, together with the necessary analog circuitry. Its tasks are:

- D/A conversion of the processor generated output signal,
- digital control of the analog output amplitude,
- digital control of the analog microphone signal amplitude
- control of two status LEDs.

Details of the AD1845 are found in the data sheet available from

<http://www.analog.com>

### 3.4.3 Auxiliary Circuitry

Two status LEDs with their driver transistors are connected to the codec module. Two static control signals (MOD0, MOD1) may be set by jumpers on header J1. The signals SYNC, ICHECK and MCHECK may be used as input to an oscilloscope to monitor the program timing (see 4.3.2).

The pins Res and Prg are used to connect two push-button switches used to download a program into the processor module's flash-ROM via TxD and RxD, which connect the processor module's RS232 port to the RS232 circuitry on the main PCB (see 3.2.5).

Three jacks connect the coder to the analog ,outside world':

- MICIN	input signal from microphone	
- MICOUT	right stereo output signal:	microphone
- PALOUT	left stereo output signal:	coded EPG signal

The microphone input signal is routed through a 16dB preamplifier (AD820) to the codec module with another 20dB amplitude gain. Both output signals are digitally attenuable under program control. The transistor 2N4123 provides for a phantom supply voltage for electret condenser microphones. Preamplifier gain and phantom power supply have to be adapted to the microphone used.

### 3.5. Display

The display is an off-the-shelf module with a 5.7" 320x240 pixel LCD touch screen (EA KIT320-8). It is loosely coupled to the processor module via its RS232 port. The module has its own display processor, RAM and flash-ROM. Display macros may be downloaded offline into the flash-ROM for later execution initiated by simple commands via the serial port. Software definable touch areas provide for the functionality of push-button switches and slides (,potentiometers'). Thus, the EPG2 device has no mechanically moving parts.

The display processor controls the complete user communication and the major operation modes of the EPG2 device (see appendices C-4.1, C-4.2 and C-4.3).

Details of the EA KIT320-8 are found in the data sheet available from

<http://www.lcd-module.de>

### 3.6 PCB Layout and Production

The PCB layouts have been designed with the layout program EAGLE V4.03 available for Windows and Linux systems from

<http://www.cadsoft.de>

The layouts are made for two layer PCBs on a 0.05" grid. The following layout files are included on the data disk (see appendix A-2 for printouts, appendix D for data files):

<u>PCB</u>	<u>layout file</u>
Main Board	<i>main_v2.0.brd</i>
Mux	<i>mux_v2.0.brd</i>
Coder	<i>coder_v2.0.brd</i>
Codec	<i>codec_v2.0.brd</i>

Though 0.05" two layer PCBs may be produced completely by a sufficiently equipped electronics lab, we preferred the production by a professional PCB company. This is possible now even for single pieces at a reasonable price. Details may be found at

<http://www.pcb-pool.de>

Links to other PCB companies may be found at

<http://www.cadsoft.de>

Professional PCB production has a number of advantages:

- layout files are downloaded via Internet  
(EAGLE files are commonly accepted),
- PCBs are delivered within about two weeks,
- any PCB shape is possible  
(see layout of the mainboard, appendix A-2.1),
- contacts are drilled and plated-through,
- solder-stop coating is put on both layers,
- PCBs are electrically tested, if ordered (and payed!!)

This justifies a price of around EUR 70 per PCB.

### 3.7 Case

A two shell plastic case has been chosen to accomodate the circuitry. The touch screen display nicely fits into the top shell with a bezel supplied by the display manufacturer (see appendix C-3.1).

The shape of the main board which carries the rest of the circuitry has been designed to fit into the bottom shell (see appendices A-3.3.1 and C-1.1 ).

The case has been purchased from a local dealer. Thus, for duplication a new solution has to be found. A similar two shell (metal) case is available with order # 52 04 03 from

<http://www.conrad.com>

## 4. Software Description

### 4.1 General Software Structure

The underlying hardware structure (see appendix B-1) is reflected in the general software structure. From the software point-of-view the EPG2 device consists of two processors, each with its own peripherals. They are working in parallel and communicate via a RS232 link.

The EPG2 device may be operated in one of three major modes:

- Display: MUX data are displayed on the screen, zero output at PALOUT.
- Calibration: a defined cyclic calibration signal is output at PALOUT.
- Coding: MUX data are coded into the EPG signal output at PALOUT.

Modes are changed by pressing the appropriate touch field on the screen. The display processor sends a mode code to the coder processor, which switches its mode and acknowledges by sending the appropriate macro command to the display processor. The execution of the macro yields a new screen with mode-specific elements.

In the mode ‚Display‘ the coder processor then continuously sends screen commands to display MUX data. In the modes ‚Calibration‘ and ‚Coding‘ touchable bar graphs („sliders“) are displayed. If touched the display processor sends the new bar graph value. The coder processor updates the codec and acknowledges by sending a macro command to update the bar graph.

This two-processor structure clearly separates the (comparatively slow) ‚human interface‘ tasks from the fast ‚real time‘ signal processing tasks.

### 4.2 Display Processor Software

The display processor software is a collection of screen macros, each consisting of a sequence of screen commands or macro commands. Details of the command syntax may be found in the data sheet of the display module (see 3.4). The macros are downloaded offline into the display processor’s flash-ROM (see 5.1). Macros are called either by special events (like power up or touching touchable elements) or by other macros or by macro commands received via the RS232 port.

To implement the display processor tasks a makro structure has been set up in the macro file *epg2.kmc* (see appendix D). At power up the macro INIT performs an initialization sequence (displaying the *ipds* logo and an initialization text, setting up screen elements common to all modes, defining touch screen buttons and selecting the default mode ‚Display‘). Delays are introduced to ensure that the coder processor initialization (see 4.3) is done when the display processor sends the mode code for ‚Display‘ to finish the initialization, thus synchronizing both processors. The coder processor acknowledges this by calling DIS\_ACK to set up the ‚Display‘ screen.

Each mode is maintained until one of the three touch buttons is pressed. The appropriate mode code is sent to the coder processor which acknowledges this by calling the appropriate macro (DIS\_ACK, CAL\_ACK or COD\_ACK) to set up a new screen.

During the mode ‚Display‘ MUX data are displayed by placing special characters at the appropriate coordinates within the palatogram area. A special font has been defined (pal.fxt) and included into the macro file as font #8. It contains only two characters: a small filled circle and a big filled circle. Screen commands are sent by the coder processor to place these characters (see appendix C-4.1).

In the modes ‚Calibration‘ and ‚Coding‘ touchable bar graphs („sliders“) are set up to control the amplitude of the codec output signals. During ‚Calibration‘ only the EPG slider may be repositioned, during ‚Coding‘ only the MIC slider. The sliders disabled are ‚greyed out‘ indicating that any repositioning is reported to the coder processor but not accepted. The acknowledgement macro resets the slider to its previous value (see appendices C-4.2 and C-4.3).

### 4.3 Coder Processor Software

The coder processor software is completely written in ,C' with some extensions for handling port pins of embedded microcontrollers. This requires a special development environment (see 5.2). The ,C' source code is found in the file *epg2.c* (see appendix D). The binary is downloaded offline into the processor's flash-ROM (see 5.2).

Appendix B-3 shows the program structure. The program consists of five blocks:

- an initialization sequence,
- three parts to accomplish the three major operation modes (see 4.1),
- an interrupt routine running in parallel to the mode blocks.

#### 4.3.1 Initialization

The initialization part of *main()* sets up hardware and software, before an endless loop is entered to perform the three mode tasks. The following tasks are done during this sequence:

- set up data directions of the processor pins (see appendix A-3.2.2),
- set up the RS232 link to the display processor (9600Bd 8N1),
- initialize the test signals for an oscilloscope,
- initialize timer T6  
(this timer drives the interrupt routine, together they establish the internal timing of the EPG2 device),
- setup and enable timer T6 interrupt  
(the timer is still stopped!),
- reset and initialize the display processor,
- compute sine tables (see 4.3.4),
- compute calibration data (see 4.3.5),
- clear signal buffers to prevent clicks at the output,
- wait some time (with blinking LEDs) until internal initialization of the codec chip AD1845 is done,
- do programmed initialization of the codec chip  
(*init\_codec()* is a very clumsy subroutine; do not change sequence or settings unless you know exactly what you are doing!! For more information see the AD1845 data sheet (see 3.4.2)),
- start timer T6  
(now the interrupt routine *timer6()* is running in parallel to *main()*),
- set default output signal attenuation (MICOUT and PALOUT).

At this time *main()* waits until the initialization sequence of the display processor is finished and the first character is sent. This should be ,3' for the default mode ,Display'. *main()* now enters an endless loop. First, the battery is checked, then *switch(mode)* selects one of three mode modules.

#### 4.3.2 Interrupt subroutine

The interrupt subroutine *timer6()* runs in parallel to *main()* as soon as T6 is started. It establishes the internal timing of the EPG2 device. Each time the interrupt routine is entered *intr\_count* is incremented (from the initial value 0). *switch(intr\_count)* selects one of 28 cases each with a specific sequence of actions. At the end of each sequence, if necessary the *timer6* time constant *CAPREL* is set to a new value valid after the next interrupt. In case 28 *intr\_count* and *CAPREL* are set to their respective initial values.

Thus, a timing structure with 28 cycles of different duration is set up. The interrupt routine is active at the very begin of the cycle, leaving the rest of the cycle to *main()*. The cycle durations sum up to 10 msec, the frame duration.

The activity of the interrupt routine and the main program may be monitored with a dual trace oscilloscope connecting the coder signal SYNC to the trigger input and ICHECK (interrupt routine active) and MCHECK (main active) to the trace inputs.

The following table shows the duration of each cycle and the action of the interrupt routine at the beginning of the cycle:

cycle#	duration [us]	action of interrupt routine
1	100	toggle ‚ping pong‘ output buffer; get one char. from serial input, if present; switch injection generator on; set MUX address to 1 (right MUX); set next cycle duration to 50us; output 16 samples from previous frame; input 7bit of MUX data; increment MUX address;
2,3,4	3x50	input 7bit of MUX data; increment MUX address;
5	50	input 7bit of MUX data; set MUX address to 9 (left MUX);
6,7,8	3x50	input 7bit of MUX data; increment MUX address;
9	50	input 7bit of MUX data; increment MUX address; set next cycle duration to 500us;
10	500	output 10 samples from previous frame; input 7bit of MUX data; set MUX address to 0 (idle state); switch injection generator off; get one char. from serial input, if present;
11...27	17x500	output 10 samples from previous frame; get one char. from serial input, if present;
28	500	output 4 samples from previous frame; get one char. from serial input, if present; set next cycle duration to 100us; set interrupt counter to 0
sum:	10.000	-> 10 ms frame duration

summary of interrupt routine actions:

- at the start of the frame 10x7bit of MUX data are input,
- every 500us 10 samples are output to the codec; the codec has a 16 sample deep FIFO, so the frame starts with 16 samples to prevent any underrun, and ends with 4 samples; the sample rate of the codec is set to 20kHz, within one frame the codec has to be fed with 200 samples,
- every 500us the serial input is checked, if a character is present it is put into the character buffer.

#### 4.3.3 Main Program: Mode ‚Display‘

This branch of main() runs unsynchronized to the interrupt routine. It just picks 10x7bit from the MUX data buffer (paldata[]). For each bit, it sends a command to display a small or a big circle (see 4.2) at the appropriate coordinates of the screen. The coordinates (hor, ver) are computed from the predefined array contact[][][]]. It contains up to four contact maps for different pseudo palate designs. If a pseudo palate other than the ipds design shall be used (e.g. a ‚Reading‘ design) this may be done by connecting the artificial palate via an appropriate adapter and by setting up an appropriate map (see appendix B-5.2). The selection of a contact map is controlled by the setting of jumper J1 on the coder PCB (see 3.4) The output to the display is comparatively slow, not every frame of MUX data is caught. This does not matter, as neither the LCD is fast enough to display such rapid changes, nor, if the display would be fast enough, is the human eye able to catch up with these changes. Thus, no synchronization is necessary. For an impressionistic overall view the display is more than sufficient.

If something is received from the serial port (this can be only a mode code) the mode ‚display‘ is left immediately.

#### 4.3.4 Main Program : Mode ‚Coding‘

This branch of main() computes the coded EPG signal from the MUX data. The EPG signal consists of 11 sine components, 10 components each coding 7 bit of MUX data and a fixed reference component. The lower two bit of a MUX ‚byte‘ select one of four possible frequencies, the upper five bits select one of 32 amplitudes. The reference component has a fixed frequency and maximum amplitude. Thus sine components with 41 frequencies may occur (see appendix B-5.1). They are computed from the same sine table. To avoid time consuming

multiplication during the real time coding process 32 sine tables are computed during initialization, one for each amplitude, and another one for the reference component (see 4.3.1).

As the ,Coding‘ branch uses MUX data acquired at the beginning of the frame, synchronization to the first cycle of the frame is necessary. From then on the routine is free running, but it is essential, that computation of the coded signal for that frame has been accomplished before the next frame starts.

The signal samples are stored into a ping-pong buffer (sample0[] or sample1[]) with the actual output to the codec done by the interrupt routine during the next frame (see 4.3.2). An additional delay of approx. 600us is introduced by the codec sound chip. Thus, the signal for a frame starts 10,6 ms after the actual frame start.

The signal for a frame of 10ms (200 samples @ 20kHz sample rate) consists of 8 ms (160 samples) compound signal with 11 sine components and 2 ms (40 samples) pause. The pauses are used by the decoder software to detect frame boundaries. The compound signal is the sum of the reference signal and 10 ,generator‘ signals . The term ,generator‘ is historic, from an earlier model of the EPG2 device, where the signal components have been produced by hardware sine generators. Here, everything is done by software.

Any input from the touch screen display is handled by the subroutine disp\_in(). This includes mode switching and readjustment of the of the sliders EPG and MIC.

#### 4.3.5 Main Program : Mode ,Calibration‘

This branch of main() computes a cyclic fixed calibration signal of 64 frames which is used for two purposes:

- it is used by the software decoder to account for the overall frequency response of the system ,
- it is used to set the gain of the codec and the sound card.

The structure of this branch is very similar to that of ,Coding‘. Instead of the (variable) MUX data, a fixed set of data from cal\_data[][] is used. This array is set setup during initialization (see 4.3.1). The resulting signal consists of sine components with selected combinations of frequencies all at maximum amplitude. This signal has the highest possible amplitude and should thus be used to set the gain of codec and sound card to a level yielding the best S/N ratio while avoiding signal clipping (see 2.2).

## 5. Device Programming

### 5.1 Display

To download the required screen macros into the display processor's flash-ROM you need the following items:

- a DOS- or WINDOWS-based PC with serial port,
- the contents of the disk EA DISK320 (available from the manufacturer of the display, see 3.4) installed on the PC,
- your screen macro file (e.g. *epg2.kmc*) on the PC,
- any include files (font files, picture files) specified by the screen macro file on the PC,
- an adapter cable EA KV24-9B (available from the manufacturer of the display, see 3.4),
- a power supply 5VDC, 1.5A.

To download proceed as follows:

- disconnect the display from the EPG2 device,
- connect the 5VDC power supply to the screw terminals of the display,
- connect the adapter cable to the display and to the PC's serial port,
- power up the display,
- edit the file *c.bat* to the appropriate path to your screen macro file,
- invoke *c.bat*
  - the following sequence now runs automatically:
    - compilation of the screen macros,
    - check of the environment,
    - connection to the display,
    - download to the display ( progress is shown on the display),
    - disconnection and restart of the display.
- power down the display,
- disconnect power supply and the adapter cable,
- reconnect the the display to the EPG2 device,
- power up the EPG2 device to check the result.

Any changes to the screen macro file are easily done with any ASCII editor (like DOS *edit* or WINDOWS *notepad*).

## 5.2 Coder

To download a binary file into the coder processor's flash-ROM you need the following items:

- a DOS- or WINDOWS-based PC with serial port,
- a flash tool like *flasht.exe* (available from the manufacturer of the processor, see 3.4.1, also on the data disk, see appendix D) on the PC,
- your binary file (e.g. *epg2.h86*) on the PC,
- an appropriate adapter cable to connect the main board's RSC232 pins to the PC's serial port,
- two push-button switches to connect to the Res and Prg pins of the coder PCB.

To download proceed as follows:

- connect the EPG2 device's RS232 pins to the serial port of the PC,
- connect RESET and PROG push-button switches to the coder PCB (see appendix C-3.2),
- power up the EPG2 device,
- set the RUN/PROG switch on the main PCB to PROG
- press and hold the PROG push-button,
- press the RESET push-button for 5 seconds, then release,
- release the PROG push-button,
  - the coder LEDs are off and stay so
- invoke *flasht.exe*
  - flasht* displays ,Loading Flash-Utilities'
- at Command: select ,7', then ,Yes',
  - flasht* shows the erasure of the flash-ROM
- press F2, then give the complete path of your binary file (e.g. *d:\epg2\epg2.h86*),
  - flasht* displays the contents of the binary file downloaded (INTEL Hex format)
  - flasht* displays ,Executing Software-Reset'
  - the coder LEDs start blinking, then stay on green ( the coder has been restarted)
- set the RUN/PROG switch on the main PCB to run (to connect the display),
- press and release the RESET push-button,
  - the EPG2 device should startup as usual (or something is wrong with the binary file! ☹)
- if o.k., disconnect push-buttons and adapter cable.

If you want to modify the coder program, you need a program development environment (compiler, assembler, linker, etc). We started with uVision166 which came with the ,Rapid Development Kit' available from the manufacturer of the coder processor module (see 3.4.1). Unfortunately, this program is limited to produce binary code only up to 4kByte. As the size of the code for the EPG2 is now above this margin, it is necessary to use the ,big' (and expensive!) environments for c16x family processors available e.g. from Keil ([www.keil.com](http://www.keil.com)) or Tasking ([www.tasking.com](http://www.tasking.com)). We now use Keil uVision 2.

For recompilation of the coder program the following requirements have to be met, otherwise program execution may be too slow to do the real time computations of the modes ,Coding' and ,Calibration' within the frame duration of 10 ms:

- set the development environment to
  - memory model ,small', DPPuse, 16kB RAM, 16kB ROM (pointers to peripherals are explicitly declared as ,huge')
  - compiler optimization: constant propagation, execution speed,
- use the startup file *st\_epg2.a66* as supplied on the data disk (see appendix D), it is speed optimized for use with the coder hardware.

To check whether the program execution meets the real time requirements use an oscilloscope as described in 4.3.2.