Anaesthetic Record System 5a

revision 2014b
www.nickalls.org/dick/papers/xenon/ARS5aDOC2014b.pdf

Richard W. D. Nickalls
March 2014

The single biggest problem we face is that of visualisation

Richard P. Feynman (1918–1988) [1]

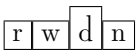
[1] The Mathematical Gazette (1996); 80, 267.

Anaesthesia Record System 5a

Richard W. D. Nickalls,

Department of Anaesthesia,
Nottingham University Hospitals,
City Hospital Campus,
Nottingham, UK.

dick@nickalls.org
http://www.nickalls.org/



March 2014
revision 2014b

Anaesthesia Record System 5a

Preface

This document brings together in one place a fairly comprehensive documentation of the development work, files, programs and some screenshots relating to my original MS-DOS prototype of the Anaesthetic Workstation computer program (written in PowerBasic) which was used in one of the thoracic operating theatres during the period 1994–2001. This revision includes a new chapter describing the Datex simulator program we used to send Datex AS/3 data to the main program (chapter 4).

After 2001 this computer program was rewritten and developed for the Linux operating system using C and Perl (detailed in a separate document on this website: `www.nickalls.org/dick/papers/xenon/xenon2009a.pdf`).

RWD Nickalls
March 2014

Contents

Part I

Screenshots

Screenshots (1997–2001)

The following screenshots are of the prototype MS-DOS (PowerBasic) version of the Anaesthesia Record program which we used until approximately 2002, when it was replaced by a more sophisticated upgrade which was written for the Linux platform.

The first four screenshots show the front-end menu system and detail of the menus. The remaining three screenshots show screens taken during operations.

Figure 1:
(Date 2001: prog vers 5a-dj501): This screen shows the "`run programs`" menu, from which one could either access the data from the Datex AS/3 monitor (during the operation), or BMI information, or the Epidural database program (estimated likely epidural depth for individual patients by accessing the database).

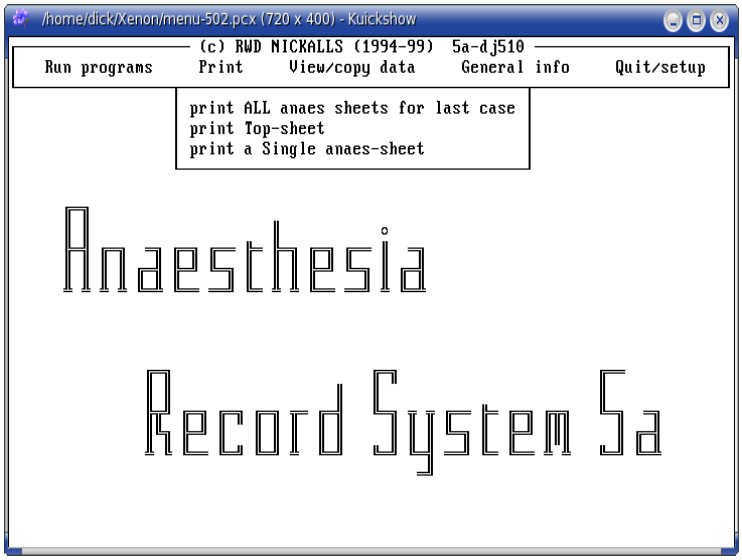


Figure 2:

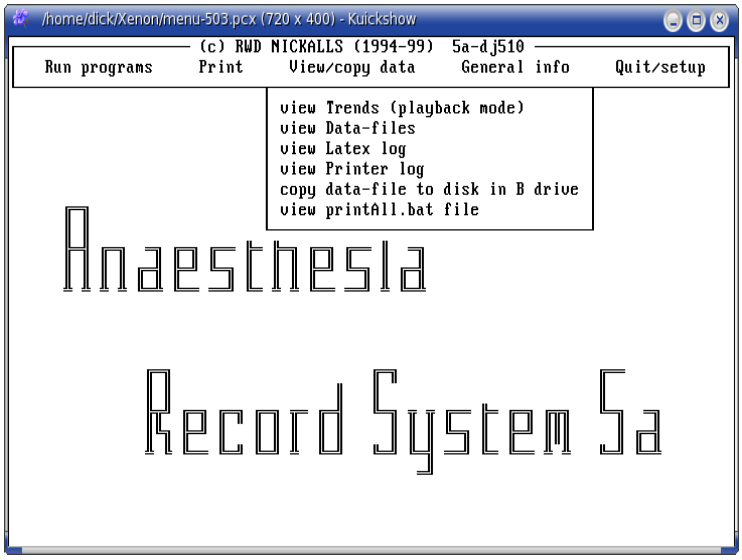


Figure 3:

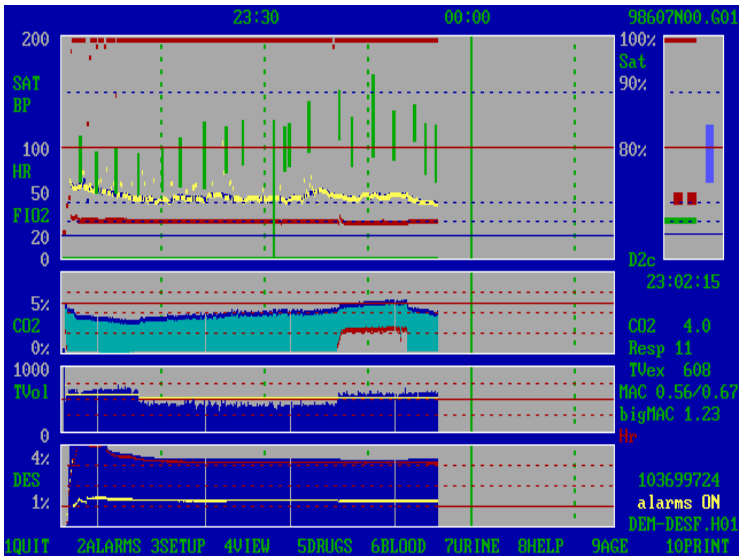


Figure 4:
June 1998: Screen in 'playback' mode displaying data from a previous operation (from file `DEM-DESF.H01`). The screen shows four horizontal 'trend data' graphic windows, one vertical 'NOW' window (top right) displaying the latest data (data is accessed every 5 seconds from the Datex AS/3 monitor), and a text area on the right (below the 'NOW' window) giving (in green) the time, current values of some respiratory, MAC & age-corrected MAC ('bigMAC') values, and (in red) parameters in an 'alarm' state (Hr in this case). There are 10 F-buttons displayed along the bottom of the screen.

(a) NOW window (top right): shows sat (red), BP (blue), HR from ECG and oximeter (red: 'low' alarm state), FIO_2 (green).
(b) TREND BP window (top): Displays the trend data from the NOW window. NIBP (green), HR-ECG (blue), HR-oxim (Yellow), FIO_2 (red).
(c) TREND CO_2 window (center): Displays $ETCO_2$ (dark blue), and inspired CO2 (red). Note here we can readily see that the circle CO_2 absorber was switched off for a short period (to replace the soda-lime) during which time the inspired CO_2 rose to approximately 2%.
(c) TREND Tidal volume window (center): Displays tidal volume (dark blue) and Resp rate (yellow)
(c) TREND Vapour window (bottom): Displays Inspired agent concentration (dark blue), ET agent concentration (red), age corrected MAC (yellow).

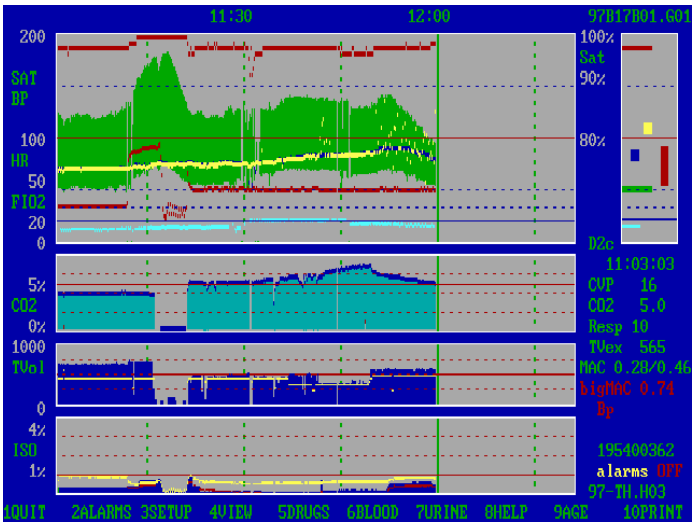


Figure 5:
November 1997: Shows that the BP is continuous (arterial line). There is a period of about 5 minutes during which there is no gas sampling (the patient is bagged by hand on 100% oxygen while fixing a double-lumen tube position problem), and sat is 100% during this time.

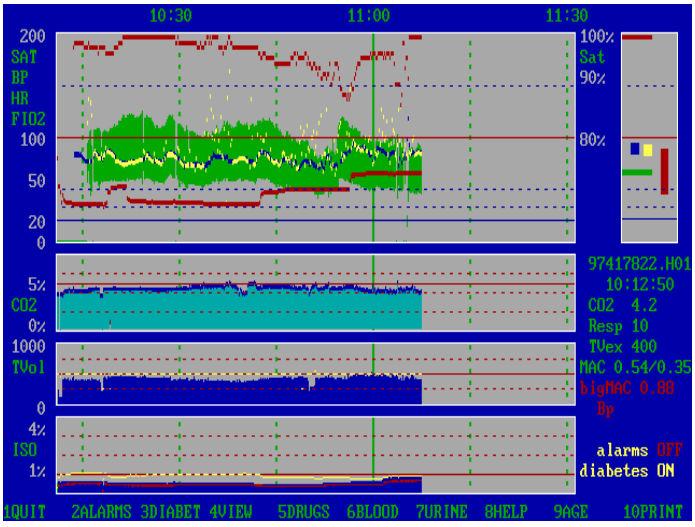


Figure 6:
April 1997: Showing a period of hypotension and low age-corrected MAC. Both BP and ‘bigMAC’ are highlighted in red in the text ‘alarm’ area, and BP is red in the NOW window.

Part II

Background

Chapter 1

An automated anaesthesia record-keeping & trending system

ch-hist

1.1 Historical background

Effective surgical anaesthesia was established in 1846 following the discovery of the effects of inhaled diethyl-ether ("*ether*"). Although Dr John Snow[1] (1813–1858), Joseph Clover (1825–1882), and Mounier (1855) demonstrated the importance of monitoring the pulse and respiration during anaesthesia (Ellis, 1995; Rushman, Davies and Atkinson, 1996) it was not until 1894 at the Massachusetts General Hospital, Boston, that surgeons Earnest Codman (1869–1940) and Harvey Cushing (1869–1939) established the practice of keeping a careful *written* record (on graph paper) of the patient's pulse and respiration rate during operations (Beecher, 1940; Hirsch and Smith, 1986). In 1901 they started including measurements of the arterial blood pressure using the newly described apparatus of Scipione Riva-Rocci (1863–1937) of Turin (Cushing, 1902, 1903; Rushman, Davies and Atkinson, 1996). Another pioneer of written anaesthesia records was Ralph Waters (1936). McKessons's *Nargraf* machine of 1930 could produce a semi-automated record of inspired oxygen, tidal volume and inspiratory gas pressure (Westhorpe, 1989). Later Noseworthy (1945) produced special cards on which to record anaesthetic details (see Rushman, Davies and Atkinson (1996), p 111, for an illustration).

Since then little of real significance was developed in the area of anaesthesia monitoring until the 1970s, when advances in chip technology gave rise to clinically useful portable electronic devices for measuring such things as arterial and central venous blood pressure, breath-by-breath concentrations of oxygen, carbon dioxide and inhalational anaesthetics, pulse oximetry, and of course, small computers.

[1] A London physician, and the first full-time professional anaesthetist.

1.2 The anaesthesia record

Until the last few years, the anaesthesia record was exclusively a hand-written record, consisting of the various measured parameters, as well as what drugs and fluids are given, the blood loss, the urine output, adverse reactions, difficulties encountered, and so on.

From a monitoring point of view, a very significant and far reaching feature was incorporated into virtually all electronic patient monitoring devices, namely a specialised communications interface known as the RS-232 serial port. Significantly, the same RS-232 serial port was also incorporated into all IBM-compatible PCs. Consequently it is now a relatively easy matter to use a PC to access the measured parameters output by the various patient monitoring devices, and anaesthetists are increasingly exploring methods for automating data collection.

1.2.1 Automated anaesthesia record keeping

The value of computers for anaesthetists in the operating theatre is that they allow excellent trend display of measured data, real-time calculation of various derived parameters, as well as freeing the anaesthetists from the work of documenting drug doses and measured parameters. Automated anaesthesia record keeping systems therefore offer the great advantage of allowing the anaesthetists to concentrate fully on the patient, leading to enhanced vigilance and improved patient care and safety.

An automated anaesthesia record chart is significantly superior to the usual hand-written record, since it samples data more frequently and more accurately, and hence it has certain medico-legal advantages regarding the documentation of patient care, particularly during complicated and/or unstable cases.

1.3 Background to this project

Since 1994 I have run an on-going project to automate the production of the anaesthetic record in the operating theatre, and to improve information display as well as the calculation of various so-called *value-added* real-time parameters (e.g. age-dependent MAC). Aspects of this project have been presented in various presentations and publications (Nickalls & Ramasubramanian, 1995; Nickalls 1996, 1997, 1998a,b,c).

This project has led to the development of a practical working PC-based prototype which is currently in use in the City Hospital Operating Theatres. It yields a continuous trend display on the PC of a variety of measured and derived parameters, and also automatically prints out sequential 1-hour trend-data in a graphic form suitable to be placed directly into the patients notes as a final record. A book on interfacing PCs to medical equipment (Nickalls & Ramasubramanian, 1995) based on the development work relating to this project, was published by Cambridge University Press.

1.4 Collaboration with Leicester University

The next phase is (a) to port the program to the Linux operating system, and (b) to extend the software to access the hospital HIS system, and also into the area of decision-support in anaesthesia, and to explore the use of predictive mathematical modeling both in physiology and pharmacology. This is an area of great potential, and hopefully will be of interest to biomedical scientists.

To this end, I have formed a collaboration with the Department of Electronic & Software Engineering at the University of Leicester. In conjunction with Dr Michael Pont (Senior Lecturer) and a number of his final-year students, we plan over the next two years to port the program to the Linux operating system, and to write a separate module to interface with the City Hospital HIS system (access patient data, and also store anaesthesia data). The aim is to develop a free 'open-source' modular program for use by anaesthetists in NHS hospitals.

References

- Beecher HK (1940). The first anesthesia records (Codman, Cushing). *Surg. Gynecol. Obstet.*, 71, 689–693.
- Cushing HW (1902). On the avoidance of shock in major amputation by cocainization of large nerve trunks preliminary to their division. With observations on blood pressure changes in surgical cases. *Annals of Surgery*, 36, 321–345. [from Hirsch & Smith (1986)]
- Cushing HW (1903).On routine determinations of arterial tension in operating rooms and clinic. *Boston Med. Surg. Journal*, 148, 250–256. [from Hirsch & Smith (1986)]
  [reproduced in 'Classical File', *Survey of Anesthesiology*, 1960; 4, 419] [from Rushman, Davies and Atkinson (1996)]
- Ellis RH (1995). *The Casebooks of Dr John Snow*, Wellcome Institute for the History of Medicine; p 22, p 30.
- Fulton JF (1946). Harvey Cushing—a biography. (Charles C Thomas, Springfield, IL, USA).
- Hirsch NP and Smith GB (1986). Harvey Cushing: his contribution to anesthesia. *Anesthesia and Analgesia*; 65, 288–293.
- Kenny GNC [ed] (1990). Automated anaesthesia records. *Bailliere's Clinical Anaesthesiology*; 4, June.
- Middleton H (1957). *Proc Roy Soc Med*; 50, 888. [from Middleton (1958a)]
- Middleton H (1958a). A cumulative anaesthesia record system. *Anaesthesia*; 13, 337–340.
- Middleton H (1958b). *Brit med Bull*; 14, 42. [from Middleton (1958a)
- Mounier CCR (1855). *Acad. Sci. Paris*, vol(40), p 530. [from Rushman, Davies and Atkinson (1996)]
- Nickalls, RWD and Ramasubramanian R. (1995).
  Interfacing the IBM-PC to medical equipment: the art of serial communication.
  ISBN 0-521-46280-0; pp 402. (Cambridge University Press).
- Nickalls RWD (1996).
  An automated Anaesthesia Record System using free text-based software.
  Oral presentation to the *16th International Symposium on Monitoring and Computing in Anaesthesia and Intensive Care* in Rotterdam, Holland (May 1996).

- Nickalls RWD (1997).
  An Anaesthesia Record-keeping System using free text-based software.
  SCATA News, 6(1), 6.
  Abstract of a presentation to the *Society for Computing and Technology in Anaesthesia* (SCATA). Glasgow, UK; November 21–22, 1996.

- Nickalls RWD (1998a).
  TeX in the operating theatre: an Anaesthesia application.
  TUGboat; 19, Proceedings of the 19th International TEX Users Group Meeting; p 7–9.
  (Toruń, Poland, August 17–20, 1998)

- Nickalls RWD (1998b).
  TeX in the operating theatre: an Anaesthesia application.
  Invited presentation to the *Annual UK TEX Users Group meeting*, Cambridge, UK.
  (September 21–22, 1998).

- Nickalls RWD (1998c).
  Automated data capture—the doctor's view.
  Invited talk at an industry workshop on *The Medical Information Bus (MIB)*.[2]
  Royal Angus Hotel, Birmingham, UK, (June 17, 1998).
  Organised by LinkTech Incorporated.

- Noseworthy M (1937). *St. Thomas's Hospital Reports (London)*; 2, 54. [from Rushman, Davies and Atkinson (1996)]

- Noseworthy M (1943). *British Journal of Anaesthesia*; 18, 4 (? p 160). [from Oldham (1963); Middleton (1958)]

- Noseworthy M (1945). *Anesthesia and Analgesia*; 24, 221. [from Rushman, Davies and Atkinson (1996)]

- Noseworthy M (1953). *Anaesthesia*; 8, 43. [from Noseworthy (1963)]

- Noseworthy M (1963). Anaesthetic record card. *Anaesthesia*; 18, 209–212.

- Oldham KW (1963). Anaesthetic and operation records. *Anaesthesia*; 18, 213–216.

- Rushman GB, Davies NJH and Atkinson RS (1996). *A short history of anaesthesia: the first 150 years*. (Butterworth-Heinmann, Oxford, UK). [see chapter 14; Monitoring, p 154–161]

- Waters RM (1936). *Indiana St. Med. Assoc.*, 29, 110. [from Rushman, Davies and Atkinson (1996)]

- Westhorpe R (1989). McKesson 'Nargraf' anaesthesic record. *Anaesthesia and Intensive Care*; 17, 250.

[2] A meeting concerned with the recent IEEE-1073 Standard regarding computer interfacing to Medical Devices.

Chapter 2

TEX in the Operating Theatre: an Anaesthesia application.

R. W. D. Nickalls BSc, PhD, MBBS, FRCA.
Consultant in Anaesthesia & Intensive Care,
Department of Anaesthesia,
City Hospital, Nottingham, UK.
dick@nickalls.org

Abstract

This article describes the author's experience of using TEX for typesetting the *Anaesthesia Record* as part of an automated data-collection system developed for use in the operating theatre.
TUGboat; 19(3), Proceedings of the 19th International TEX Users Group meeting, Toruń, Poland, August 17–20, 1998; pp. 7–9.
http://www.tug.org/TUGboat/Articles/tb19-3/tb60nick.pdf

Introduction

Since the theme of this year's conference is "*Integrating TEX with the surrounding world*" I would like to describe my integration of TEX with the world of the operating theatre—specifically with the domain of anaesthesia.

One of the many things that occupies anaesthetists during an operation is documentation. This takes the form of a log of various physiological parameters (see Figure 1), drugs used, blood lost, fluids administered, procedures performed etc., otherwise known as the *Anaesthesia Record*. Since this is generally a handwritten record, the documentation side of things can become rather neglected during busy periods, and consequently, anaesthetists are increasingly using computers to automate the collection of such data. This has many advantages including allowing real-time processing of data, generation of various derived parameters, and greatly enhanced information display facilities.

Collecting and processing the data

Since most monitoring equipment used in Critical Care environments has an RS-232 serial interface the process of data-collection, construction of trend graphics, formatting and typesetting can be automated reasonably easily.

My own system is a menu-driven research application which uses compiled QuickBASIC programs to coordinate the

access, display and printing of both real-time physiological data and keyboard inputs. The printing module uses LaTeX to typeset the text and graphics to create the *Anaesthesia Record* in a format suited to the hospital notes.

The data from the various anaesthesia monitors is accessed via the serial port using a multiplexing device. Individual parameters are then extracted using the relevant software for each of the various monitors—see [1] for interfacing details relating to particular anaesthesia monitors. Unfortunately there is currently no standardisation with regard to data formats for medical monitoring devices, but this may well soon change with the development of the new international Medical Information Bus (MIB) standard (IEEE 1073).

During anaesthesia the program accesses and displays all the data in real-time as graphic trends, as well as deriving a number of so-called 'value-added' parameters and processing keyboard entries. At the end of the operation the program typesets the text and graphics to form the *Anaesthesia Record*.

The graphics are created using the excellent *freeware* program GNUPLOT[1] which allows batch processing and will output graphics in LaTeX picture format.

Armed with the maximum and minimum values for each of the measured parameters, the program writes the GNUPLOT input files, and then calls GNUPLOT, outputting the graphics in LaTeX picture format, and placing them into the appropriate directories. The program then writes the LaTeX input `.tex` file, and then calls LaTeX to typeset the text and graphics. Finally the `.dvi` file is printed and put into the hospital notes. In practice all this is performed locally within the operating theatre, such that the *Anaesthesia Record* is printed and placed in the patient notes just as the patient is returned to the recovery area. Figure 1 shows the graphics page of a typical *Anaesthesia Record*.

Advantages of ASCII-based systems

The fact that both TEX and GNUPLOT use inputs which are ASCII-based has the great advantage that their input files can be written on-the-fly by the coordinating computer program. Such flexibility allows the final text and graphics of the document to be tailored to the data. For example, this allows the axes of graphs to be automatically adjusted depending on maximum and minimum values. Similarly, text layout can be made to vary depending on the particular keyboard entries made during the operation.

Small is beautiful

An automated system for data collection, display and printing has clear advantages over the usual hand-written method; it is certainly a more accurate record, and physiological data can be sampled much more frequently. Furthermore, keyboard entry of drugs and other information can be made simple and fast by careful design of the interface.

Since this is a specific stand-alone application, it is possible to use a much cut-down version of LaTeX consisting only of the essential files, fonts and style options required for the application, with the effect that the size of the printing module can be made extremely small. A not insignificant bonus, therefore, of using TEX as the typesetting engine is that I am able to make use of old 386 PCs having relatively small hard-drives, which have been discarded by my memory-hungry colleagues!

References

1. Nickalls RWD and Ramasubramanian R (1995). *Interfacing the IBM-PC to medical equipment; the art of serial communication.* Cambridge University Press, Cambridge, UK. pp 402. ISBN: 0 521 46280 0

[1] `http://www.cs.dartmouth.edu/gnuplot_info.html`

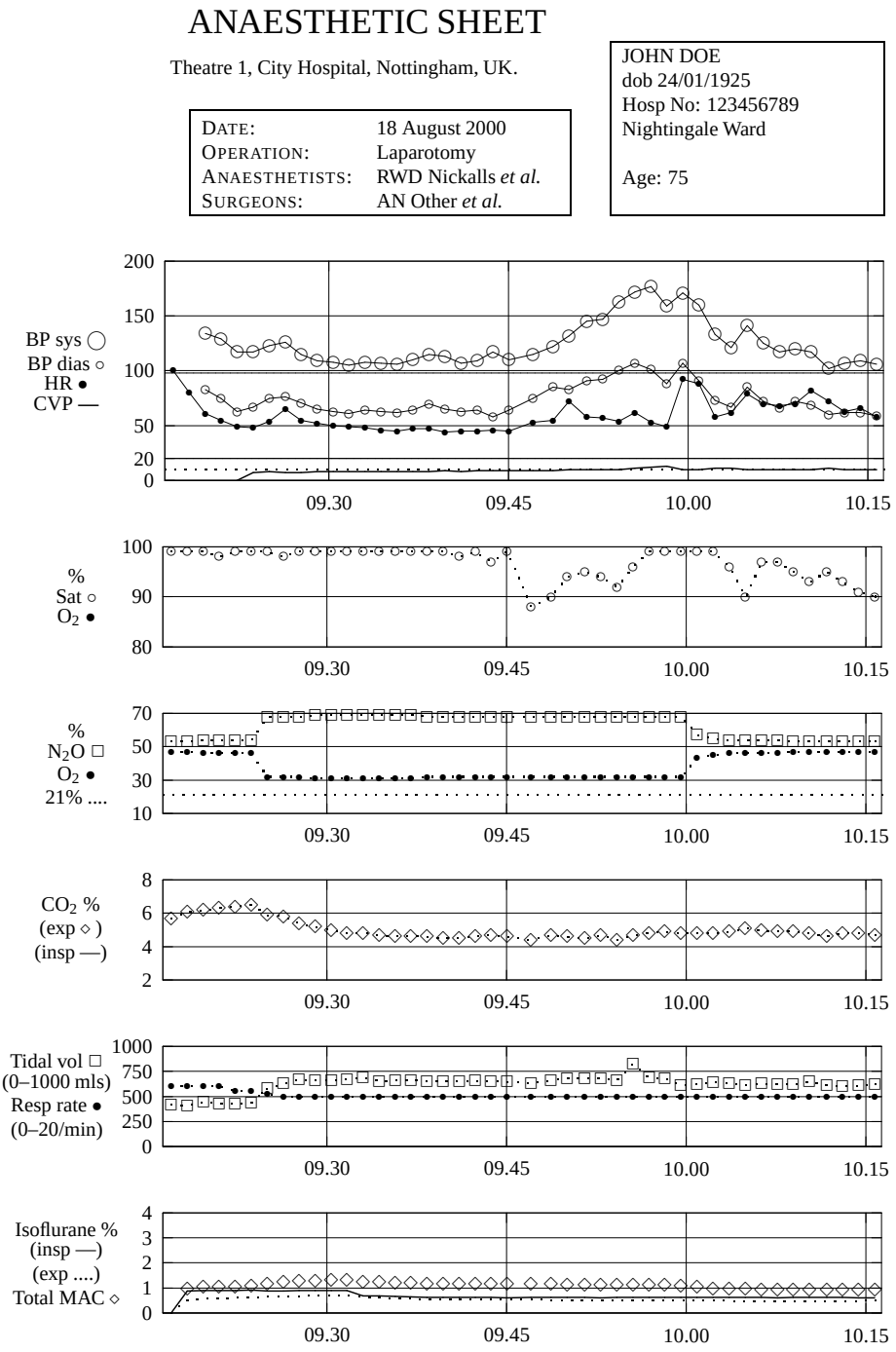


Figure 2.1: Example of the graphics section of a typical *Anaesthesia Record*. The six graphs are output by GNUPLOT in LaTeX picture format. The record shows blood pressure (BP), heart rate (HR), central venous pressure (CVP), oxygen saturation of haemoglobin (Sat), inspired oxygen (O_2), inspired nitrous oxide (N_2O), expired carbon dioxide (CO_2), tidal volume, respiration rate, isoflurane and MAC.

Chapter 3

The Datex AS/3 anaesthesia monitor

ch-dxmon.tex

3.1 Introduction

The Datex-Ohmeda[1] AS/3 and CS/3 monitors are versatile modular anaesthesia monitoring systems, which have an asynchronous serial interface for data acquisition. The various modules access a comprehenensive range of physiological parameters. Note that the technical latest manual regarding the serial interface is *AS/3 and CS/3 Monitor Product specification—computer information. v.3.4* March 1999 (G-version update by Rene Coffeng, 23/Nov/1998).

The electrical safety Type classifications of the various Applied Parts (e.g. NIBP cuff, temperature probe) are shown in Table 3.1.

Table 3.1: Applied Parts and their Types.

Applied Part	Type
ECG	CF
NIBP	BF
Invasive BP/CVP/PA	CF
Temperature probe	CF
Cardiac output	?

[1] Datex-Ohmeda Division, Instrumentarium Corp., P. O. Box 900, FIN-00031 Datex-Engstrom, Finland. Tel: +358–9–39411; FAX: +358–9–146–3310.
Datex-Ohmeda, 71, Great North Road, Hatfield, Hertfordshire, AL9-5EN, UK; Tel: 01707-263-570, FAX 01707-260-065.

3.1.1 Software version

Software is frequently revised, and different monitors may have different software versions. The software version is displayed on the screen when the monitor is switched on, and is also indicated as a 1-byte code (the 5^{th} byte) in the 40-byte 'header' which precedes all data output via the serial port. The 1-byte software version codes are shown in Table 3.2.

Table 3.2: Software versions and their *Datex Read Interface* codes (`r_dri_level`).

Software version	code
S-STD93	0
S-STD94, S-ARK94	1
S-STD95, S-ARK95, S-STD96, S-ARK96	2
S-ANE97, S-ARK97, S-ICU97	3

3.1.2 Available software

A program for PCs called COLLECT.EXE, which saves data from the Datex AS/3 monitor, is available from Datex. This program is known as the *AS/3 PC Data Collection Software*. The program collects the data-strings output by the monitor and saves them to the hard disk of the PC either as an ASCII file, a binary file, or in a form compatible with LOTUS 1-2-3. The package consistes of three program files as follows.

COLLECT.EXE
COLLECT.CFR (used for storing setup information)
AUTOFILE.CFR (used for writing an automatic date-dependent filename

3.2 Serial port

The monitors have a male 9-pin D-type serial port which conforms to the RS-232-E standard. The serial port is located at the back of the monitor.

The serial port allows commands to be sent to the monitor, and also allows CTS/RTS flow control (hardware handshaking) via pins 7 and 8 of the serial port.

Table 3.3: Datex AS/3 & CS/3 RS-232 serial port.

Pin No.	Name	Comments
2	RxD	Receives data
3	TxD	Transmits data (LOW on power-up)
5	GND	Signal ground
7	RTS	Set HIGH when powered up
8	CTS	Can be used to control data flow

3.2.1 Cable connections

The wiring configuration for interfacing the Datex AS/3 monitor to a PC is shown in Figure 3.1.

- **CTS** [NB: not fully checked out for AS/3] Data output from the Datex monitor is usually controlled by influencing the voltage status of the Datex-AS3 CTS line. Data output is enabled only if its CTS is held HIGH (positive). [BUT in my experience data output was only stopped by setting the datex RTS line LOW !!]

  However, if it is necessary to use hardware handshaking to control data output, then it is probably best to connect the Datex monitor's CTS line to the computer's RTS line, which can then be used to control data output by setting the status of the computer's RTS line HIGH or LOW as necessary (see Section 5.16 in *Nickalls & Ramasubramanian, 1995*).

- **RTS** The Datex-AS3's RTS line is held HIGH on power up. Holding the Datex RTS line LOW will stop all data output until it is pulled HIGH again.

  Consequently it is usual to connect this line to the computer's CTS line, to enable the computer program to control data output from the Datex monitor.

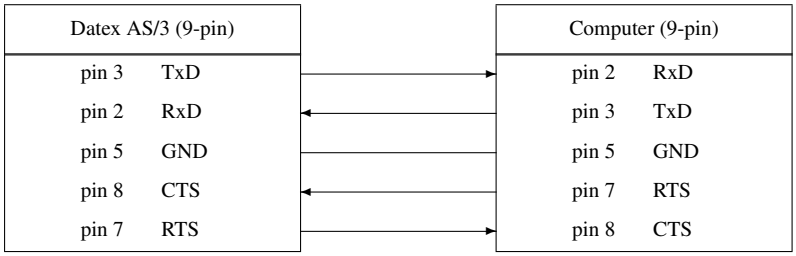


Figure 3.1: Wiring configuration for the Datex AS/3 & CS/3 monitors.

3.2.2 Protocol

The serial protocol is shown in Table 3.4. Note that this protocol is slightly unusual in that it uses an 11-bit character-frame (1 start-bit, 8 data-bits, EVEN parity-bit, 1 stop-bit). Consequently some older software which uses a ten-bit character frame (e.g. QuickBASIC 4.5, QBASIC 1.1) cannot be used to program the Datex-AS3 serial interface. PowerBASIC 3.5, FirstBASIC (PB1.0) and VisualBASIC can all handle 11-bit character-frames. Note that the recent 3.15 version of KERMIT (1998) also accomodates the 11-bit character frame (see the `SET PARITY HARDWARE` command), and so can be used to access data from the Datex AS/3 monitor.

3.3 Command format

The monitor is able to output data in a number of modes; either (a) only the current displayed measurement values; (b) values averaged over the last 10 seconds, (c) values averaged over the last 60 seconds. See the Datex manual for full details.

Table 3.4: Serial protocol for the Datex AS/3 & CS/3 monitors.

Bit rate	19200
Start bits	1
Data bits	8
Parity	Even
Stop bits	1

Unfortunately the AS/3 and CS/3 monitors use a rather complicated and somewhat confusing 'transmission request' command (a string of 52 bytes) to instruct the monitor to output data (the complete output data-string is 321 bytes). The frequency of data output (every 10 seconds, 60 seconds etc) is set using bytes 43 and 44. In practice we require data output every 10 seconds, for which is encoded using byte 43 → `0Ahex`, and byte 44 → `00hex` (see below).

The Transmission Request string which is the one currently used is described below. In practice it is assembled by the `SUB requeststring` (page 195), which is part of the Datex module (Chapter 13, page 178). This string is sent only once (by the Main module) soon after system initialisation, as shown in the following code extract from the Main Module (Chapter 11, page 102), which sends the string and then waits a maximum of 5 seconds for the first incomming data-string before timing-out.

```
...
REM now trigger data output (every 10 sec) from Datex AS/3 monitor
CALL RequestString    :REM in DatexAS3 module
REM  start timer and wait max 5 sec for data to arrive
thistime=timer
DO
  IF TIMER > thistime + 5 then
     PRINT
     BEEP
     PRINT " No data --- quiting program"
     SLEEP 1
     END
  END IF
  REM if data in buffer, then continue
  IF LOC(datexAS3comportfilenumber%) > 0 then
     PRINT " data output OK"
     SLEEP 1
     EXIT DO
  END IF
  SLEEP 1
  REM print dots ... while waiting
  PRINT ".";
LOOP
...
```

3.3.1 Transmission request command

The structure of the Transmission request command-string used in this project is that of type-2 (see the correspondence at the end of this chapter), and triggers data-outout every 10 seconds. The following few points are relevant here.

- The string starts and ends with a `7Eh` byte).
- I have numbered the bytes (decimal) starting with 1 (1–52)
- The byte values are given in Hexadecimal (h) and Decimal (d).
- The bytes are divided up into their functional groups (1, 2 4 bytes etc)
- The following string is the one currently used and assembled by the `SUB requeststring` (page 195), which is part of the Datex module (Chapter 13, page 178)—see above.

```
Byte   Hex     Decimal   Description
no     value   value
===================================
1      7E      126     Start flag
-----------------------------------
2      31      49      (start of header)
3      00       0      Total length = 0031h = 49d bytes (word r_len)
-------------------------------
4      00       0      Reserved, set to zero (byte res1)
---------------------------
5      00       0      Ignored by monitor, set to zero (byte r_dri_level)
---------------------------
6      00       0      Reserved = 0000H  (byte res2[2])
7      00       0
---------------------------
8      00       0      Transmission time = 0x00000000, ignored by monitor
9      00       0      when sending transmission request (dword r_time).
10     00       0      However, time can be meaningful in outputted
11     00       0      messages, which use the header of the same
                       structure (dword r_time).
-------------------------------
12     00       0       Reserved = 00000000H (dword res3)
13     00       0
14     00       0
15     00       0
-------------------------------
16     00       0       Main type of record = DRI_MT_PHDB = 0
17     00       0       (r_maintype)
-------------------------------
18     00       0       Offset to the first subrecord = 0000H
19     00       0       (sr_desc[0].offset)
-------------------------------
20     00       0       Type of first subrecord, DRI_PH_XMIT_REQ = 0
                        (sr_desc[0].sr_type)
```

```
-------------------------------
21     00       0       Offset to the second subrecord = 0000H,
22     00       0       calculated from the
                        beginning of the data area after the header part.
                        Value is not meaningful, since there is only one
                        subrecord in the request (sr_desc[1].offset).
-------------------------------
23     FF     255      "No more subrecords" (sr_desc[1].sr_type)
-------------------------------
24       0x00
25       0x00      sr_desc[2].offset = 0x0000, no meaning since only
                      one subrecord transmitted...
26       0x00      sr_desc[2].sr_type, no meaning
------------------------
27       0x00
28       0x00      sr_desc[3].offset = 0x0000, no meaning
29       0x00      sr_desc[3].sr_type, no meaning
------------------------
30       0x00
31       0x00      sr_desc[4].offset = 0x0000, no meaning
32       0x00      sr_desc[4].sr_type, no meaning
------------------------
33       0x00
34       0x00      sr_desc[5].offset = 0x0000, no meaning
35       0x00      sr_desc[5].sr_type, no meaning
------------------------
36       0x00
37       0x00      sr_desc[6].offset = 0x0000, no meaning
38       0x00      sr_desc[6].sr_type, no meaning
------------------------
39       0x00
40       0x00      sr_desc[7].offset = 0x0000, no meaning
41       0x00      sr_desc[7].sr_type = 0x00, no meaning
-----------------------------------------
START OF THE TRANSMISSION REQUEST SUBGROUP
42       0x01      Request current values of physiological database
                    = DRI_PH_DISPL (field phdb_rcrd_type of struct
                      phdb_req)
------------------------
43       0x0A
44       0x00      Transmission interval in seconds = 0x000A, i.e.,
                     send current values of physiological database every
                     10 seconds (field tx_interval of struct phdb_req).
------------------------
45       0x00
46       0x00      reserved[0] of struct phdb_req, must be zeroed
------------------------
47       0x00
48       0x00      reserved[1] of struct phdb_req, must be zeroed
------------------------
```

```
49        0x00
50        0x00      reserved[2] of struct phdb_req, must be zeroed
---------------------------
51        0x3B      checksum
---------------------------
52        0x7E      End flag
---------------------------
```

In the Data Program, this command string is send by the `SUB RequestString` (page 195), which is part of the Datex module detailed in Chapter 13 (see also Figure 5.1, page 49).

3.4 Output data-string format

The data format for the Datex AS/3 & CS/3 monitors is described in the Datex document *AS/3 & CS/3 Computer Interface Specification, Revision 3.1, 15/5/1997*. This is a 37-page A4 document available as a WORD document from Datex, and covers the software versions listed in Table 3.2.

After the computer sends the above Transmission Request command-string, the Datex AS/3 monitor responds by outputting the following 321-byte string every 10 seconds, which consists of

(a) a 1-byte 'start' flag `<7Eh>`,
(b) a 40-byte 'header',
(c) a number of so-called 'sub-records', and finally
(d) a checksum (1 byte) followed by a 1-byte 'stop flag' `<7Eh>`.

The following few points are relevant.

- The string starts and ends with a `7Eh` byte = 126d.
- I have numbered the bytes (decimal) starting with 1 (1–321). Note that byte-1 is the FIRST byte to be received by the PC.
- The byte values are given in Hexadecimal (h)
- In the following listing the bytes are divided up into their functional groups (1, 2, 3, 4 bytes etc). Note that the order of the bytes within a group is shown in the left-hand column.
- When decoding the groups of bytes, note that UNIX rules apply, and the within-group byte-order needs to be 'reversed' (i.e. in order to have the highest byte number to the left-hand side, and lowest byte number to the right-hand side). For example, the four-byte double-word group `<CBh><CFh><F2h><33h>` (bytes 8–11) encodes for the time in seconds since 1970.00 yrs. Reversing the the byte order (i.e. having bit-0 on the right-hand side) gives the double-word `<33F2CFCBh>` which is 871550923 seconds → 10087 days → 27 July, 1997.
- Each funtional grouping (what Datex calles a 'sub-record') has a group of status-bytes (usually 4 bytes) and a group of label-bytes (usually two bytes). These status and label bytes are mostly bit-encoded to indicate such things as the source of the particular measurement, or the existance of an error state etc—some of the encodings for the more important parameters are included in this list, but it is not comprehensive just now (see manual for full details).

```
--------------------
1      <7E>        Start Flag
--------------------
START OF HEADER
2,3    <3E> <01>   Total no of bytes in transmission
                   <01h><3eh> = 318d = 318 bytes (header + data)
                   Total bytes=321= 1 start + 318 + 1checksum + 1 stop
--------------------
4      <94>
5      <03>        Interface version supported by device (0-3)
                   see version code Table

6      <00>
7      <00>
8-11   <CB> <CF> <F2> <33>   Time in secs since 1970.00 yrs
                                  = <33><f2><cf><cb>=871550923 secs
                                  = 10087 days= 27yrs 7 months 22 days
                                  = 27 July 1997
12-15  <00> <00> <00> <00>
16-17  <00> <00>
18-20  <00> <00> <01>    The <1> here is sr_type for output data (1-4, p 10)
21-23  <BD> <BD> <ff>    ?? the <ff> indicating no more subrecords??
24-26  <BD> <BD> <BD>    ?? why are these fields filled with <BD> ????
27-29  <BD> <BD> <BD>    Note <bdh> = 189d
30-32  <BD> <BD> <BD>
33-35  <BD> <BD> <BD>
36-38  <BD> <BD> <BD>
39-41  <BD> <BD> <BD>
------end of header, always  40 bytes------------

-----start of the data area-------------------------
42-45  <CB> <CF> <F2> <33>   Time in secs since 1/1/1970 = 8.7155092 E08
----------------
       ECG subrecord
46-49  <0B> <3A> <00> <00>   ECG group header (status- 4 bytes)
50-51  <74> <32>                               (label - 2 bytes)
52-53  <02> <80>       HR
54-55  <0A> <81>       st1 (mm/100)
56-57  <05> <81>       st2
58-59  <08> <81>       st3
60-61  <01> <80>       rr  (resp rate/min)
--------------
       INV Press(1) subrecord
62-65  <1> <0> <0> <0>   Inv Press 1 header (status)
66-67  <1> <0>                             (label)
68-69  <2> <80>      sys     x100
70-71  <2> <80>      diast   x100
72-73  <2> <80>      mean    x100
74-75  <1> <80>      heart rate/min
```

```
--------------
       INV Press(2) subrecord
76-79  <1> <0> <0> <0>   status
80-81  <2> <0>           label
82-83  <2> <80>          sys      x100
84-85  <2> <80>          diast    x100
86-87  <2> <80>          mean     x100
88-89  <1> <80>          heart rate/min
---------------
        INV Press(3) subrecord
90-93   <1> <0> <0> <0>  status
94-95   <B> <0>          label
96-97   <2> <80>         sys       x100
98-99   <2> <80>         diast     x100
100-101 <2> <80>         mean      x100
102-103 <1> <80>         heart rate/min
----------------
        INV Press(4) subrecord
104-107 <1> <0> <0> <0>  status
108-109 <3> <0>          label
110-111 <2> <80>         sys       x100
112-113 <2> <80>         diast     x100
114-115 <2> <80>         mean      x100
116-117 <1> <80>         heart rate/min
--------------------
        NIBP subrecord
118-121 <3> <0> <0> <0>   status
122-123 <3> <1>           label (bit-8 --> 1 after 60 secs)
124-125 <1> <80>   sys       x100
126-127 <1> <80>   diast     x100
128-129 <1> <80>   mean      x100
130-131 <1> <80>   HR        /min
--------------------
        Temp (1)  subrecord
132-135 <3> <0> <0> <0>    status
136-137 <B> <0>            label
138-139 <1> <80>           deg C x100
------------------
        Temp (2)  subrecord
140-143 <3> <0> <0> <0>    status
144-145 <C> <0>            label
146-147 <1> <80>           deg C x100
------------------
        Temp (3)  subrecord
148-151 <0> <0> <0> <0>    status
152-153 <D> <0>            label
154-155 <1> <80>           deg C x100
------------------
        Temp (4)  subrecord
156-159 <0> <0> <0> <0>    status
```

```
160-161 <E> <0>            label
162-163 <1> <80>           deg C x100
------------------
        Saturation (SpO2)  subrecord
164-167 <3> <0> <0> <0>  status
168-169 <0> <0>          label(00=SaO2 01=SvO2 10=error)
170-171 <1> <80>     (SAT% * 100)
172-173 <1> <80>     HR
174-175 <2> <80>     IR-amp (infra red amplitude)
176-177 <1> <80>     label for SaO2 = 1 /SvO2 = 2 /SO2= 0 / 3 not used
------------------
        Carbon dioxide (CO2)  subrecord
178-181 <47> <0> <0> <0>  status
182-183 <1> <0>           label ( source: 01=CO2 10=ECG)
184-185 <1> <80>    ET (% x100)
186-187 <1> <80>    FI (% x100)
188-189 <1> <80>    RR
190-191 <28> <1D>   amb_P (x10 mmHg ambient pressure)
 ------------------
        Oxygen (O2) subrecord
192-195 <3> <0> <0> <0>  status
196-197 <0> <0>          label
198-199 <1> <80>  ET O2  (% x100)
200-201 <1> <80>  FI O2  (% x100)
----------------------
        Nitrous Oxide (N2O) subrecord
202-205 <3> <0> <0> <0>     status
206-207 <0> <0>             label
208-209 <1> <80>     ET N2O (% x100)
210-211 <1> <80>     FI N2O (% x100)
----------------------
        Anaesthetic agent
212-215 <3> <0> <0> <0>     status
216-217 <2> <0>             label
218-219 <1> <80>     ET AA (% x100)
220-221 <1> <80>     FI AA (% x100)
222-223 <0> <0>      MAC sum (x100)
----------------------
        Flow & Volume
224-227 <3> <0> <0> <0>  status
228-229 <0> <0>          label
230-231 <0> <0>          RR  (resp rate)
232-233 <1> <80>         pPeak      x100
234-235 <1> <80>         peep       x100
236-237 <1> <80>         pPlat      x100
238-239 <1> <80>         TV-insp    x10
240-241 <1> <80>         TV-exp     x10
242-243 <1> <80>         compliance x100 cms H2O
244-245 <1> <80>         MV exp     x100/min
-----------------
```

```
        Cardiac Output & Wedge press
246-249 <3> <0> <0> <0>  status
250-251 <7> <0>          label
252-253 <1> <80>         CO
254-255 <1> <80>         Blood Temp
256-257 <1> <80>         Ref
258-259 <1> <80>         pcwp
----------------------
        Neuro-Muscular J (NMJ)
260-263 <20> <0> <0> <0>  status
264-265 <0> <0>           label
266-267 <1> <80>
268-269 <1> <80>
270-271 <FF> <8d>
-----------------
        ECG (2)  (no header)
272-273 <2> <80>
274-275 <1> <80>
276-277 <1> <80>
-----------------------
        Reserved-1 (8 bytes)
278-285 <0> <0> <0> <0> <D3> <0> <1> <80>
------------
        Invas Press (5) subrecord
286-289 <0> <0> <0> <0>
290-291 <D> <0>
292-293 <2> <80>
294-295 <2> <80>
296-297 <2> <80>
298-299 <1> <80>
--------
        Invas Press (6) subrecord
300-303 <0> <0> <0> <0>
304-305 <E> <0>
306-307 <2> <80>
308-309 <2> <80>
310-311 <2> <80>
312-313 <1> <80>
------------------
        Reserved-2 (2 bytes)
314-315 <0> <0>
------------------
    Marker Byte
316 <0>
---------------
    Reserved-3 (1 byte)
317 <0>
-----------------
        Last WORD
318-319 <31> <0>         (2 --> 319 = 318 bytes)
```

```
------------------
320 <B9>    checksum
-------------------
321 <7E>    stop Flag
 ---------------------------- end of transmission-------------
```

3.5 Example of data output

The following Datex AS/3 output data-string was received during an operation and saved in D-data (decimal), and is the same as that shown in Chapter 14, page 203. For details of the format of the D-data see page 203.

```
....
....
AS300,09:36:49,05-03-1991,(m/d/y) Datex AS/3 monitor
AS301,126,062,001,111,005,000,000,166,052,241,058,000,000,000,000,000,000,000
AS302,000,001,000,074,255,097,220,044,000,000,000,044,000,000,000,189,189,032
AS303,000,189,189,032,000,166,052,241,058,019,048,000,000,000,034,067,000,021
AS304,000,001,128,001,128,001,128,003,000,000,000,001,000,062,058,231,028,049
...
...
AS316,141,001,128,067,000,066,000,000,000,000,000,189,189,001,128,000,000,000
AS317,000,013,000,002,128,002,128,002,128,001,128,000,000,000,000,014,000,002
AS318,128,002,128,002,128,001,128,000,000,000,064,081,000,222,126
```

One of the Datex AS/3 invasive blood pressure 'sub-records' is encoded in bytes 62–75, as shown in the following Table.

Table 3.5: Decoding invasive blood pressure 1 (bytes 62–75). The systolic, diastolic and mean blood pressure values $\times 100$ are encoded as Hex words (Unix). The decimal value therefore has to be divided by 100 to obtain the physiological value, and in this particular case the decoded values are: systolic BP 149.1, diastolic BP 73.99, mean BP 105.45. In practice we would only pass on the integer values for blood pressure.
This decoding is performed by the `SUB decode` (page 189) of the Datex module (Chapter 13, page 178).

	mean	diastolic	systolic
Byte number	73 72	71 70	69 68
Hex values	29h 31h	1Ch E7h	3Ah 3Eh
Hex word	2931h	1CE7h	3A3Eh
decimal	10545	7399	14910
BP = decimal/100	105.45	73.99	149.1

The following is the same data but placed in byte order (1–321), together with the Dec and Hex equivalent.

```
byte,Hex,Dec
 -----------
 001,7E,126
 002,3E,062
 003,01,001
 004,6F,111
 005,05,005
 006,00,000
 007,00,000
 008,A6,166
 009,34,052
 010,F1,241
 011,3A,058
 012,00,000
 013,00,000
 014,00,000
 015,00,000
 016,00,000
 017,00,000
 018,00,000
 019,00,000
 020,01,001
 021,00,000
 022,4A,074
 023,FF,255
 024,61,097
 025,DC,220
 026,2C,044
 027,00,000
 028,00,000
 029,00,000
 030,2C,044
 031,00,000
 032,00,000
 033,00,000
 034,BD,189
 035,BD,189
 036,20,032
 037,00,000
 038,BD,189
 039,BD,189
 040,20,032
 041,00,000
 042,A6,166
 043,34,052
 044,F1,241
 045,3A,058
 046,13,019
 047,30,048
 048,00,000
```

```
049,00,000
050,00,000
051,22,034
052,43,067
053,00,000
054,15,021
055,00,000
056,01,001
057,80,128
058,01,001
059,80,128
060,01,001
061,80,128
062,03,003
063,00,000
064,00,000
065,00,000
066,01,001
067,00,000
068,3E,062
069,3A,058
070,E7,231
071,1C,028
072,31,049
073,29,041
074,43,067
075,00,000
076,03,003
077,00,000
078,00,000
079,00,000
080,02,002
081,00,000
082,F7,247
083,08,008
084,F4,244
085,05,005
086,2C,044
087,07,007
088,43,067
089,00,000
090,00,000
091,00,000
092,00,000
093,00,000
094,0B,011
095,00,000
096,02,002
097,80,128
098,02,002
```

```
099,80,128
100,02,002
101,80,128
102,01,001
103,80,128
104,00,000
105,00,000
106,00,000
107,00,000
108,03,003
109,00,000
110,02,002
111,80,128
112,02,002
113,80,128
114,02,002
115,80,128
116,01,001
117,80,128
118,03,003
119,00,000
120,00,000
121,00,000
122,03,003
123,01,001
124,01,001
125,80,128
126,01,001
127,80,128
128,01,001
129,80,128
130,01,001
131,80,128
132,03,003
133,00,000
134,00,000
135,00,000
136,0B,011
137,00,000
138,D2,210
139,0D,013
140,03,003
141,00,000
142,00,000
143,00,000
144,0C,012
145,00,000
146,04,004
147,80,128
148,00,000
```

```
149,00,000
150,00,000
151,00,000
152,0D,013
153,00,000
154,01,001
155,80,128
156,00,000
157,00,000
158,00,000
159,00,000
160,0E,014
161,00,000
162,01,001
163,80,128
164,03,003
165,00,000
166,00,000
167,00,000
168,00,000
169,00,000
170,DE,222
171,26,038
172,44,068
173,00,000
174,6C,108
175,00,000
176,01,001
177,80,128
178,03,003
179,00,000
180,00,000
181,00,000
182,09,009
183,00,000
184,8A,138
185,01,001
186,00,000
187,00,000
188,0C,012
189,00,000
190,66,102
191,1D,029
192,03,003
193,00,000
194,00,000
195,00,000
196,00,000
197,00,000
198,71,113
```

```
199,0E,014
200,A5,165
201,0F,015
202,03,003
203,00,000
204,00,000
205,00,000
206,00,000
207,00,000
208,07,007
209,17,023
210,F1,241
211,16,022
212,03,003
213,00,000
214,00,000
215,00,000
216,04,004
217,00,000
218,00,000
219,00,000
220,00,000
221,00,000
222,3A,058
223,00,000
224,03,003
225,00,000
226,00,000
227,00,000
228,00,000
229,00,000
230,0C,012
231,00,000
232,0A,010
233,0F,015
234,08,008
235,02,002
236,C0,192
237,0D,013
238,82,130
239,16,022
240,E5,229
241,14,020
242,F4,244
243,06,006
244,7E,126
245,02,002
246,00,000
247,00,000
248,00,000
```

```
249,00,000
250,07,007
251,00,000
252,01,001
253,80,128
254,01,001
255,80,128
256,01,001
257,80,128
258,01,001
259,80,128
260,20,032
261,00,000
262,00,000
263,00,000
264,00,000
265,00,000
266,01,001
267,80,128
268,01,001
269,80,128
270,FF,255
271,8D,141
272,01,001
273,80,128
274,43,067
275,00,000
276,42,066
277,00,000
278,00,000
279,00,000
280,00,000
281,00,000
282,BD,189
283,BD,189
284,01,001
285,80,128
286,00,000
287,00,000
288,00,000
289,00,000
290,0D,013
291,00,000
292,02,002
293,80,128
294,02,002
295,80,128
296,02,002
297,80,128
298,01,001
```

```
299,80,128
300,00,000
301,00,000
302,00,000
303,00,000
304,0E,014
305,00,000
306,02,002
307,80,128
308,02,002
309,80,128
310,02,002
311,80,128
312,01,001
313,80,128
314,00,000
315,00,000
316,00,000
317,40,064
318,51,081
319,00,000
320,DE,222
321,7E,126
----------
```

3.6 Correspondence with Datex

```
...

The subrecord types are intended to be used in the sr_type field of the
sr_desc -struct:s (see section 3.2, page 8 of the specification) and 0
(=DRI_PH_XMIT_REQ) is the correct value for that field. However, the
phdb_rcrd_type field in the data structure "struct phdb_req" is used
for a different purpose, though the used enumeration is the same, and I
admit that it is confusing.

The phdb_rcrd_type field indicates what kind of physiological data you
are requesting, for example:

sr_type = 0, phdb_rcrd_type = 1    => Send current values of the
                                      physiological database.

sr_type = 0, phdb_rcrd_type = 2    => Send 10 s trended values

sr_type = 0, phdb_rcrd_type = 3    => Send 60 s trended values
sr_type = 0, phdb_rcrd_type = 4    => Send auxiliary phys. information

values 1, 2, 3 and 4 for field sr_type are reserved for output values,
as you suggested.
```

So DRI_PH_DISPL = 1, DRI_PH_10S_TREND = 2, DRI_PH_60S_TREND = 3 and DRI_PH_AUX_INFO = 4. The values correspond to subrecord type listed on page 10 of the specification, although the constant names are not explicitly defined in the table.

In addition, the texts in the "Value" field of the table on page 11 of the specification (related to tx_interval) are incorrect: Instead of texts "Any positive value together with subrecord type ..." the texts should be "Any positive value together with physiological record type ..." referencing to field phdb_rcrd_type of struct phdb_req rather than to the sr_type field of struct sr_desc.

.... the tx_interval field specifies the transmission interval for the physiological data records, the type of which is specified by the field phdb_rcrd_type ("... together with subrecord type ..."). For 10s and 60s trends the transmission interval is, however, always fixed (10s and 60s). In addition, the special values -1 and 0 have special side effects as documented in the table on page 11 of the specification.

Chapter 4

A PowerBasic (DOS) program for simulating the AS/3 data output

ch-as3sim.tex

4.1 Introduction

This chapter describes a simple PowerBasic (DOS) program which simulates the Datex AS/3 data output, by outputting previously saved data-strings.

Testing the Anaesthetic Record 5a program (`a5-anes.pb`) requires using a second PC to simulate the AS/3 monitor, and output serial data previously collected during an anaesthetic, and stored in our usual `Dnn` data format.

```
AS300,10:16:29,10-11-2001 (m/d/y), Datex AS/3 monitor (AS322.dat)
AS301,126,062,001,126,005,000,000,044,113,197,059,000,000,000,000,000,000,000
AS302,000,001,000,189,255,097,220,044,000,000,000,189,189,000,000,189,189,032
AS303,000,189,189,032,000,044,113,197,059,019,048,000,000,000,034,076,000,018
AS304,000,001,128,001,128,001,128,003,000,000,000,001,000,184,040,120,021,188
AS305,029,076,000,001,000,000,000,002,000,002,128,002,128,002,128,001,128,000
AS306,000,000,000,011,000,002,128,002,128,002,128,001,128,000,000,000,000,003
AS307,000,002,128,002,128,002,128,001,128,003,000,000,000,003,001,001,128,001
AS308,128,001,128,001,128,003,000,000,000,011,000,004,128,003,000,000,000,012
AS309,000,004,128,000,000,000,000,013,000,001,128,000,000,000,000,014,000,001
AS310,128,003,000,000,000,000,000,028,037,077,000,199,000,001,128,003,000,000
AS311,000,009,000,216,001,000,000,010,000,192,029,003,000,000,000,000,000,184
AS312,022,171,024,003,000,000,000,000,000,145,013,131,013,003,000,000,000,006
AS313,000,103,000,132,000,084,000,003,000,000,000,000,000,010,000,028,012,004
AS314,001,194,011,222,017,026,017,054,006,175,001,000,000,000,000,007,000,001
AS315,128,001,128,001,128,001,128,032,000,000,000,000,000,001,128,001,128,255
AS316,141,076,000,077,000,076,000,000,000,000,000,189,189,001,128,000,000,000
AS317,000,013,000,002,128,002,128,002,128,001,128,000,000,000,000,014,000,002
AS318,128,002,128,002,128,001,128,000,000,000,064,081,000,049,126
```

The following PB35 program is what I currently use, and it appears to work alright. Some important points regarding the simulator program are as follows:

- The bytes must be sent as the characters, i.e. as `CHR$(asciivalue%)` (say, `32`), and *not* as a three digit string (eg `032`).
- The bytes must be sent without any CR/LF characters. In BASIC (QB4.5 and PB3.5) this is achieved by using a ';' after the `PRINT` command eg `PRINT CHR$(asciivalue%);`
- There should be a 10 second delay between sending each data set (321 bytes), as this is the data frequency of the Datex AS/3 monitor (in my program I have 1 pixel $\equiv$ 10 seconds). This can be speeded up when just testing to see if the bytes are being received OK, but the 10 sec delay needs to be present when testing the screen display.
- Handling ASCII characters 125 and 126.

  Note that the Datex data collected by our main anaesthesia program is actually saved to the harddrive *after* all character pairs `<125><93>` and `<125><94>` input data stream from the AS3 monitor have been converted to `<125>` and `<126>` respectively.

  Consequently, in order to be able to use previously saved data, and make it appear as though it were original data comming from an AS3 monitor, then this translation needs to be reversed accordingly, with the exception of the `<126>` characters at the beginning and end (i.e. in positions `k=1` and `k=321`). This is done using the following code.

```
...
...
   FOR k = 1 TO n
       SELECT CASE k
          CASE 1, 321
            PRINT mydata%(k)
            PRINT #2, CHR$(mydata%(k));

          CASE ELSE
             SELECT CASE mydata%(k)
               CASE 125
                  PRINT mydata%(k)
                  PRINT #2, CHR$(125) + CHR$(93);

               CASE 126
                  PRINT mydata%(k)
                  PRINT #2, CHR$(125) + CHR$(94);

               CASE ELSE
                  PRINT mydata%(k)
                  PRINT #2, CHR$(mydata%(k));

             END SELECT
          END SELECT
```

```
        NEXT k
    ...
    ...
```

4.2 Configuration file (as3sim.cfg)

The simulator program reads a configuration file which contains the serial protocol. This file also contains a test string which can be sent via the PC serial port simply for testing purposes (e.g., for testing the leds monitoring the serial data traffic).

```
% AS3sim.cfg
set baud=2400
set parity=N
set stopbits=1
set databits=8
%%set inputfile=dnntest.dat
set inputfile=91607C53.d01
set timeinterval=5
set comport=com1
%%set sendstring=abcdefghij12345
%-----------
```

4.3 The simulator program

Note that this simulator simply outputs Datex data strings at a specified frequency. It does not respond to a data request message from the PC.

```
REM as3sim.PB   (modified from as3sim.bas = QB45 version)
REM sends as3 data via serial port from Dnn files

REM -----------
rem change COMMON SHARED --> SHARED for PB
REM -----------
DECLARE SUB sendthestring ()
DECLARE SUB help ()
SHARED SENDSTRING$
SHARED baud$, parity$, databits$, stopbits$, timeinterval$, comport$
SHARED cfgnumber%, cfgfile$
REM set the buffer size
$com 1024
REM -----------------
COLOR 15, 1
CLS
REM ----------------------
switch$ = COMMAND$
REM remember that caommand$ yields an UPPERCASE string
IF INSTR(UCASE$(switch$), "/?") > 0 OR INSTR(UCASE$(switch$), "/H") > 0 THEN
       CALL help
       END
```

```
END IF
REM -----------------------
PRINT
PRINT " AS3sim.exe"
PRINT " A MS-DOS simulator program for sending .Dnn data via the serial port"
PRINT " Requires a configuration file: default = AS3sim.cfg"
PRINT " Press Q to Quit"
PRINT " For help: AS3sim /?  or /h          (C) copyright RWD Nickalls 2001"
PRINT
SLEEP 1
REM ----------------------
REM first read the .cfg file
IF switch$ <> "" THEN
      cfgfile$ = switch$
      ELSE
      cfgfile$ = "as3sim.cfg"
      PRINT " No config file specified, so using default file as3sim.cfg"
      END IF
OPEN cfgfile$ FOR INPUT AS #1
cfgnumber% = 0
PRINT " Reading the .cfg file ("; cfgfile$; ")"
PRINT
SLEEP 1

DO WHILE NOT EOF(1)
  keyy$ = INKEY$
  IF UCASE$(keyy$) = "Q" THEN
       CLOSE
       END
  END IF
  REM input each line
  INPUT #1, cfgline$
  REM    PRINT cfgline$
  IF LEFT$(cfgline$, 1) = "%" THEN GOTO endofloop

  IF INSTR(UCASE$(cfgline$), "SET BAUD=") THEN
           REM count no of parameters
           cfgnumber% = cfgnumber% + 1
           baud$ = RIGHT$(cfgline$, LEN(cfgline$) - 9)
           PRINT "baud = "; baud$
           END IF

  IF INSTR(UCASE$(cfgline$), "SET PARITY=") THEN
           REM count no of parameters
           cfgnumber% = cfgnumber% + 1
           parity$ = RIGHT$(cfgline$, LEN(cfgline$) - 11)
           PRINT "parity = "; parity$
           END IF

  IF INSTR(UCASE$(cfgline$), "SET DATABITS=") THEN
```

```
          REM count no of parameters
          cfgnumber% = cfgnumber% + 1
          databits$ = RIGHT$(cfgline$, LEN(cfgline$) - 13)
          PRINT "databits = "; databits$
          END IF

  IF INSTR(UCASE$(cfgline$), "SET STOPBITS=") THEN
          REM count no of parameters
          cfgnumber% = cfgnumber% + 1
          stopbits$ = RIGHT$(cfgline$, LEN(cfgline$) - 13)
          PRINT "stopbits = "; stopbits$
          END IF

  IF INSTR(UCASE$(cfgline$), "SET INPUTFILE=") THEN
          REM count no of parameters
          cfgnumber% = cfgnumber% + 1
          inputfile$ = RIGHT$(cfgline$, LEN(cfgline$) - 14)
          PRINT "inputfile = "; inputfile$
          END IF

  IF INSTR(UCASE$(cfgline$), "SET TIMEINTERVAL=") THEN
          REM count no of parameters
          cfgnumber% = cfgnumber% + 1
          timeinterval$ = RIGHT$(cfgline$, LEN(cfgline$) - 17)
          PRINT "timeinterval = "; timeinterval$
          END IF

  IF INSTR(UCASE$(cfgline$), "SET COMPORT=") THEN
          REM count no of parameters
          cfgnumber% = cfgnumber% + 1
          comport$ = RIGHT$(cfgline$, LEN(cfgline$) - 12)
          PRINT "comport = "; comport$
          END IF

  IF INSTR(UCASE$(cfgline$), "SET SENDSTRING=") THEN
          REM count no of parameters
          SENDSTRING$ = RIGHT$(cfgline$, LEN(cfgline$) - 15)
          cfgnumber% = cfgnumber% + 1
          CLOSE #1
          IF SENDSTRING$ <> "" THEN
                    CALL sendthestring
                    ELSE
                    PRINT "error with sendString$ in .cfg file"
                    END
          END IF
   END IF
endofloop:
LOOP
REM close the configuration file
CLOSE #1
```

```
REM ----------------------------
PRINT
REM do some basic checking
IF cfgnumber% <> 7 THEN
        REM error
        BEEP
        PRINT " ** problem with " + cfgfile$ + " file: ? missing data"
        END
        PRINT
    ELSE
        IF switch$ <> "" THEN
           PRINT switch$ + " file OK"
        ELSE
           PRINT " " + cfgfile$ + " file OK"
        END IF
        PRINT
END IF
SLEEP 2

REM create array for the data
DIM mydata%(350)

OPEN inputfile$ FOR INPUT AS #1
REM OPEN "com1: 19200,N,8,1,cs,ds,op2000,TB2000" FOR OUTPUT AS #2
protocol$ = baud$ + "," + parity$ + "," + databits$ + "," + stopbits$
thisopen$ = comport$ + ":" + protocol$ + ",cs,ds"

PRINT " opening the serial port as:-"
PRINT " OPEN " + thisopen$
SLEEP 3
OPEN thisopen$ FOR OUTPUT AS #2
PRINT
PRINT " com port open OK"
IF VAL(timeinterval$) < 1 THEN timeinterval$ = "1"
PRINT " sending AS3 data from file " + inputfile$ + " at "; VAL(timeinterval$);
                                                          "second intervals"
PRINT " Press Q to quit"
PRINT
SLEEP 4

DO WHILE NOT EOF(1)
   REM n is the  number of data points in each data set (= 321)
   REM need to cound the data points to detect errors
   n = 0
   DO
      keyy$ = INKEY$
      IF UCASE$(keyy$) = "Q" THEN
         CLOSE
         END
      END IF
```

```
REM input the first line of data
REM The heading has 6 chars AS3nnn
INPUT #1, indata$
REM   PRINT indata$

SELECT CASE indata$
  CASE "AS300"

  CASE "AS301"
  REM input the next 18 bytes (1-18)
  FOR j = 1 TO 18: INPUT #1, mydata%(j): NEXT j
  n = n + 18

  CASE "AS302"
     REM input the next 18 bytes (19-36)
     FOR j = 19 TO 36: INPUT #1, mydata%(j): NEXT j
     n = n + 18

  CASE "AS303"
     REM input the next 18 bytes (37-54)
     FOR j = 37 TO 54: INPUT #1, mydata%(j): NEXT j
     n = n + 18

  CASE "AS304"
     REM input the next 18 bytes (55-72)
     FOR j = 55 TO 72: INPUT #1, mydata%(j): NEXT j
     n = n + 18

  CASE "AS305"
     REM input the next 18 bytes (73-90)
     FOR j = 73 TO 90: INPUT #1, mydata%(j): NEXT j
     n = n + 18

  CASE "AS306"
     REM input the next 18 bytes (91-108)
     FOR j = 91 TO 108: INPUT #1, mydata%(j): NEXT j
     n = n + 18

  CASE "AS307"
     REM input the next 18 bytes (109-126)
     FOR j = 109 TO 126: INPUT #1, mydata%(j): NEXT j
     n = n + 18

   CASE "AS308"
     REM input the next 18 bytes (127-144)
     FOR j = 127 TO 144: INPUT #1, mydata%(j): NEXT j
     n = n + 18

  CASE "AS309"
     REM input the next 18 bytes (145-162)
```

```
   FOR j = 145 TO 162: INPUT #1, mydata%(j): NEXT j
   n = n + 18

CASE "AS310"
   REM input the next 18 bytes (163-180)
   FOR j = 163 TO 180: INPUT #1, mydata%(j): NEXT j
   n = n + 18

CASE "AS311"
   REM input the next 18 groups (181-198)
   FOR j = 181 TO 198: INPUT #1, mydata%(j): NEXT j
   n = n + 18

CASE "AS312"
   REM input the next 18 bytes (199-216)
   FOR j = 199 TO 216: INPUT #1, mydata%(j): NEXT j
   n = n + 18

CASE "AS313"
   REM input the next 18 bytes (217-234)
   FOR j = 217 TO 234: INPUT #1, mydata%(j): NEXT j
   n = n + 18

CASE "AS314"
   REM input the next 18 bytes (235-252)
   FOR j = 235 TO 252: INPUT #1, mydata%(j): NEXT j
   n = n + 18

CASE "AS315"
   REM input the next 18 bytes (253-270)
   FOR j = 253 TO 270: INPUT #1, mydata%(j): NEXT j
   n = n + 18

CASE "AS316"
   REM input the next 18 bytes (271-288)
   FOR j = 271 TO 288: INPUT #1, mydata%(j): NEXT j
   n = n + 18

CASE "AS317"
   REM input the next 18 bytes (289-306)
   FOR j = 289 TO 306: INPUT #1, mydata%(j): NEXT j
   n = n + 18

CASE "AS318"
   REM input the next 15 bytes (307-321)
   FOR j = 307 TO 321: INPUT #1, mydata%(j): NEXT j
   n = n + 15
   GOTO printall

CASE ELSE
```

```
         PRINT indata$ + SPACE$(1)

  END SELECT
LOOP
REM -----------
printall:
s = s + 1
REM check number of data points
IF n <> 321 THEN
    BEEP
    PRINT "ERROR: only "; n; " data points - getting next set"
    SLEEP 2
    GOTO jump
END IF
REM -----------
FOR k = 1 TO n
    SELECT CASE k
       CASE 1, 321
         PRINT mydata%(k)
         PRINT #2, CHR$(mydata%(k));

       CASE ELSE
          SELECT CASE mydata%(k)
            CASE 125
               REM   BEEP
               PRINT mydata%(k)
               PRINT #2, CHR$(125) + CHR$(93);

            CASE 126
               REM  BEEP
               PRINT mydata%(k)
               PRINT #2, CHR$(125) + CHR$(94);

            CASE ELSE
               PRINT mydata%(k)
               PRINT #2, CHR$(mydata%(k));

          END SELECT
       END SELECT

NEXT k

REM ----------
jump:
REM now write the screen time since start of screen
REM based on each data set = 10 sec
PRINT "-----------------------------": PRINT
m = INT(s * 10 / 60)
r = INT((s * 10) - (m * 60))
PRINT "Data Set "; s, , m; "mins"; r; "secs"
```

```
   REM create a time delay
   PRINT "TimeInterval = "; VAL(timeinterval$); " seconds"
   PRINT "Press Q to quit"
   IF VAL(timeinterval$) < 1 THEN timeinterval$ = "1"
      FOR delayy = 1 TO VAL(timeinterval$)
          PRINT ".";
          SLEEP 1
          keyy$ = INKEY$
          IF UCASE$(keyy$) = "Q" THEN
             CLOSE
             END
          END IF
    NEXT delayy
    PRINT
    PRINT
LOOP
END



SUB help
PRINT
PRINT "AS3sim.exe: commandline format is  <AS3sim>  <configfilename>"
PRINT "If no <configfilename> given, then default <AS3sim.cfg> is used"
PRINT "Press Q to quit"
PRINT "Configuration file format"
PRINT "%% as3sim.cfg"
PRINT "SET BAUD=19200"
PRINT "SET PARITY=E"
PRINT "SET DATABITS=8"
PRINT "SET STOPBITS=1"
PRINT "SET INPUTFILE=91607C53.d01"
PRINT "SET TIMEINTERVAL=10"
PRINT "SET COMPORT=COM1"
PRINT "%%SET SENDSTRING=abcdEfghij123245"
PRINT "(1) The above configuration file for the AS3 monitor shows"
PRINT "the required format. Take care NOT to introduce any"
PRINT "additional spaces or characters. Can use uppercase OR lowercase"
PRINT "%% this is a comment line; MUST be preceeded with % char"
PRINT "** comments % are NOT allowed at ends of lines"
PRINT "(2) Testing sending a string.Define the test string to be sent"
PRINT "using SET SENDSTRING=  as shown above "
PRINT "Inputfiles MUST only be .dnn files, eg 91607C53.d01"
PRINT
END SUB



SUB sendthestring
```

```
PRINT "sendstring = "; SENDSTRING$
PRINT
IF cfgnumber% <> 8 THEN
    REM error
    PRINT "problem with "; cfgfile$ + "file: ? missing data"
    PRINT
    ELSE
    PRINT " " + cfgfile$ + "file OK"
END IF
REM OPEN "com1: 19200,N,8,1,cs,ds,op2000,TB2000" FOR OUTPUT AS #2
protocol$ = baud$ + "," + parity$ + "," + databits$ + "," + stopbits$
REM QB45 thisopen$ = comport$ + ":" + protocol$ + ",cs,ds,op2000,TB2000"
REM cannot use OP or TB with PB
thisopen$ = comport$ + ":" + protocol$ + ",cs,ds"
PRINT " opening the serial port as:-"
PRINT " OPEN " + thisopen$
SLEEP 3
OPEN thisopen$ FOR OUTPUT AS #2
PRINT
PRINT " com port open OK"
IF VAL(timeinterval$) < 1 THEN timeinterval$ = "1"
PRINT " sending datastring from configuration file at "; VAL(timeinterval$);
                                                          "second intervals"
PRINT " press Q to quit"
PRINT
SLEEP 1

DO
  keyy$ = INKEY$
  IF UCASE$(keyy$) = "Q" THEN
       CLOSE
       END
  END IF
  PRINT SENDSTRING$
  PRINT #2, SENDSTRING$

  REM create a time delay
  REM check for zero timeinterval error - done above
  FOR delayy = 1 TO VAL(timeinterval$)
        PRINT ".";
        SLEEP 1
        keyy$ = INKEY$
        IF UCASE$(keyy$) = "Q" THEN
            CLOSE
            END
        END IF
     NEXT delayy
     PRINT
LOOP
END
```

```
END SUB
```

Chapter 5

System overview

ch-syst.tex

5.1 Introduction

The Anaesthesia Record System (ARS) accesses data from the Datex AS/3 anaesthesia monitor in the operating theatre, and displays the data as trends in real-time on the screen. It also processes keyboard inputs, and prints-out the trend-data in a (paper) format suitable for placing directly into the patient notes, known as the Anaesthetic Record. The Anaesthetic Record System software consists of three main functional components as follows (see Figure 5.1, page 49).

- A 'front-end' menu program (Chapter 6)
- A data collection and display program (Chapter 7)
- A printing program (Chapter 15)

These are now described briefly in turn.

5.2 Front-end menu

The front-end menu program (`menu-5a.com`, page 53) is a compiled batch file. It displays a number of pull-down menus and allows various parts of the system to be run separately. At start-up, the option for launching the data program (`a5-main.exe`) is automatically highlighted, and so all the anaesthetist has to do at the beginning of an operation is to press the <CR> key. The following extract from the menu program shows that on exit from the ARS program the menu program then executes the batch file `anes-prt.bat` which coordinates the printing process.

```
CASE errorlevel = 1
     ....
     REM now START the ARS programme <a5-anes.exe>
     A5-main.exe
     ...
     REM now PRINT out the anaesthetic record
```

```
REM if file <anes-prt.bat> has been written by main program
IF EXIST anes-prt.bat
    CALL anes-prt.bat
    ....
ENDIF
....
```

5.3 Data program

This is a compiled DOS PowerBasic 3.5 program which was originally written in QuickBasic 4.5, and later ported to PowerBasic in order to accommodate the serial-port protocol used by the recent Datex AS/3 anaesthesia monitor, which uses an 11-bit character frame (see Chapter 3). The Data Program consists of three main functional components as follows (see Figure 5.2, page 50).

- The **main module**, M1–M6 (Chapter 8).

  This initialises the Data Program, as well as creating filenames, and opening the various input and output files etc. The main module ends by CALLing the loop section

- The **loop section**, L1–L7 (Chapter 9)

  The main module end by CALLing the SUB `LoopOne` (page 133). This is a `DO-LOOP` which coordinates (a) data access from the Datex AS/3 anaesthesia monitor, (b) writing trend data to the screen, (c) handling F-key inputs, (d) saving data to the harddrive, (e) other general housekeeping (eg updating the screen clock, cycling the screen at hourly intervals etc.

- The **F-key section**, F1–F10 (Chapter 10)

  If an F-key is pressed, this results in setting some general and key-specific FLAGs, which are processed by the loop-section. The loop section is arranged so that key flags are only processed *between* critical episodes, i.e. not while accessing serial port data, or during screen maintenance etc.

All incoming raw data from the Datex AS/3 anaesthesia monitor via the serial port (COM port) is accessed and processed by the DatexAS3 module (Chapter 13), and then displayed graphically in real-time on the screen. All the raw data (D-data)[1] is saved by writing it to the hard drive. A modified version of the data (G-data)[2], which is used for printing out using the GNUplot and TEX systems, is saved in separate files each containing 1 hour of data. Error messages are written to the screen on a line at bottom of screen, and are visible for a variable period of time, depending on the significance or importance of the error.

The screen trend windows (Figure 5.3, page 51) hold 90 mins of data (1 pixel = 10 secs). At the end of the first 90 min period, the screen is refreshed, and the last 20 mins of trend data is replotted before plotting any new 'real-time' data. After this, the screen is refreshed at the end of each hour. The last 20 mins of screen data is saved to a temporary file (`lasthour.dat`, page 206) (see SUB `LoopOne`, page 133).

[1] D-data is so-called because it is raw data from the Datex AS/3 monitor and it is therefore saved to files having a `.Dnn` file extension.

[2] G-data is so-called because it is saved in a format suitable for processing with GNUplot, and is therefore saved to files having a `.Gnn` file extension.

At the end of the anaesthetic, the Data Program is terminated by pressing either F1 (quit only) or F10 (quit and start printing). In practice, once the Data Program has been quit, command returns to the Menu program (see above) which then proceeds to coordinate the printing process (see below).

5.4 Printing program

At the end of the anaesthetic all the relevant data (the Anaesthesia Record) is printed out in a form suitable for inclusion in the patient notes. The printing process (see Chapter 15) is initiated by pressing the F10 key (see above, and also Figure 5.2, page 50). Following printing all files are closed and the program terminates.

When F10 is pressed (see Figure 5.2, page 50) the Data Program creates a one-line batch file called `anes-prt.bat` (page 202) which holds a pointer to a second batch file (`printall.bat`, page 201) which actually coordinates (a) the processing of all the G-data files by `prtan3a.exe` (Chapter 17, page 217), and graph plotting (using GNUplot) and (b) the subsequent typesetting (using LaTeX 2ε) printing-out the hard copy which constitutes the Anaesthesia Record (Chapter 20 and Chapter 21).

Once the Data Program has been quit (F10), then command returns to the Menu program (see above) which now looks to see if the batch file `anes-prt.bat` exists, and if so, then it CALLs this and starts the printing process.

The subset of data which is used for printing the Anaesthesia Record (G-data, page 204) is stored in the files `.G01, .G02, ... .G0n`) which are output by the Data Program (Figure 5.1, page 49). This data is sampled approximately every 40 seconds from the incoming raw data (received every 10 seconds) from the Datex AS/3 anaesthesia monitor.

The processing of each `.Gnn` data file is coordinated by the program `plotan3a.pb` (page 217). The G-data is initially processed by the excellent and free GNUplot[3] utility to generate a number of separate `.dat` data files (page 231). Next all the necessary GNUplot `.gnu` scripts (page 238), are written and then run, generating the graphs in LaTeX 'picture' (`.pic`) format. Finally, these graphs are then typeset on a single page using LaTeX 2ε which generates a `.dvi` file. This file (which represents 1 hour of data) is then printed on a sheet of A4-paper. A typical example is shown in Figure 2.1 (page 13).

[3] `http://www.cs.dartmouth.edu/`; See also the Usenet group: `comp.graphics.apps.gnuplot`; *Nickalls & Ramasubramanian (1995)*, Appendix 7, page 369.

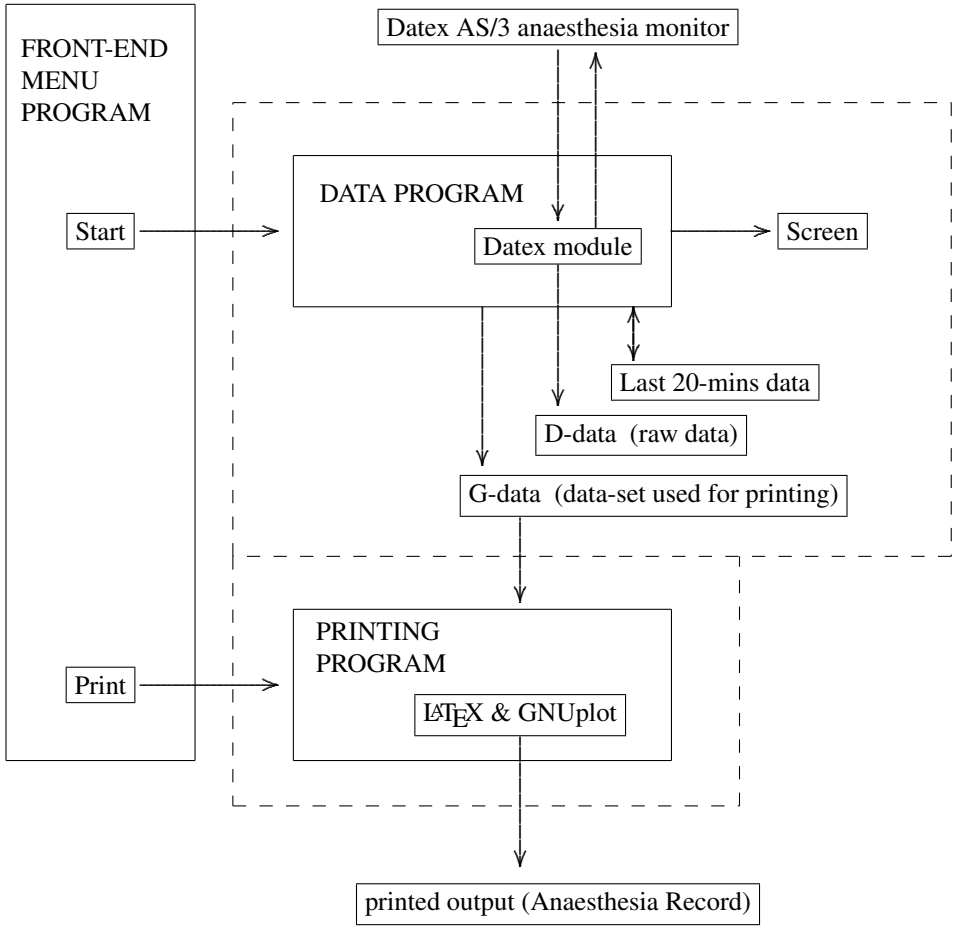


Figure 5.1: Overview of the Anaesthesia Record System programs. There are three main components (a) the front-end menu program, (b) the data program, and (c) the printing program. The data program is started by selecting from the pull-down menu. Printing proceeds either automatically after quitting the data program (by pressing F10 key), or by selecting a print option from the pull-down menu.

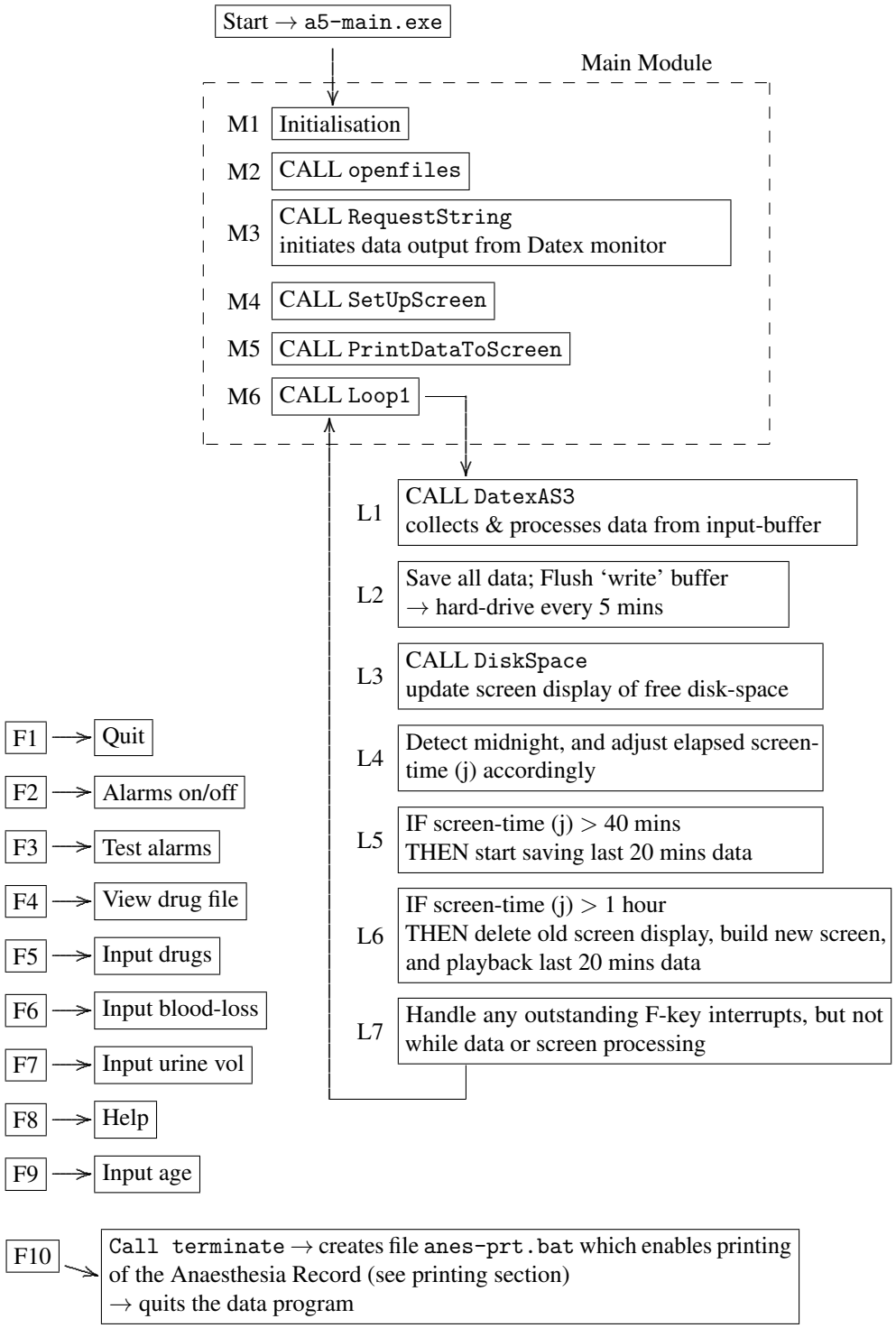


Figure 5.2: Overview of the data program. There are three components (a) the main module (M1–M6), (b) the data-collecting loop (L1–L7), and (c) F-key actions (F1–F10). The program is terminated by either pressing F1 (quit), or pressing F10 (prints out the Anaesthesia Record, and then quits).

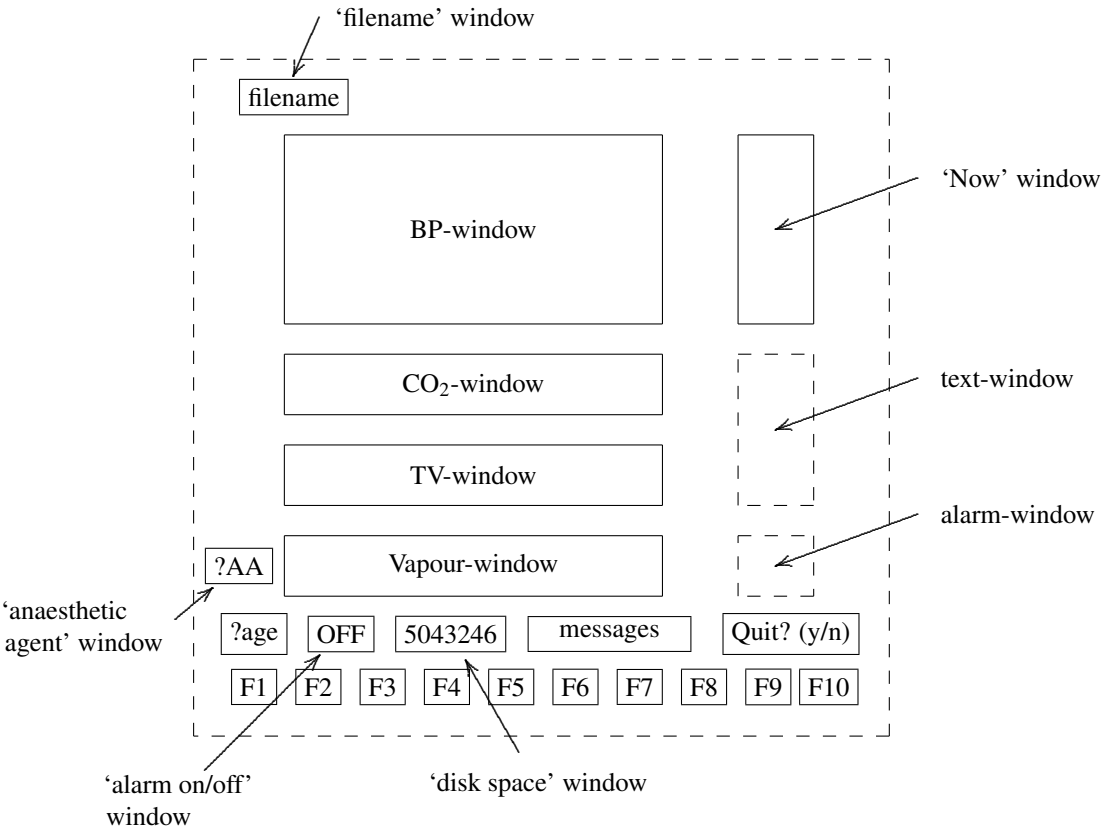


Figure 5.3: Location of screen graphic and text windows.

Part III

The front-end & menus

Chapter 6

Front-end batch-file (`menu-5a.bat`)

ch-menu.tex

6.1 Introduction

The menu-driven front-end for the Anaesthesia Record System is written using the proprietory enhanced batch language FromBAT v3 (1991) for MS-DOS (Clockwork Software, Bidbury House, Bidbury Lane, Havant, PO9-3JG, UK; tel 0705-483217; fax 0705-454233). FromBAT is version-2 of their BATtoCOM utility. The file `menu-5a.bat` was compiled to `menu-5a.com` using the command > `frombat menu-5a.bat`.

The front-end displays a number of pull-down menus and allows various parts of the system to be run separately. The menus are grouped into the following sections:

Run programs———-Print———-View/Copy data———-General info———-Quit/Setup

The current menus are as follows:

- Run programs
  - Datex AS/2 system (errorlevel 1)
  - Diet program (errorlevel 2)
  - Epidural/DL-tube info (errorlevel 3)
- Print
  - Print all anaes sheets for last case (errorlevel 21)
  - Print top sheet (errorlevel 22)
  - Print a single anaes sheet (errorlevel 23)
- View/Copy data
  - View trends (playback mode) (errorlevel 41)
  - View data-files (errorlevel 42)

  - View LaTeX log (errorlevel 43)
  - View Printer log (errorlevel 44)
  - Copy data-files to disk (errorlevel 45)
  - View printall.bat (errorlevel 46)

- General info
  - Serial protocol (errorlevel 61)
  - Wiring configuration (errorlevel 62)
  - RS-232-E standard (errorlevel 63)
  - 25/26-pin connector (errorlevel 64)
  - 9-pin connector (errorlevel 65)
  - Copyright information (errorlevel 66)
  - References (errorlevel 67)
  - Info on this package (errorlevel 68)

- Quit/Setup
  - Quit (errorlevel 81)
  - Setup (errorlevel 82)

6.2 Start-up

At start-up (see colour screen-shot at the beginning of this book), the option for launching the data program (`a5-main.exe`) is automatically highlighted, and so all the anaesthetist has to do at the beginning of an operation is to press the <CR> key. The following extract from the menu program shows that this immediately launches the Data Program `a5-main.exe` (Chapter 8, page 82).

Note that on exit from the Data Program control returns to the menu program, which then executes the batch file `anes-prt.bat` (if it exists) which coordinates the printing process.

```
CASE errorlevel = 1
     ....
     REM now START the ARS programme <a5-anes.exe>
     A5-main.exe
     ...
     REM now PRINT out the anaesthetic record
     REM if file <anes-prt.bat> has been written by main program
     IF EXIST anes-prt.bat
         CALL anes-prt.bat
         ....
     ENDIF
     ....
```

6.3 Quit option

The Quit option exits the menu-program and shuts the computer down, as follows:

```
CASE errorlevel = 81
        REM this is QUIT/ EXIT
        GOTO finish
....
....
:finish
  CLEAR, blue
  WINDOW SHUT
  RESTORECURSOR
  MOUSE
  CD \shutdown
  SHELL park
  STOP
```

6.4 The program

```
REM MENU-5a.BAT   FILE   -- for use with DeskJet printer for CITY HOSPITAL
REM for Datex AS/3
REM -----------------------------------
REM front-end menus for the anesthesia record keeping system
REM Developed using FROMBAT v3.0
REM RWD Nickalls (November 20, 1999)

COPYRIGHT MENU-5a.COM (c) RWD NICKALLS 1994-9

NODECODE

REM  allow for differend versions of MS-DOS
LOCALENVIRONMENT

REM now clear the screen
CLS

  WINDOW
  BOX OPEN

REM ***********************************************************
REM (= example of STUFF ) IF menucode = 25  STUFF &4d00  &5000
REM ***********************************************************

MOUSE OFF
REM start menu at TREND option (1)
LET menucode = 1
REM start the menus with the "Print topsheet" option (menucode 21)
REM LET menucode = 21
```

```
LET colourmode = colour

:start
REM write the name
  IF colourmode = mono
        CLEAR ,black
        COLOUR white,black
    ELSE
          CLEAR ,blue
          COLOUR black,white
    ENDIF

  BIG 8,4,pipe," Anaesthesia "
  BIG 16,13,pipe, " Record System 5a "

:menu
REM make menu return to its initial position: needs /p switch
REM this line must go just before the dropdown menu
    LET errorlevel = menucode

REM start the dropdown menus
  IF colourmode = mono
      COLOUR white, black
      SHADOW +white,green
  ELSE
      COLOUR white,red
      SHADOW +white,green
  ENDIF

 DROPDOWN /p /h " (c) RWD NICKALLS (1994-99)  5a-dj510 "
~Run~programs~   ~Print~   ~View/copy~data~   ~General~info~   ~Quit/setup~
 Datex~AS/3~system   BMI~info  Epidural/dl-tube~info
 print~ALL~anaes~sheets~for~last~case   print~Top-sheet
print~a~Single~anaes-sheet
 view~Trends~(playback~mode)  view~Data-files  view~Latex~log  view~Printer~log
                          copy~data-file~to~disk~in~B~drive  view~printAll.bat~file
 serial~Protocol  Wiring~configuration  RS-232-E~standard 25/26-piN~connector
             9-pin~Connector  coPyright~information  References Info~on~this~package
 Quit/exit Setup
 ENDDROPDOWN

REM ************************************************

DOCASE
 CASE errorlevel = 0
   REM <ESC> case
   REM make <ESC> ultimately go to the QUIT option
   LET menucode = 81
   BEEP 1,8
   GOTO menu
```

```
CASE errorlevel = 1
   REM Run the A5-anes.bas --> a5-main.pb (PB3.5) prog (dj510-PRINTER)
   REM the Datex AS/3 program A5-main.exe
   REM when prog ends then set menu to print ALL anes sheets (22)
   LET menucode = 1
    IF colourmode = mono
      CLEAR,black
      rem  this is the A5-main.exe programme( --> .exe)
      rem (renamed - used to be called a5-anes.exe)
      REM mode = colour/mono
      A5-main.exe
      CLEAR,black
      CURSOR off
      GOTO start
   ELSE
     REM ie = colour mode
     CLEAR,blue
     REM  now run the ARS programme( --> .exe)
     A5-anes.exe
     REM on exit then print out the anaesthetic sheets
      CLEAR,blue
      IF EXIST anes-prt.bat
          CALL anes-prt.bat
   REM       DEL anes-prt.bat
          CLEAR, blue
          ENDIF
      CURSOR off
      GOTO start
 ENDIF


CASE errorlevel = 2
   REM Diet program
   REM Run the anes-2c prog (dj510-PRINTER)
   REM the DATEX Capnomac & Ultima (II series) program anes-2c.exe
   REM when prog ends then set menu to print ALL anes sheets (22)
   LET menucode = 1
    IF colourmode = mono
      CLEAR,black
      REM mode = mono
      REM run DIET program from diet.com
      diet.com
      CLEAR,black
      CURSOR off
      GOTO start
   ELSE
     REM ie = colour mode
     CLEAR,blue
     REM run DIET program from diet.com
```

```
      diet.com
         CLEAR, blue
         ENDIF
       CURSOR off
       GOTO start
  ENDIF


 CASE errorlevel = 3
    REM the Thoracic Epidural program (thorn10.exe)
    LET menucode = 2
    IF colourmode = mono
       CLEAR,black
       rem  this is the EPIDURAL programme( --> thorn10.exe)
       REM mode = colour/mono
    Thorn10.exe
       CLEAR,black
       CURSOR off
       GOTO start
    ELSE
      REM ie = colour mode
      CLEAR,blue
      REM  now run the epidural (THORN10 ) programme( --> .exe)
      thorn10.exe
           CLEAR, blue
           ENDIF
       CURSOR off
       GOTO start
  ENDIF


REM ************************************************


 CASE errorlevel = 21
    REM print out all the last case Record sheets (DeskJet printer)
    REM by calling the batch file PrintAll.bat
    REM when prog ends then set menu to 1
    LET menucode = 1
    IF colourmode = mono
       CLEAR,black
       COLOUR black,white
    ELSE
      CLEAR,blue
      COLOUR blue,white
    ENDIF
     IF not exist printall.bat
         BEEP
         PUT 5,, " I can't find any files to print! "
         PUT 7,, " Press any key to return "
```

```
         LOCATE 12,2
         PAUSE
         CLEAR,blue
         POPDIR
         CURSOR off
         GOTO start
       ENDIF
      CALL PrintAll.bat
      CURSOR off
      GOTO start


CASE errorlevel = 22
    REM print top-sheet
     REM make the menu go to the correct program after this
     LET menucode = 1
    IF colourmode = mono
       CLEAR,black
       COLOUR black,white
    ELSE
      CLEAR,blue
      COLOUR blue,white
  ENDIF
REM ******** use if no printer ******
REM   PUT 22,2 Press <ESC> to QUIT
REM   BOX INTERNAL 4,22,33,1,1
REM   PUT 5,, " sorry -- printer not available "
REM   GET 0
REM   GET
REM   CURSOR off
REM   GOTO start
REM *********************************
   REM go to \emtex and run latex prt-tops.tex
   BOX INTERNAL 2,22,33,1,1
   PUT 3,, " typesetting TOPSHEET with LaTeX "
   LOCATE 8,1
   PUSHDIR
   REM topsheet.tex and .dvi is in \emtex\texinput
       REM now printout the topsheet.dvi file from emtex
    CLEAR,blue
    COLOUR blue,white
    BOX INTERNAL 2,21,36,1,1
    PUT 3,, " printing TOPSHEET to DeskJet-510 "
    LOCATE 8,1
       CHDIR c:\emtex\texinput
    REM now print to the DeskJet 510 printer
    REM prthplj /od+ topsheet.dvi prn
    REM copy topsheet.jet /b prn
    copy topsheet.djt /b prn
    REM return to orig Directory
```

```
     POPDIR
     CURSOR off
     GOTO start


CASE errorlevel = 23
        REM print a single  DATA-FILE (anes sheet)  to PRINTER
        LET menucode = 23
     IF colourmode = mono
        CLEAR,black
        rem  PROG HERE
        rem *******

        CLEAR,black
        CURSOR off
        GOTO start
     ELSE
       CLEAR,blue
       rem  viewing the data
       REM prog here
       REM save the current directory info
       PUSHDIR
       CD \qb45\rs232\theatre
       PUT 1,2 Select a data-file to PRINT-OUT as an anaesthetic sheet
       PUT 2,2 Takes approximately 2 mins  to process
       PUT 22,2 Press <ESC> to QUIT
       WINDOW 4,20,45,15
       BOX open
       WINDOW 5,21,43,13
         REM select files with .Gxx extension in name order (/N)
         REM do not select the .GNU ones, ie use .G01 etc
         filename *.G0*, *.G1*, *.G2*  /N
         IF  errorlevel = 0
             REM close windows and look at file
             WINDOW CLOSE
             WINDOW SHUT
             CLEAR,BLUE
             REM remember to fill in the parameters for VIEW
             REM view 1 25 2 gre blu lgy %filename%
             REM now plot the data using plotan8C.exe
             plotan8C %filename%
             POPDIR
             CURSOR off
             GOTO start
         ENDIF
         WINDOW CLOSE
  WINDOW SHUT
  REM delete writing on main screen
  CLEAR,BLUE
  REM return to the original directory
```

```
  POPDIR
  CURSOR off
  GOTO start
ENDIF

REM ******** section for VIEW DATA-FILES **********************


CASE errorlevel = 41
        REM DEMO mode
        REM VIEW DATA-FILES to select a file
        LET menucode = 41
     IF colourmode = mono
        CLEAR,black
        rem  PROG HERE
        rem *******

        CLEAR,black
        CURSOR off
        GOTO start
     ELSE
       REM ie = colour
       CLEAR,blue
       rem  viewing the data
       REM prog here
       rem ****
       REM save the current directory info
  PUSHDIR
REM   CD \qb45\rs232\theatre
  PUT 1,2 Select a data-file to view
  PUT 22,2 Press <ESC> to QUIT
  WINDOW 4,20,45,15
  BOX open
      WINDOW 5,21,43,13
         REM select files with .Hxx extension in Date order (/D)
         REM select files with .Hxx extension in Alphabet order (/N)
         FILENAME *.H* /N
         IF  errorlevel = 0
             REM ie a file HAS been selected
             REM close windows and look at file
             WINDOW CLOSE
             WINDOW SHUT
             CLEAR,BLUE
             rem now run the program
             REM remember to fill in the parameters for VIEW
             REM now call the DEMO-2c program
             demo-2c.exe %filename%
             POPDIR
             CURSOR off
             GOTO start
```

```
        ENDIF
        WINDOW CLOSE
  WINDOW SHUT
  REM delete writing on main screen
  CLEAR,BLUE
  REM return to the original directory
  POPDIR
  CLEAR, blue
  CURSOR off
  GOTO start
ENDIF


 CASE errorlevel = 42
        REM VIEW DATA-FILES
        LET menucode = 42
     IF colourmode = mono
        CLEAR,black
        rem  PROG HERE
        rem *******

        CLEAR,black
        CURSOR off
        GOTO start
     ELSE
       CLEAR,blue
       rem  viewing the data
       REM prog here
       rem ****
       REM save the current directory info
  PUSHDIR
REM   CD \qb45\rs232\theatre
  PUT 1,2 Select a data-file to view
  PUT 22,2 Press <ESC> to QUIT
  WINDOW 4,20,45,15
  BOX open
      WINDOW 5,21,43,13
         REM select files with  in name order (/N)
         REM select all the .H* ones  and *.g and *.DRG
         filename printall.bat, *.DRG, *.G*,*.H*,*.DAT  /N
         IF  errorlevel = 0
             REM close windows and look at file
             WINDOW CLOSE
             WINDOW SHUT
             CLEAR,BLUE
             REM remember to fill in the parameters for VIEW
             view 1 25 2 gre blu lgy %filename%
             POPDIR
             CURSOR off
             GOTO start
```

```
        ENDIF
        WINDOW CLOSE
  WINDOW SHUT
  REM delete writing on main screen
  CLEAR,BLUE
  REM return to the original directory
  POPDIR
  CURSOR off
  GOTO start
ENDIF


CASE errorlevel = 43
        REM VIEW LaTeX LOG FILE  (\emtex\texinput\prt-anes.log)
        LET menucode = 43
     IF colourmode = mono
        CLEAR,black
        rem  PROG HERE
        rem *******
        CLEAR,black
        CURSOR off
        GOTO start
     ELSE
       CLEAR,blue
     ENDIF
     REM save the current directory info
     PUSHDIR
     REM  Now view the data file
     CD \emtex\texinput
     REM remember to fill in the parameters for VIEW
     view 1 25 2 gre blu lgy prt-anes.log
     REM now return to the original directory and exit
     POPDIR
     CURSOR off
     GOTO start


 CASE errorlevel = 44
        REM VIEW LaTeX PRINTER (dvi) LOG FILE (\emtex\texinput\dvihplj.dlg)
        LET menucode = 44
     IF colourmode = mono
        CLEAR,black
        rem  PROG HERE
        rem *******
        CLEAR,black
        CURSOR off
        GOTO start
     ELSE
       CLEAR,blue
     ENDIF
```

```
    REM save the current directory info
    PUSHDIR
    REM  Now view the data file
    CD \emtex\texinput
    REM remember to fill in the parameters for VIEW
    view 1 25 2 gre blu lgy dvihplj.dlg
    REM now return to the original directory and exit
    POPDIR
    CURSOR off
    GOTO start


CASE errorlevel = 45
       REM COPY  DATA-FILES   to DISK B-drive
       LET menucode = 45
    IF colourmode = mono
       CLEAR,black
       rem  PROG HERE
       rem *******
       CLEAR,black
       CURSOR off
       GOTO start
    ELSE
      CLEAR,blue
    REM  viewing the data files to copy
      REM prog here
      REM save the current directory info
      PUSHDIR
REM      CD \qb45\rs232\theatre
      PUT 2,2 Select a data-file to COPY to the disk in drive B
      PUT 22,2 Press <ESC> to QUIT
      WINDOW 5,20,45,15
      BOX open
      WINDOW 6,21,43,13
        REM select files with .Gxx extension in name order (/N)
        REM do not select the .GNU ones, ie use .G01 etc
   :linestart
        filename *.G0*, *.G1*, *.G2*  /N
        IF  errorlevel = 0
            REM now copy the file using CPY as it will not  copy
            REM need to check disk and size/ free space etc
            REM if disk too small
            SHELL copy c:\qb45\rs232\theatre\%filename% b:\%filename%
            REM put cursor at top again
            CURSOR 0,0
            GOTO linestart
        ENDIF
        WINDOW CLOSE
        WINDOW SHUT
        REM delete writing on main screen
```

```
         CLEAR,BLUE
         REM return to the original directory
         POPDIR
         CURSOR off
         GOTO start
   ENDIF


CASE errorlevel = 46
        REM VIEW printall.bat file
        REM ? this is written to the emtex\texinput dir
        LET menucode = 46
     IF colourmode = mono
        CLEAR,black
        rem  PROG HERE
        rem *******
        CLEAR,black
        CURSOR off
        GOTO start
     ELSE
       CLEAR,blue
     ENDIF
     REM save the current directory info
     PUSHDIR
     REM  Now view the data file
     REM CD \emtex\texinput
     REM CLEAR,BLUE
     REM remember to fill in the parameters for VIEW
     view 1 25 2 gre blu lgy printall.bat
     REM now return to the original directory and exit
     POPDIR
     CURSOR off
     GOTO start

REM ***********************************************

CASE errorlevel = 61
      rem view serial PROTOCOL
        LET menucode = 61
     IF colourmode = mono
        CLEAR,black
        COLOUR black,white
     ELSE
       CLEAR,blue
       COLOUR blue,white
   ENDIF
     BOX INTERNAL 2,27,23,1,1
     PUT 3,, " DATEX SERIAL PROTOCOL "
     WINDOW 9,10,60,10
     CLEAR , white
```

```
     PUT
     PUT
     PUT ,18, Bit rate (Baud)  1200
     PUT ,18, Data bits        8
     PUT ,18, Parity           None
     PUT ,18, Stop bits        1
     GET 0
     GET
     WINDOW SHUT
     CURSOR off
     GOTO start


CASE errorlevel = 62
      rem view theWiring configuration
        LET menucode = 62
     IF colourmode = mono
        CLEAR,black
        COLOUR black,white
     ELSE
       CLEAR,blue
       COLOUR blue,white
   ENDIF
     BOX INTERNAL 2,28,22,1,1
     PUT 3,, " WIRING CONFIGURATION "
     WINDOW 7,10,60,15
     CLEAR , white
     PUT
     PUT
     PUT ,8, ULTIMA      CARDIOCAP       COMPUTER serial port
     PUT ,8, ------      ---------       --------------------
     put
     PUT ,9, TxD  -----> RxD
     PUT ,22,             TxD  -------------> RxD
     PUT ,9, GND  ------  GND  -------------- GND
     GET 0
     GET
     WINDOW SHUT
     CURSOR off
     GOTO start


CASE errorlevel = 63
      rem view the RS232 standard chapter
      LET menucode = 63
     IF colourmode = mono
        CLEAR,black
        COLOUR black,white
     ELSE
       CLEAR,blue
```

```
       COLOUR blue,white
   ENDIF
      BOX INTERNAL 2,24,30,1,1
      PUT 3,, " THE RS-232-E STANDARD (1991) "
      WINDOW 7,5,70,15
      CLEAR , white
      SHOW rs232inf.doc
      WINDOW shut
      CURSOR off
      GOTO start


CASE errorlevel = 64
      rem view the 25-pin connector
      LET menucode = 64
     IF colourmode = mono
        CLEAR,black
        COLOUR black,white
     ELSE
       CLEAR,blue
       COLOUR blue,white
   ENDIF
      BOX INTERNAL 2,28,21,1,1
      PUT 3,, " 25/26-PIN CONNECTOR "
      WINDOW 7,5,70,15
      CLEAR , white
      SHOW 25-pin.doc
      WINDOW shut
      CURSOR off
      GOTO start


CASE errorlevel = 65
      rem view the 9-pin connector
        LET menucode = 65
     IF colourmode = mono
        CLEAR,black
        COLOUR black,white
     ELSE
       CLEAR,blue
       COLOUR blue,white
   ENDIF
     BOX INTERNAL 2,27,24,1,1
     PUT 3,, " RS-232 9-PIN CONNECTOR "
     WINDOW 7,10,60,15
     CLEAR , white
     PUT ,12,   PIN No              FULL NAME
     PUT ,12,   ------------------------------------
     PUT ,12,   pin 1    DCD     Data carrier detect
     PUT ,12,   pin 2    RxD     Receives data
```

```
     PUT ,12,   pin 3    TxD     Transmits data
     PUT ,12,   pin 4    DTR     Data terminal ready
     PUT ,12,   pin 5    GND     Signal ground
     PUT ,12,   pin 6    DSR     Data set ready
     PUT ,12,   pin 7    RTS     Request to send
     PUT ,12,   pin 8    CTS     Clear to send
     PUT ,12,   pin 9    RI      Ring indicator
     GET 0
     GET
     WINDOW SHUT
     CURSOR off
    GOTO start


CASE errorlevel = 66
     rem view the Copyright info
     LET menucode = 66
     IF colourmode = mono
        CLEAR,black
        COLOUR black,white
     ELSE
       CLEAR,blue
       COLOUR blue,white
   ENDIF
     BOX INTERNAL 2,33,11,1,1
     PUT 3,, " COPYRIGHT "
     WINDOW 7,10,60,15
     CLEAR , white
     PUT 1,, Copyright (c) RWD Nickalls (1994-1997)
     PUT
     PUT ,2, Dr RWD Nickalls, BSc, PhD, MBBS, FRCA
     PUT ,2, Department of Anaesthesia,
     PUT ,2, Nottingham City Hospital NHS Trust,
     PUT ,2, Nottingham, UK.
     PUT
     PUT ,2, email (INTERNET) dick.nickalls@nottingham.ac.uk
     PUT ,2, email (INTERNET) 100115.1010@CompuServe.com
     PUT
     PUT ,2, Tel:    +44-(0)115-9691169
     PUT ,2, Fax:    +44-(0)115-9627713
     GET 0
     GET
     WINDOW SHUT
     CURSOR off
     GOTO start


CASE errorlevel = 67
      rem view the REFERENCES
        LET menucode = 67
```

```
     IF colourmode = mono
        CLEAR,black
        COLOUR black,white
     ELSE
       CLEAR,blue
       COLOUR blue,white
   ENDIF
     BOX INTERNAL 2,33,12,1,1
     PUT 3,, " REFERENCES "
     WINDOW 7,10,60,13
     CLEAR , white
     PUT
     PUT ,2, Nickalls RWD and Ramasubramanian R (1995).
     PUT
     PUT ,7, The Datex CARDIOCAP and ULTIMA series
     PUT ,7, of monitors.
     PUT
     PUT ,7, In: Interfacing the IBM-PC to medical equipment:
     PUT ,11,    the art of serial communication.
     PUT ,11,    (Cambridge University Press, UK).
     GET 0
     GET
     WINDOW SHUT
     CURSOR off
     GOTO start


CASE errorlevel = 68
      rem view the Info-about-this-package (info.doc)
      LET menucode = 68
     IF colourmode = mono
        CLEAR,black
        COLOUR black,white
     ELSE
       CLEAR,blue
       COLOUR blue,white
   ENDIF
      BOX INTERNAL 2,24,30,1,1
      PUT 3,, " Info on this package "
      WINDOW 7,5,70,15
      CLEAR , white
      SHOW info.doc
      WINDOW shut
      CURSOR off
      GOTO start

REM ***********************************************


CASE errorlevel = 81
```

```
        REM this is QUIT/ EXIT
        GOTO finish


CASE errorlevel = 82
   REM this is the small colour/mono window prog
   LET menucode = 82
   REM        2          12       19    7
   REM window top row /left col /wide/deep
      WINDOW 4,        57,       19,   7
      CLEAR white,red

REM box open row/col/width/ depth/border
   BOX OPEN  0,   1, 17,     7,    2
   REM put row/col/string
       PUT 1,  2,  " MONO screen "
       PUT 2,  2, " COLOUR screen "
       PUT 3,  2,  " mouse ON "
       PUT 4,  2, " mouse OFF "
       PUT 5,  2, " EXIT "

   REM make the shadow start at the top
    LET errorlevel = 1
   REM empty buffer before using POINT
     GET 0
  REM make a small menu table using POINT
  REM point row/col/width/depth/across/waitfor secs/gap/ primw
      POINT  1, 2,  15,    5,,           10,

  DOCASE
       CASE errorlevel = 0
          REM <ESC> case
           window close
            GOTO start

       CASE errorlevel = 1
            REM mono case
            LET colourmode = mono
            WINDOW CLOSE
            GOTO start

       CASE errorlevel = 2
           REM colour case
           LET colourmode = colour
           window close
           goto start

    CASE errorlevel = 3
         REM mouse-on case
         MOUSE
```

```
         window close
            goto start

     CASE errorlevel = 4
       REM mouse-off case
       MOUSE OFF
       window close
      goto start

      CASE errorlevel = 5
           REM exit case
           WINDOW CLOSE
           GOTO start
 ENDCASE



CASE errorlevel = 83
     rem  go to SHELL
     LET menucode = 83
     IF colourmode = mono
        CLEAR,black
        COLOUR black,white
     ELSE
       CLEAR,blue
       COLOUR blue,white
   ENDIF
     BOX INTERNAL 2,30,17,1,1
     PUT 3,, " CALLING PCSHELL "
     PUSHDIR
     SHELL pcshell
     POPDIR
     CURSOR off
     GOTO start

REM -------------------------
REM this is the final endcase
ENDCASE


:finish
  CLEAR, blue
  WINDOW SHUT
  RESTORECURSOR
  MOUSE
  CD \shutdown
  SHELL park
  STOP

REM ----------------- end ----------------------
```

Part IV

The data program

Chapter 7

Overview

ch-dpin.tex

7.1 Introduction

The overall structure of the Data Program is shown in Figure 7.1 (page 75) The Data Program can be viewed as consisting of three functional parts, as follows.

- The main module, M1–M6 (Chapter 8)
- The loop section, L1–L7 (Chapter 9)
- The F-key section, F1–F10 (Chapter 10)

The main module (`a5-main.pb`, page 102) initialises the system (creating filenames, opening the input and output files etc), and finally calls the SUB `LoopOne` (page 133), which is a `DO-LOOP` for coordinating (a) data access from the Datex AS/3 anaesthesia monitor, (b) writing trend data to the screen, (c) handling F-key inputs, (d) saving data to the harddrive, (e) other general housekeeping (eg updating the screen clock, cycling the screen at hourly intervals etc.

All incomming raw data from the Datex AS/3 anaesthesia monitor is accessed via the serial port (COM port), processed, and then displayed graphically in real-time on the screen. All the raw data (D-data) is saved by writing it to the hard drive, and a modified version of the data (G-data), which is used for printing out using the GNUplot and TEX systems, is saved in separate files each containing 1 hour of data. Error messages are written to the screen on a line at bottom of screen, and are visible for a variable period of time, depending on the significance or importance of the error.

At the end of the anaesthetic all the relevant data (the Anaesthesia Record) is printed out in a form suitable for inclusion in the patient notes. This process (see section VI) is initiated by pressing one of the F-keys (F10). Following printing all files are closed and the program terminates.

7.2 PowerBasic program

The compiled data program is a DOS PowerBasic 3.5 executable (A5-main.exe; 135 KB). The program was originally written in QuickBasic 4.5, but ported to PowerBasic

in order to accommodate the different serial port protocol (e.g. 11-bit character frame) used by the more recent Datex AS/3 anaesthesia monitor.

The data program consists of a main module (`a5-main.pb`, page 102) and six files (see end of chapter for listing) of subroutines which are included at compile time, using the `$INCLUDE` command towards the end of the main module, as follows

```
$INCLUDE  "a5-anes2.pb"
$INCLUDE  "a5-draw.pb"
$INCLUDE  "a5-lib.pb"
$INCLUDE  "a5-keyf.pb"
$INCLUDE  "a5-plot.pb"
$INCLUDE  "a5-DXas3.pb"
```

The data program was compiled using the following command.

```
PBC /ce  a5-main.pb
```

7.3 Main module (`a5-main.pb`)

The main module consists of six functional parts M1–M6 as shown in Figure 7.1 (page 75). The subroutines CALLed by the main module are as follows.

`Openfiles`	(page 138) opens all input and output files
`RequestString`	(page 195) enables dataoutput from the Datex AS/3 monitor
`SetupScreen`	(page 173) initialises and setsup the screen
`PrintDataToScreen`	(page 158) writes the labels to the screen
`LoopOne`	(page 133) collects and processes all the input data
`Terminate`	(page 175) closes all open files and shutsdown program
`KeyHandler`	(page 122) processes all F-key activity

The various functional components of the main module are detailed in Chapter 8 (page 82).

7.4 LoopOne

See L1–L7 (Figure 7.1, page 75). The SUB `LoopOne` (page 133) is the main working loop which collects all the data and coordinates all the necessary housekeeping. The program stays in this loop for the whole anaesthetic, until it is quit (press F1), or printing the data record is initiated (press F10).

The SUB `LoopOne` calls two 'housekeeping' SUBS, namely `LoopTwo` (page 135), and SUB `HouseKeeping` (page 120). The SUB `LoopTwo` is really an extension of `LoopOne`, and is kept as a separate SUB purely for convenience. The SUB `HouseKeeping`, which is only called every 10 seconds, coordinates saving the last 20 minutes of screen data (for playing back to the next screen) and triggering the building of a new screen at the end of each hour. These three SUBs together coordinate the activities L1–L7.

7.5 F-keys

See Figure 8.2 (page 87). Pressing an F-key usually sets a FLAG which is then actioned by the program at the next available suitable time (i.e. when the current job has been

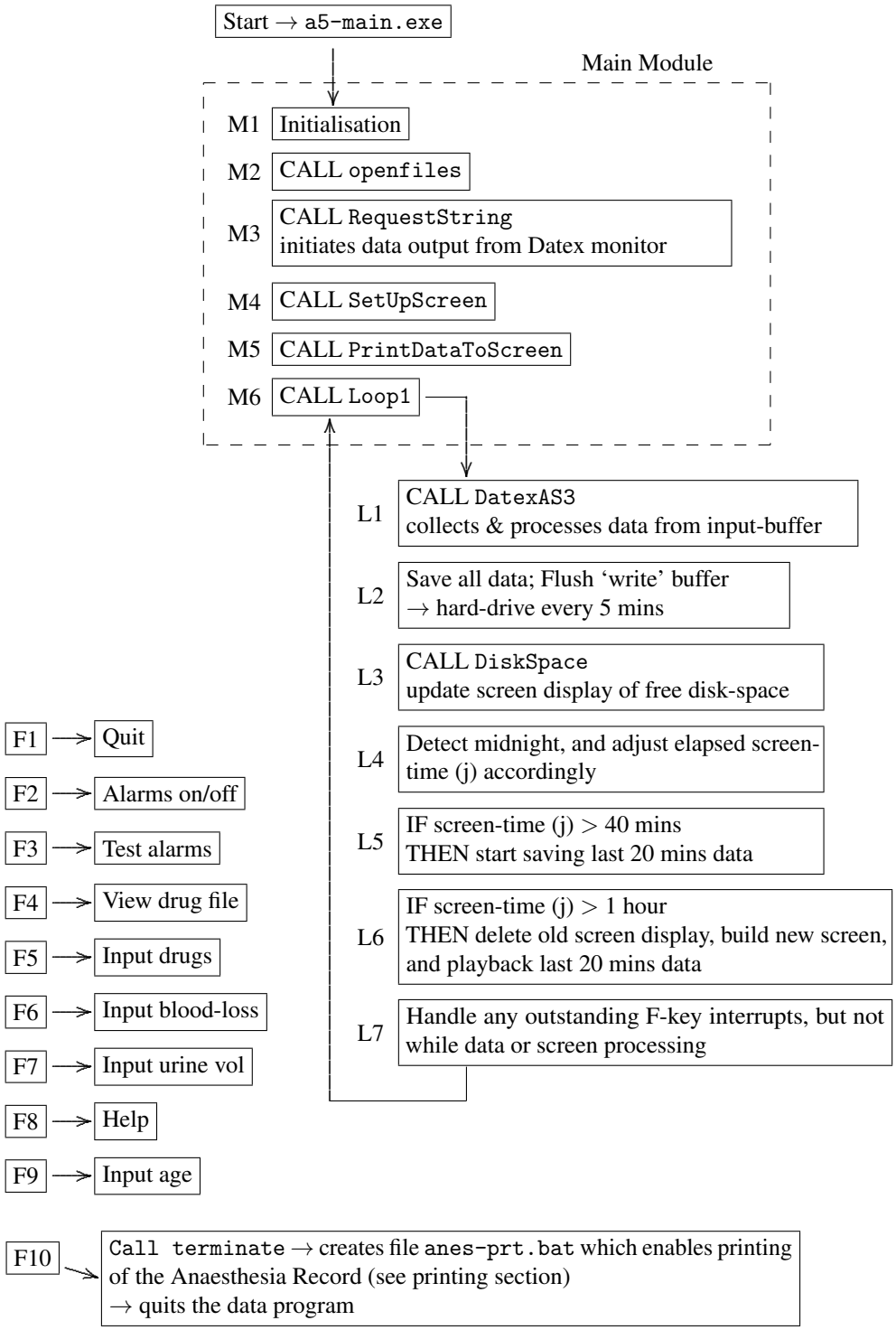


Figure 7.1: Overview of the data program. There are three components (a) the main module M1–M6 (Chapter 8, page 82), (b) the data-collecting loop L1–L7 (Chapter 9, page 90), and (c) F-key actions F1–F10 (Chapter 10, page 96). The program is terminated by either pressing F1 (quit), or pressing F10 (prints out the Anaesthesia Record, and then quits).

completed). FLAG status is checked for at several places in SUB `LoopOne` (page 133), and also in the subroutines called by SUB `LoopTwo` (page 135).

7.6 Classification of subroutines

The various subroutines can be grouped according to function as follows.

7.6.1 Data storage

`SaveGNUdata`	(page 163) saves GNUplot data (used for printing out)
`SaveLastHourData()`	(page 165) saves last 20 mins data (for replaying back)
`SaveDrugData()`	(page 162) saves drug data (input) to drugfile (for printing)

7.6.2 Screen setup/maintenance

`SetupScreen`	(page 173) coordinates settingup of the screen & windows
`ScreenBOFF`	(page 167) saves the bottom half of the screen
`ScreenBON`	(page 167) replaces the bottom half of the screen
`Screen9Save`	(page 167) saves the whole screen to memory
`Screen9Restore`	(page 166) replaces the whole screen from memory
`PrintLast20minsFast`	(page 159) plots the last 20 mins data to screen
`PrintNewScreen`	(page 160) coordinates making a new screen and 20min replay

7.6.3 Window setup/maintenance

These subs draw the various windows to the screen when the screen is refreshed at the end of each hour (mostly called by SUB `SetupScreen`, page 173).

`DrawBPwindow`	(page 115)
`DrawBPwindowHlines`	(page 116)
`DrawCO2window`	(page 116)
`DrawHlinesBPnow`	(page 117)
`DrawNowWindow`	(page 117)
`DrawTimeMarks`	(page 118)
`DrawTVwindow`	(page 119)
`DrawVapourWindow`	(page 120)
`PrintDataToScreen`	(page 158) updates some text data to the screen

7.6.4 Plotting data in windows

These SUBs plot new data to the relevant window. They are called by SUB `decode()` (page 189) which is part of the Datex AS/3 module (Chapter 13, page 178). The 'Now' in the name refers to the small Now-window in which the latest values are plotted separately.

7.6.5 File handling

7.6.6 Alarm handling

7.6.7 Datex AS/3 monitor I/O (Chapter 13, page 178)

7.6.8 F-key handling (Chapter 10, page 96)

7.6.9 Miscellaneous

`LoopTwo`	(page 135) does some general housekeeping (called by SUB `LoopOne`)
`Housekeeping`	(page 120) saves data & housekeeping jobs (called by SUB `LoopOne`)
`Message()`	(page 138) prints a message to the screen
`PrintSubname`	(page 162) prints the subroutine name to screen (for debug purposes)
`Terminate`	(page 175) shuts down program, and initiates printing
`MAC()`	(page 136) calculates age-corrected MAC value and prints to screen

7.7 Alphabetical list of subroutines and functions

```
Alarms
AlarmSound
AlarmTester
Byte1? (n1%)
Bytes2?? (n1%,n2%)
bytes4?? (n1%,n2%,n3%,n4%)
ClearNIBPnow
DatexAS3
Decode (ddata$)
DiskSpace (bytes#)
DrawBPwindow
DrawBPwindowHlines
DrawCO2window
DrawHlinesBPnow
DrawNowWindow
DrawTimeMarks
DrawTVwindow
DrawVapourWindow
Fixstring$ (bad$)
Housekeeping
KeyHandler
KeyScreenHandler (k$)
label?? (n1%,n2%)
Long& (n1%,n2%,n3%,n4%)
LoopOne
LoopTwo
MAC (N2Opercent, vapourname$, etvapour, ageofpatient%, bmac)
Message
OpenFiles
PlotBP
PlotBPdataNow
PlotCO2
PlotFast
PlotFIO2
PlotNIBP
PlotNIBPdataNow
PlotOximDataNow
PlotRR
PlotSAT
```

```
PlotTidalVol
PlotTrendData
PlotVapour
PrintDataToScreen
PrintLast20minsFast
PrintNewScreen
PrintSubname
RequestString
SaveAS3Data (ddata$)
SaveDrugData (druginfo$)
SaveGNUdata
SaveLastHourData
Screen9Restore
Screen9Save
ScreenBOFF
ScreenBON
Screendrug
ScrollFile (filetoscroll$, r1&, r2&)
SetupScreen
Short% (n1%,n2%)
Status??? (n1%,n2%,n3%,n4%)
Terminate
```

7.8 Allocation of SUBs to files

All the SUBs are currently collected into six files simply for convenience. The six files are loaded using the `$INCLUDE` command in the main module. The contents of the six files are as follows.

a5-anes2.pb some miscellaneous SUBs.

```
Alarms
AlarmSound
AlarmTester
Housekeeping
LoopOne
LoopTwo
Message
OpenFiles
PrintSubname
SaveGNUdata
SaveLastHourData
```

a5-draw.pb SUBs for maintaining the screen

```
DrawBPwindow
DrawBPwindowHlines
DrawCO2window
```

```
DrawHlinesBPnow
DrawNowWindow
DrawTimeMarks
DrawTVwindow
DrawVapourWindow
PrintDataToScreen
```

a5-keyf.pb SUBS for handling the F-keys

```
KeyHandler
KeyScreenHandler (k$)
SaveDrugData (druginfo$)
Screendrug
Terminate
```

a5-lib.pb standalone SUBs

```
DiskSpace (bytes#)
ScreenBOFF
ScreenBON
Screen9Save
Screen9Restore
ScreenRESTORE
ScreenSAVE
ScrollFile (filetoscroll$, r1&, r2&)
MAC (N2Opercent, vapourname$, etvapour, ageofpatient%, bmac)
```

a5-plot.pb SUBs for plotting data in the screen windows

```
ClearNIBPnow
PlotBP
PlotBPdataNow
PlotCO2
PlotFast
PlotFIO2
PlotNIBP
PlotNIBPdataNow
PlotOximDataNow
PlotRR
PlotSAT
PlotTidalVol
PlotTrendData
PlotVapour
PrintLast20minsFast
PrintNewScreen
SetupScreen
```

a5-DXas3.pb the Datex AS/3 module—SUBS for interfacing to the Datex AS/3 monitor, and for processing the Datex data.

```
DatexAS3
FixString$ (bad$)
SaveAS3Data (ddata$)
Decode (ddata$)
RequestString
Short% (n1%,n2%)
Long& (n1%,n2%,n3%,n4%)
Bytes2?? (n1%,n2%)
Bytes4??? (n1%,n2%,n3%,n4%)
Byte1? (n1%)
Label?? (n1%,n2%)
Status??? (n1%,n2%,n3%,n4%)
```

Chapter 8

Main module

ch-dpmm.tex

8.1 Introduction (`a5-main.pb`)

The main module consists of six functional parts M1–M6 as shown in Figure 8.1 (page 84). The subroutines CALLed by the main module are as follows.

`Openfiles` (page 138)	opens all input and output files
`RequestString` (page 195)	enables data output from the Datex AS/3 monitor
`SetupScreen` (page 173)	initialises and sets up the screen
`PrintDataToScreen` (page 158)	writes the labels to the screen
`LoopOne` (page 133)	collects and processes all the input data
`Terminate` (page 175)	closes all open files and shuts down program
`KeyHandler` (page 122)	processes all F-key activity

The various functional components of the main module (listed M1–M6 in Figure 8.1, page 84), are now described in turn.

8.2 Initialisation (M1)

1. Define and allocate DIM and SHARED variables/arrays
2. Allocate space for ScreenSave and ScreenRestore commands
3. Define mode strings for use as FLAGS
4. Set the COM port buffer size
5. Allocate version name/number to `version$`
6. Set screen variables for the various plotting windows
7. Setup the F-keys
8. Enable error-trapping and error handler
9. `$INCLUDE` all additional files and SUBs.

8.3 Open files (M2)

The setting up and opening of the COM port (COM1), and of all necessary files is coordinated by the SUB `OpenFiles` (page 138). Some files have a time/date-encoded `<filename>`. The list of files opened is as folows—details of the particular files and their formats is given in Chapter 18 (page 231).

`<filename>.Dnn`	for holding all raw data (page 203, page 203)
`<filename>.Gnn`	for holding data for printing (page 203, page 204)
`<filename>.drg`	for holding drug data (page 203, page 206)
`LastHour.dat`	for data from last 20 mins (page 206)
`Anes-prt.bat`	used to initiate printing (page 202)
`PrintAll.bat`	used to coordinate printing (page 201)

Note that the maximum COM port buffer size is set as a PowerBasic meta-statement `$COM 30000`, early in the main module before any executable statements. The steps covered by the SUB `OpenFiles` (page 138) are now detailed in order.

Create a time/date-encoded filename string (`timename$`)

The filename (page 203) is a 8-alpha-numeric string in the sequence [year][month][day][hour][minutes], allocate as follows: year (last two digits of year), month (one alpha-numeric character 0–9, A–C), day(two digits 00–31), hour (one alpha-numeric character 0–9, A–N), minutes (two digits 00–59). For example, the filename for a record started at 14.34 hrs on 26 November 2001 would be 01B26E34;
i.e. 01 B 26 E 34

Open COM1 for data input from the Datex AS/3 monitor

The serial protocol for the Datex AS/3 monitor (see Chapter 3, page 14) is as follows:– bit rate (19200), parity (N), data bits (8), stop bits (1). The PowerBasic codes `CS` and `DS` used here automatically disable the CTS and DTS timeouts, and so allow communications to proceed without first checking the computer's serial port CTS and DTS status. These communications issues are more fully dealt with in Chapter 3 and also in the Datex program module Chapter 13, page 178 (see also Chapter 5 in Nickalls and Ramasubramanian, 1995)[1]. For example, if we wish to allocate the file number 1 to COM port 1, then the complete PowerBasic command for connecting to the Datex AS/3 monitor would be as follows (see SUB `openfiles`, page 138).

```
OPEN "COM1:19200,N,8,1,CS,DS" FOR INPUT AS #1
```

In practice we use the `FREEFILE` command to give an unused filehandle number, and the following code.

```
datexAS3protocol$ = "19200,N,8,1,CS,DS"
datexAS3comportfilenumber% = FREEFILE
OPEN "COM1:" + datexAS3protocol$ FOR INPUT AS #datexAS3comportfilenumber%
```

[1] Nickalls RWD and Ramasubramanian R (1995). *Interfacing the IBM-PC to medical equipment; the art of serial communication* Cambridge University Press, UK.

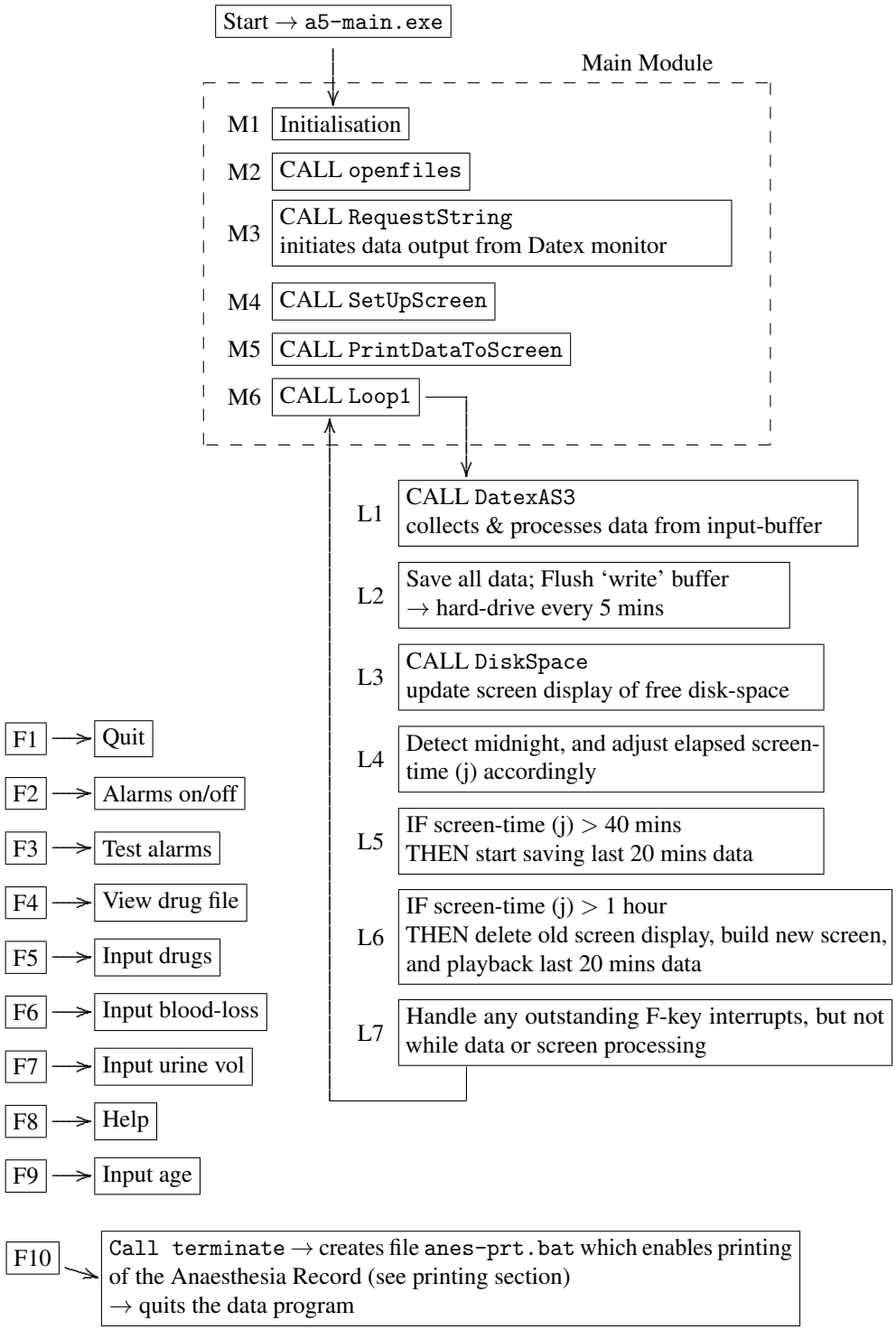


Figure 8.1: Overview of the data program. There are three components (a) the main module M1–M6 (Chapter 8, page 82), (b) the data-collecting loop L1–L7 (Chapter 9, page 90), and (c) F-key actions F1–F10 (Chapter 10, page 96). The program is terminated by either pressing F1 (quit), or pressing F10 (prints out the Anaesthesia Record, and then quits).

Open a data file for the raw data (D-data)

Since this is the first D-data file, then we give it the filename-extension `.D01`, and OPEN the file `...FOR OUTPUT`.

```
Ddatafilename$ = timename$ + ".D01"
Ddatafilenumber% = FREEFILE
OPEN Ddatafilename$ FOR OUTPUT AS #Ddatafilenumber%
```

Open a data file for the gnuplot data (G-data)

Since this is the first G-data file, then we give it the filename-extension `.G01`, and OPEN the file `...FOR OUTPUT`.

```
Gdatafilename$ = timename$ + ".G01"
Gdatafilenumber% = FREEFILE
OPEN Gdatafilename$ FOR OUTPUT AS #Gdatafilenumber%
```

Open a data file for the drug data (`.drg` file)

We use the filename-extension `.drg`, and OPEN the file `... FOR OUTPUT`. Since this file is only opened for writing to, we now write a suitable header to the file and then close it. Note that since this file is to be printed verbatim by LaTeX we need to start the header with `\begin{verbatim}`.

```
drugfile$ = timename$ + ".DRG"
filenumber% = FREEFILE
OPEN drugfile$ FOR OUTPUT AS #filenumber%
PRINT #filenumber%, "\begin{verbatim}"
PRINT #filenumber%, SPACE$(2); "Drug filename = "; drugfile$
PRINT #filenumber%, SPACE$(2); DATE$; SPACE$(2); "Code: "; timename$
PRINT #filenumber%, SPACE$(2); LEFT$(TIME$, 5); SPACE$(2); "START"
PRINT #filenumber%, "  ---------------------------------"
CLOSE #filenumber%
```

Note that the final `\end{verbatim}` LaTeX command is written by the SUB `terminate` (page 175) in response to pressing F10 (to Quit the Data Program).

Open the `PrintAll.bat` file

We use the filename-extension `.bat`, and OPEN the file `... FOR OUTPUT`. Since this is a MS-DOS batch file it does not need a header; we just write the first line and then close it. Each line of `PrintAll.bat` (page 201) is just a command to process a G-data file using the program `plotan3a.exe` (page 217), so we indicate the G-data file we have just opened.

```
PrintAll$ = "PrintAll.bat"
PrintAllfilenumber% = FREEFILE
OPEN PrintAll$ FOR OUTPUT AS #PrintAllfilenumber%
PRINT #PrintAllfilenumber%, "plotan3a.exe" + SPACE$(1) + Gdatafilename$
CLOSE #PrintAllfilenumber%
```

Open the error file

We use the filename-extension `.err`, and OPEN the file `...FOR OUTPUT`. This is a text-file where each line will be an error message. Since this file will be kept open, we just write a header (filename, date, time).

```
errorfile$ = timename$ + ".ERR"
errorfilenumber% = FREEFILE
OPEN errorfile$ FOR OUTPUT AS #errorfilenumber%
PRINT #errorfilenumber%, "ERROR file " + errorfile$
PRINT #errorfilenumber%, TIME$, DATE$
```

8.4 Initiate data output from Datex AS/3 monitor (M3)

Data output from the Datex AS/3 monitor (at 10 second intervals) is initiated by sending (via the serial port) a 52-character string having a particular format. This action is performed by the SUB `RequestString` (page 195), which is part of the Datex module `a5-DXas3.pb` (page 178).

```
CALL requestString
```

Full details of the structure of the requesting string is given in Chapter 3 (page 14).

8.5 Setting up the screen (M4)

The setting up of the screen appearance and graphic windows is performed by the SUB `SetupScreen` (page 173).

```
CALL SetupScreen
```

The screen consists of four large horizontal windows and one small vertical window as shown in Figure 8.2 (page 87). The horizontal windows are for plotting trend data, and are known as the BPwindow (BP, HR, saturation, CVP), CO_2window (etCO_2, inspired CO_2), TVwindow (tidal volume—TV, respiratory rate—RR), and vapourwindow (etVapour, inspired Vapour, age corrected MAC). The small vertical window is known as the 'now' window, and is used for showing some data received during the last 10 seconds (saturation, HR, BP, CVP).

The SUB `SetupScreen` (page 173) sets up the screen using `SCREEN 9` (640 x 350 pixels) and colours defined in the main module; it then writes the text labels for the graphic windows, writes the hour and half-hour times at the top of the screen (above the BP window), writes the age status, shows the alarm status, sets up the separate the graphic windows by calling the five window-specific SUBs, and finally enables the F-key bar (F1—F10) at the bottom of the screen, as follows.

```
SCREEN 9     :REM this is 640 x 350 pixels
COLOR screenforecolour, screenbackcolour
VIEW ,1,15   :REM view with no parameters => whole screen
.....
COLOR screenforecolour, screenbackcolour
....
```

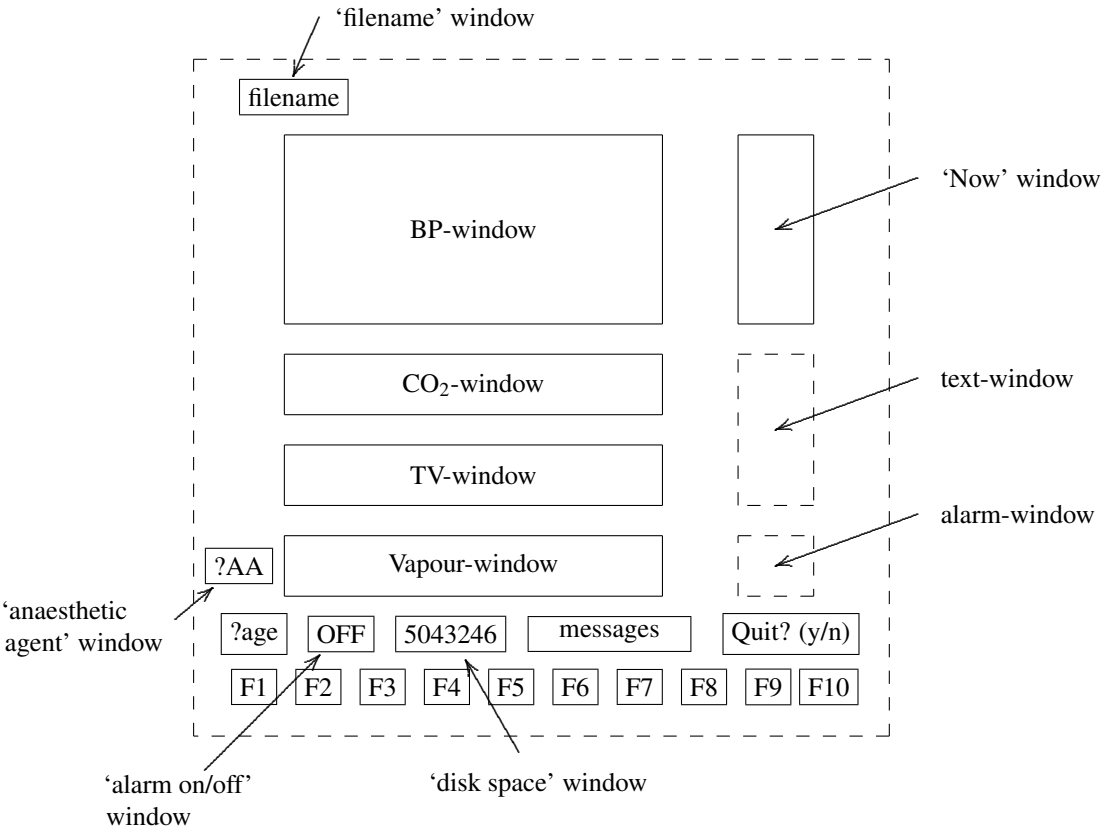


Figure 8.2: Location of screen graphic and text windows.

```
CALL drawTimeMarks  :REM writes the hour & half-hour times
....
CALL drawBPwindow
CALL drawNowWindow
CALL drawCO2window
CALL drawTVwindow
CALL drawVapourwindow
....
KEY ON       :REM the F-key labels
```

The `VIEW ,1,15` command defines the whole screen colour to be blue (1), with border colour intense white (15).

Age status

The age (input via F9 key) is shown in the bottom left of the screen. Initially, the screen writes `?age` in red to emphasise that the age has not been input yet. Once the age has been input then the `?age` is deleted and replaced by the correct age in green, for example `27`.

Alarm status

The alarms are initially switched on by the main module, and so the alarm status is initially shown as `ON` in white, immediately above the F2 key. If the alarm status is toggled to `OFF` (using F2-key) then this is shown in red.

Drawing graphic windows

Drawing a graphic window involves (a) defining the size/location, windowColour, borderColour with a `VIEW...` command, (b) defining the zero, and horizontal and vertical axes scales with the `WINDOW...` command, (c) calculating and drawing the vertical quarter-hour time markers, and (d) drawing the horizontal lines by calling the appropriate horizontal line-drawing SUB , e.g. `CALL drawBpWindowhLines` (page 116). Some lines are drawn solid, and some drawn dashed (in PowerBasic the code `&H707` in the `LINE` command forces a dashed line

The SUB `DrawBPwindow` (page 115) is a typical example, and is shown below. The pixel values defining the horizontal screen space are `bpx1` to `bpx2` (defined in main module; page 102); the pixel values defining the vertical screen space are `bpy1` to `bpy2` (defined in main module); the vertical graphic scale for BP is `deltaBP=200`; the horizontal graphic scale for time is `deltatime=4800` (1 pixel = 10 seconds; 4800 seconds = 1 hour 20 minutes). The vertical timeline colour `bpvline` is green, and the horizontal line colour `bphline` is red respectively (all these values are defined in the main module; page 102).

```
SUB drawbpwindow
REM ----- draw the window -------
VIEW (bpx1, bpy1)-(bpx2, bpy2), windowcolour, windowbordercolour
WINDOW (0, 0)-(deltatime, deltabp)
REM ------draw vertical time lines------
REM calculate when the next quarter-hr marker is (nq secs)
```

```
nq = 15 * 60 - starttime MOD (15 * 60)
REM calculate when the next hr marker is (hq secs)
hq = 60 * 60 - starttime MOD (60 * 60)
REM draw the 1/4 hr markers (900secs = 15 mins)
REM screen width = 1hr 20 mins
FOR g = nq TO 5400 STEP 900
   IF g = hq OR g = hq + 3600 THEN
        REM use solid lines for hour markers
        LINE (g - 5, 0)-(g - 5, deltabp), bpvline
        LINE (g, 0)-(g, deltabp), bpvline
        LINE (g + 5, 0)-(g + 5, deltabp), bpvline
    ELSE
       REM use a dashed line for quarter markers
       REM the &H707 code forces a dashed line
       LINE (g - 5, 0)-(g - 5, deltabp), bpvline, , &H707
       LINE (g, 0)-(g, deltabp), bpvline, , &H707
       LINE (g + 5, 0)-(g + 5, deltabp), bpvline, , &H707
   END IF
NEXT g
REM ----draw horizontal lines----
CALL drawbpwindowhlines
END SUB
```

8.6 Print data to the screen text-window area (M5)

Underneath the vertical 'Now' window is a small area of screen where certain parameter values are printed (see Figure 8.2, partly as additional information, but mainly as these are parameters whose values sometimes fall outside the range of the relevant horizontal graphic trend window.

```
CALL PrintDataToScreen
```

The SUB `PrintDataToScreen` (page 158) is called initially by the main module, and later by the SUB `LoopOne` (page 133) to update the values after each data input from the Datex monitor. The four values printed by this SUB to this area are CVP, end-tidal CO2, respiratory rate (rr), tidal volume (TV).

8.7 `CALL LoopOne` (M6)

This is detailed in Chapter 9, page 90.

Chapter 9

The loop

ch-dploo.tex

9.1 Introduction

The SUB `LoopOne` (page 133) is the main working loop which collects all the data and coordinates all the necessary housekeeping—see components L1–L7 in Figure 9.1, page 92. The program stays in this loop for the whole anaesthetic, until the Data Program is quit (press F1), or printing the data record is initiated (press F10).

The SUB `LoopOne` calls two 'housekeeping' SUBS, namely SUB `LoopTwo` (page 135), and SUB `HouseKeeping` (page 120). The SUB `LoopTwo` is really an extension of `LoopOne`, and is kept as a separate SUB purely for convenience. The SUB `HouseKeeping`, which is only called every 10 seconds, coordinates saving the last 20 minutes of screen data (for playing back to the next screen) and triggering the building of a new screen at the end of each hour. These three SUBs together (which could perhaps really be better all incorporated into a single loop) coordinate seven key functional activities (L1–L7) as follows.

9.2 Get Datex data (L1)

All the Datex data-processing subroutines and functions are bundled together as a separate interface module `a5-dxas3.pb` (Chapter 13, page 178). Note that regular data-output (every 10 seconds) from the Datex AS/3 monitor is initiated by the main module (Chapter 8, page 82) by calling the SUB `RequestString` (page 195)—see M3 in Figure 9.1.

Each round of collecting Datex data from the computer's input buffer, checking the format, decoding the input data-string, saving the data, plotting the data to the screen display is coordinated by the SUB `datexas3` (page 186). This SUB is CALLed by SUB `loopone` (page 133) if more than 5 characters (bytes) are in the input-buffer, as follows.

```
IF LOC(datexAS3comportfilenumber%) > 5 OR LEN(DatexAS3Buffer$) > 5
                                                    THEN CALL DatexAS3
```

Note that each data-string output by the Datex AS/3 monitor is 321 bytes—see Chapter 3.

9.3 Saving G-data (L2)

The so-called G-data[1] is sampled and saved every 40 seconds by the SUB `SaveGNUdata` (page 163), as follows.

```
IF TIMER > (oldGNUtime& + 40) THEN CALL saveGNUdata
```

The PowerBasic numeric variable `oldGNUtime&` (4-byte 'long' integer)[2] holds the time in seconds since midnight, and is reset at the beginning of the SUB `saveGNUdata` (page 163).

9.4 Determine free space on the hard-drive (L3)

The free space is printed periodically (about every 300 seconds) to the screen in the 'disk-space' window (see Figure 8.2, page 87), by the SUB `looptwo` (page 135). The free space value is printed in white, unless the free space is less than 4 MBytes in which case it is printed in red (to alert the anaesthetist), as follows (see SUB `looptwo`, page 135).

```
...
IF TIMER > LastSaveDataTime& + 300 THEN
   ...
   CALL diskspace(freebytes#)
   REM first delete existing value
   LOCATE 24, 18: PRINT SPACE$(14)
   IF freebytes#>4000000 THEN
         COLOR white, screenbackcolour
         LOCATE 24, 18: PRINT "DISK"; freebytes#
         COLOR screenforecolour, screenbackcolour
   ELSE
         COLOR red, screenbackcolour
         LOCATE 24, 18: PRINT "DISK"; freebytes#
         COLOR screenforecolour, screenbackcolour
   END IF
   ...
END IF
```

Determining the free disk space on the hard-drive is performed by the SUB `diskspace` (page 115), as follows. Note that the integer variable `freespace#` is an 8-byte 'double precision' variable.

```
...
REM PowerBASIC 3.5
REM uses INTERRUPT 21h (=33); function 36h (54) in HIGH byte of AX (ie AH)
REM uses disk DRIVE code in LOW byte of DX (ie in DL)
REM 0 = default drive;  1 = A drive;  2 = B drive;  3 = C drive....etc
    AX%=1:BX%=2:CX%=3:DX%=4
```

[1] GNUdata—a subset of data in a format for processing with GNUplot used for printing the anaesthetic Record.

[2] See the PowerBasic 'user guide' Chapter 5 for details of data variables.

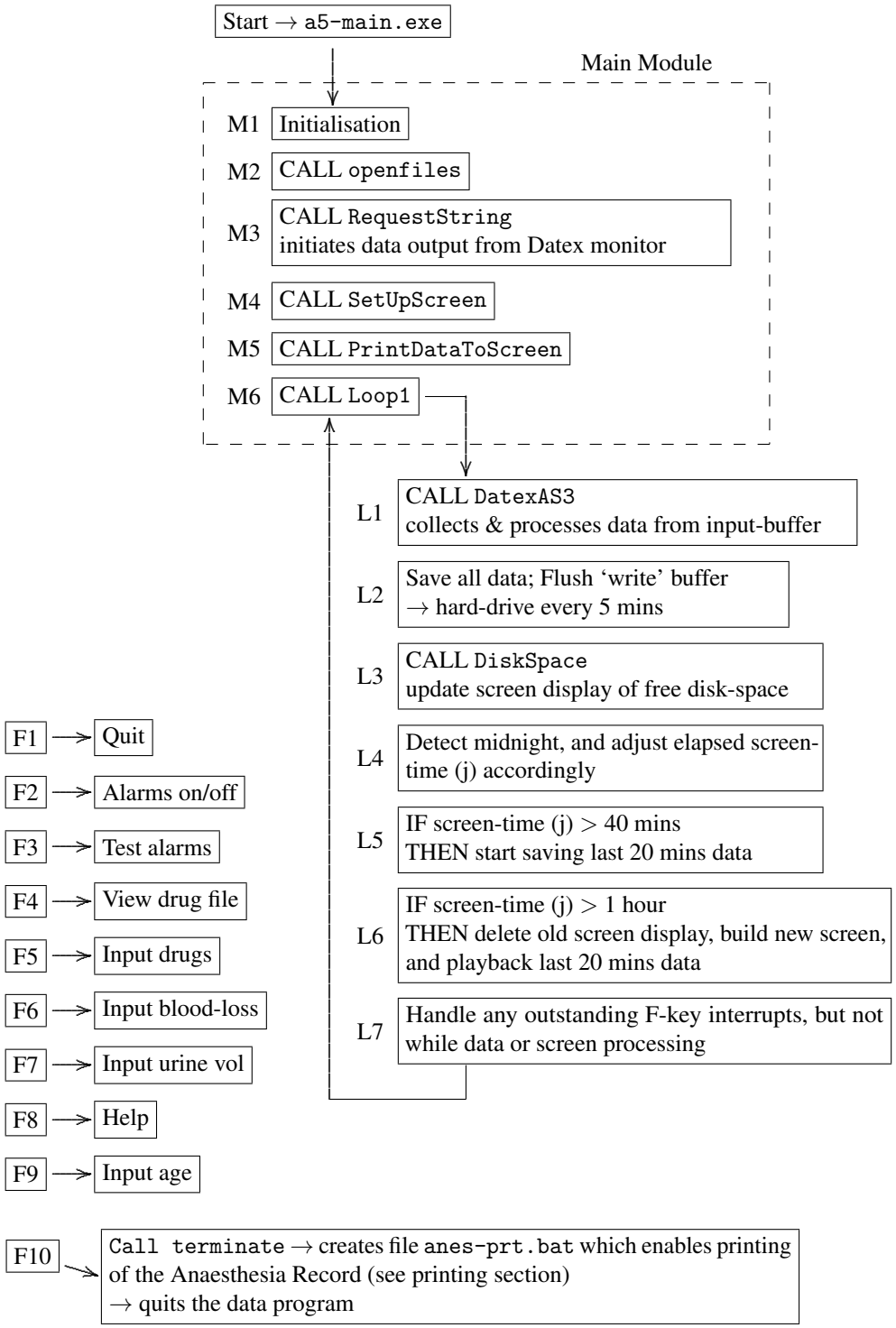


Figure 9.1: Overview of the data program. There are three components (a) the main module M1–M6 (Chapter 8, page 82), (b) the data-collecting loop L1–L7 (Chapter 9, page 90), and (c) F-key actions F1–F10 (Chapter 10, page 96). The program is terminated by either pressing F1 (quit), or pressing F10 (prints out the Anaesthesia Record, and then quits).

```
    REG AX%, &h3600           :REM AX
    REG DX%, 3                :REM DX
    CALL INTERRUPT &h21       :REM =33
    AXout%=REG(AX%)
    BXout%=REG(BX%)
    CXout%=REG(CX%)
freespace#=CXout%*AXout%
freespace#=freespace#*BXout%
bytes#=freespace#
END SUB
```

Note that this PowerBasic routine will 'crash' the computer if there is no disk in the drive, so need to first check for the presence of a disk (obviously not a problem with the hard-drive).

9.5 Detect midnight and adjust elapsed time (L4)

The screen-time (since the start a given screen as used for the screen windows), is the integer `j` where j $\leq$ 4800 seconds (i.e. a maximum of 80 minutes). Note that `TIMER` returns the time in seconds since midnight, and `starttime` is the time since the start of the Data Program.

If the PC clock time `TIMER` is less than 60, then the FLAG (a string) `midnight$` is SET. This is in SUB `looptwo` (page 135) as follows.

```
...
IF TIMER > LastSaveDataTime& + 300 THEN
   ...
   REM in looptwo
   IF midnight$ = "" AND TIMER < 60 THEN midnight$ = "SET"
   REM (see LOOPone for midnight timing and j)
   ...
END IF
```

Once the `midnight$` FLAG is SET, then this is detected during the next run through the SUB `loopone` (page 133), and triggers the addition of 86400 seconds to the value of the internal clock time `INT(TIMER)`, as follows.

```
REM in loopOne
REM get the j value and check for midnight
REM midnight Flag set in housekeeping
IF midnight$ = "SET" THEN
         j = (INT(TIMER + 86400) - starttime)
ELSE
         j = (INT(TIMER) - starttime)
END IF
```

The clock time `TIME$` is also written to the screen in `looptwo`.

```
COLOR green, screenbackcolour
LOCATE 13, 71: PRINT TIME$
COLOR screenforecolour, screenbackcolour
```

9.6 Save the last 20 minutes of data to a file (L5)

At the beginning of each new screen (except the first screen) the last 20 minutes of the previous screen data (saved to a file) is played back before starting to show new data. We keep track of which screen we are currently dealing with by holding the 'screen number' (1, 2, 3...) in the integer variable `currenthour%` (this value is set to 1 in the main module (page 102) at start-up).

Exactly how we trigger the saving of the last 20 minutes of data therefore depends on whether we are dealing with the first screen (i.e. `currenthour%` $= 1$) or subsequent screens, since we only run the first screen for 60 minutes, but we run all subsequent screens for $60 + 20 = 80$ minutes (= 4800 seconds). We therefore trigger saving the last 20 minutes data after either 2500 seconds (first hour), or 3600 seconds (subsequent hours), by doing two things:

- Set a FLAG (i.e. `windowtime$ = "lasthour"`)
- Open a file (`lasthour.dat`) to hold the data

This is done in the SUB `housekeeping` (page 120) as follows.

```
...
REM trigger the opening of the lasthourdatafile
IF currenthour% = 1 THEN
   IF j > (3600 - 1100) AND j <= 3600 AND windowtime$ = "" THEN
         windowtime$ = "lasthour"
         LastHourDataFilename$ = "lasthour.dat"
         LastHourDataFilenumber% = FREEFILE
         OPEN LastHourDataFilename$ FOR OUTPUT AS #LastHourDataFilenumber%
         lasthourstarttime = (INT(TIMER) - starttime): REM this =j
   END IF
ELSE
   REM ie  for hours 2 and greater
   IF j > (4700 - 1100) AND j <= 4700 AND windowtime$ = "" THEN
         windowtime$ = "lasthour"
         LastHourDataFilename$ = "lasthour.dat"
         LastHourDataFilenumber% = FREEFILE
         OPEN LastHourDataFilename$ FOR OUTPUT AS #LastHourDataFilenumber%
         lasthourstarttime = (INT(TIMER) - starttime): REM this =j
   END IF
END IF
...
```

Once the FLAG is set (`windowtime$ = "lasthour"`), then with each subsequent pass through the SUB `looptwo` we save the screen data to the file by CALLing the SUB `savelasthourdata` (page 165), as follows (note that the SUB `housekeeping` is called approximately every 10 seconds by SUB `loopone`).

```
...
IF  windowtime$ = "lasthour" THEN CALL savelasthourdata
...
```

Note that for optimum results, we require a way of (a) writing this data to the file, and (b), reading it back and printing it to the screen FAST, in order to allow us to replay the last 20 mins FAST.

9.7 Delete old screen and build new screen (L6)

The build of new screens is triggered by code at the end of the SUB `housekeeping` (page 120), which CALLS the SUB `PrintNewScreen` (page 160) as follows.

```
...
REM trigger the making of a new screen
IF currenthour% = 1 THEN
      REM for 1st hour trigger new window when J exceeds 3600 secs
      IF j > 3600 THEN
           REM print  a new screen
           lasthourendtime = (INT(TIMER) - starttime)
           CALL PrintNewScreen
           REM note the last20mins data-replay is called by SUB PrintNewScreen
     END IF
ELSE
     REM ie for 2nd and subsequent hours
     REM trigger new window when J exceeds 4700 (ie just *before* end of main window)
     IF j > 4700 THEN
           REM we have about 100 sec leeway before hitting the end of the screen
           REM data-replay is  called from SUB  PrintNewScreen
           lasthourendtime = (INT(TIMER) - starttime)
           CALL PrintNewScreen
     END IF
END IF
...
```

Note the screen rebuild and replay of the last 20 minutes data is coordinated by the SUB `PrintNewScreen` (page 160). During this process, keyboard input is temporarily disabled until the system is ready to process new data to the screen.

9.8 Process outstanding F-key interrupts (L7)

The F-keys are handled in the loop subroutines only in certain places in order to avoid processing during critical moments. They are not handled as true interrups, but in a simplified way by setting global string FLAGS (and sometimes integer FLAGS), which are checked for, and then acted upon as necessary.

For example, the following code in SUB `loopone` (page 133) checks the status of the FLAG `screenmode$` which is set when one of the F-keys is pressed (see Chapter 10, page 96). If the `screenmode$` FLAG is SET then it CALLs the SUB `keyhandler` (page 122).

```
IF screenmode$ = "OFF" THEN CALL KeyHandler
```

Chapter 10

F-keys

ch-dpfk.tex

10.1 Introduction

See Figure 10.2 (page 101). Pressing an F-key usually sets one or more string FLAGs, and then CALLs either the SUB `terminate`, page 175 (F1, F10), or the SUB `KeyHandler`, page 122 (all other F-keys). For example, pressing the F5-key results in setting three FLAGS (`drugMode$, screenMode$, keyMode$`, and then CALLs the SUB `KeyHandler` as follows (actioned in `a5-main.pb` Chapter 11, page 102).

```
REM main module
...
KEY 5, "DRUGS"  :REM this writes the label "DRUGS" above F5 on screen
ON KEY(5) GOSUB key5line
KEY(5) ON
...
key5line:
  REM allows inputting of drugs
  DO: LOOP UNTIL LEN(INKEY$) = 0
  drugmode$="ON" :REM enables printing for drug list (in terminate SUB)
  REM note that drugmode$ is also set with F6(blood) and F7(urine)
  screenmode$ = "OFF"  :REM disables building new screen
  keymode$ = "FIVE"    :REM  used by SUB KeyHandler
  CALL KeyHandler
RETURN
...
```

The FLAG `keyMode$` holds the pointer to the appropriate section of the SUB `KeyHandler` (page 122). Consequently if a particular F-key requires special screen clearing of the bottom half of the screen (e.g. to allow data input from the keyboard) then this is coordinated by the SUB `KeyScreenHandler` (page 123), as follows: Note the SUB `KeyHandler` (page 122) first saves the whole screen, then disables the F-keys (to stop further F-keys being actioned), CALLs the SUB `KeyScreenHandler` (page 123), and then restores the screen, and enables the F-keys.

```
SUB keyHandler
...
REM get current time
KeyClock = TIMER
REM save the whole screen
CALL screen9SAVE
REM disable all the F-keys
FOR keynum = 1 TO 10
   KEY(keynum) OFF
NEXT
REM --------
REM now process the F-key
IF keymode$<>"" THEN CALL KeyScreenHandler(keymode$)
REM --------
REM on return replace stored whole screen
CALL screen9RESTORE
REM -----------
REM reset the screenMode$ and KeyMode$ flags
screenmode$="ON"  : Keymode$=""
REM-------enable the F-keys ---------
FOR keynum = 1 TO 10
   KEY(keynum) ON
NEXT
REM stop the clock (timeaway is time spent processing the F-key)
timeaway = TIMER - KeyClock
...
```

Note that only F-keys F4 → F9 require handling via the SUB `KeyHandler` (page 122). Keys F5, F6, F7, F9 require the SUB `KeyScreenHandler` (page 123) to clear the bottom half of the screen to allow inputting of data etc. Keys F4 and F8, also require the scrolling of a file in the bottom half of the screen (F4 → drug file; F8 → help file).

10.2 F-keys

The F-Key actions are determined towards the end of the main module (Chapter 11, page 102). They are summarised as follows.

- F1 Quit without printing

  CALL Terminate (page 175)

- F2 Toggles alarm status ON/OFF. The ON/OFF status is shown on the screen in the on/off window (see Figure 10.1, page 100).

  Sets FLAG `alarmMode$` → ON

  This string is used to facilitate the toggeling and also to sound an audible alarm by SUB `Alarms` (page 112), made by SUB `alarmSound` (page 114), as follows.

  ```
  REM main module
  ...
  key2line:
  ```

```
DO: LOOP UNTIL LEN(INKEY$) = 0
REM (alarms toggle) set flag to be picked up in the main loop
REM AlarmMode$ is used to enable beep alarm (see alarm SUB)
IF alarmmode$ = "ON" THEN
   alarmmode$ = "OFF"
   LOCATE 24, 11: PRINT SPACE$(3)
   COLOR red, screenbackcolour
   LOCATE 24, 11: PRINT "OFF"
ELSE
   alarmmode$ = "ON"
   LOCATE 24, 11: PRINT SPACE$(3)
   COLOR white, screenbackcolour
   LOCATE 24, 11: PRINT "ON"
END IF
COLOR screenforecolour, screenbackcolour
RETURN

SUB alarms
...
IF code = 1 AND alarmmode$ = "ON" THEN
   REM alarmmode$ is toggled on/off by F2key
   CALL alarmsound
   subname$ = "alrm-c"
   code = 0
END IF
...
```

Note here the variable `code` is set in the SUB `alarms` (page 112) if it detects an 'alarm' state

- F3 Tests the visible on-screen alarm codes (they show for 3 secs approximately). The testing is coordinated by the SUB `AlarmTester` (page 114).

  Sets FLAG `alarmtest$` → SET

```
SUB LoopTwo
...
IF alarmtest$ = "SET" THEN CALL alarmtester
...
```

- F4 Scrolls the drug file in a screen window.

  Sets FLAG `screenMode$` → OFF which prevents the program from renewing the main screen (ie a the end of each hour) if the scroll window is open.

  Sets FLAG `ScreenMode$` → OFF

  Sets FLAG `keyMode$` → FOUR

  CALL KeyHandler (page 122)

- F5 Opens the drug-input window (allows drugs/comments etc to be written to the `drugfile.drg` file.

  Sets FLAG `drugMode$` → ON which prevents the program from renewing the main screen (ie at the end of each hour) if the scroll window is open.

Sets FLAG `ScreenMode$` → OFF

Sets FLAG `keyMode$` → FIVE

CALL KeyHandler (page 122)

- F6 Opens the blood-loss window (allows blood-loss volume to be written to the `drugfile.drg` file

  Sets FLAG `drugMode$` → ON which prevents the program from renewing the main screen (ie at the end of each hour) if the scroll window is open.

  Sets FLAG `ScreenMode$` → OFF

  Sets FLAG `keyMode$` → SIX

  CALL KeyHandler (page 122)

- F7 Opens the Urine volume (allows urine volume to be written to the `drugfile.drg` file

  Sets FLAG `drugMode$` → ON which prevents the program from renewing the main screen (ie at the end of each hour) if the scroll window is open.

  Sets FLAG `ScreenMode$` → OFF

  Sets FLAG `keyMode$` → SEVEN

  CALL KeyHandler (page 122)

- F8 Opens the HELP window (scrolls the `helpfile$` file) [note `helpfile$` = help.dat]

  Sets FLAG `ScreenMode$` → OFF

  Sets FLAG `keyMode$` → EIGHT

  CALL KeyHandler (page 122)

- F9 Opens the AGE window (allows you to input the AGE of the patient; after setting the age, the program deletes the ”?age” message on the screen (in RED), and replaces it with the input AGE value (in GREEN)).

  Sets FLAG `ScreenMode$` → OFF

  Sets FLAG `keyMode$` → NINE

  CALL KeyHandler (page 122)

- F10 Initiates printing of the anaesthetic record.

  Sets FLAG `PrintRecord$` → SET

  CALL Terminate (page 175)

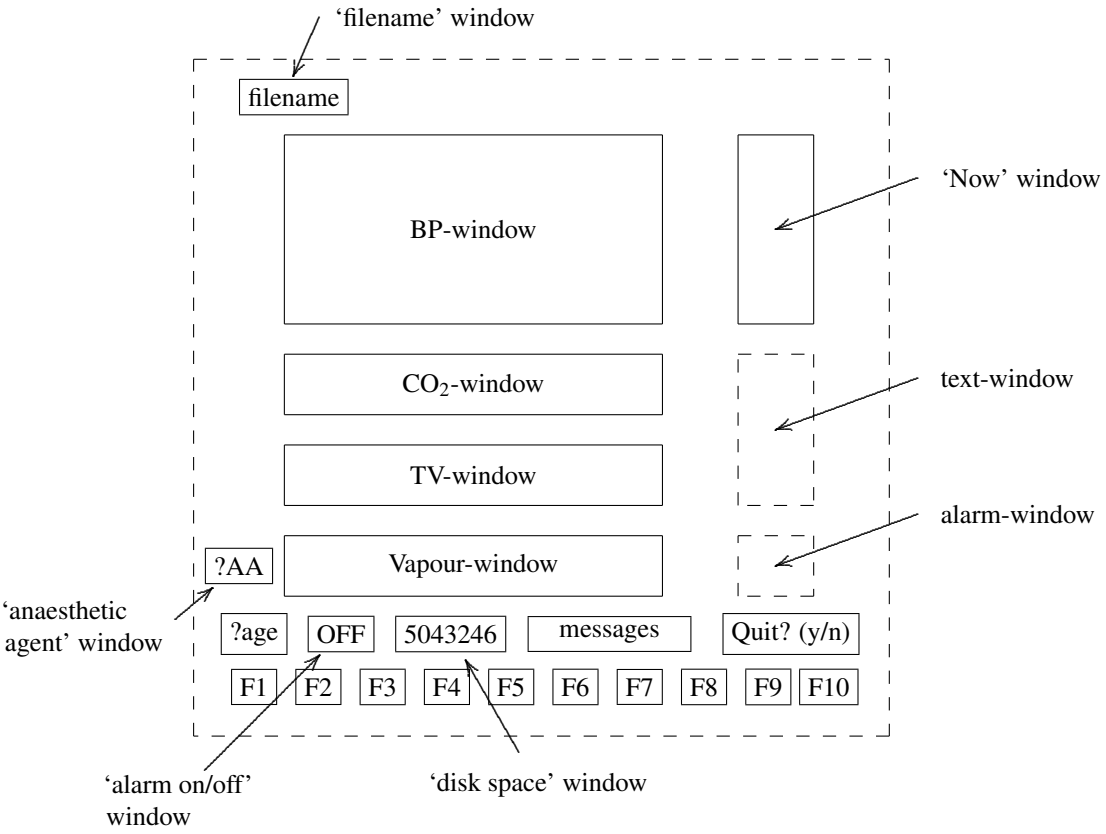


Figure 10.1: Location of screen graphic and text windows.

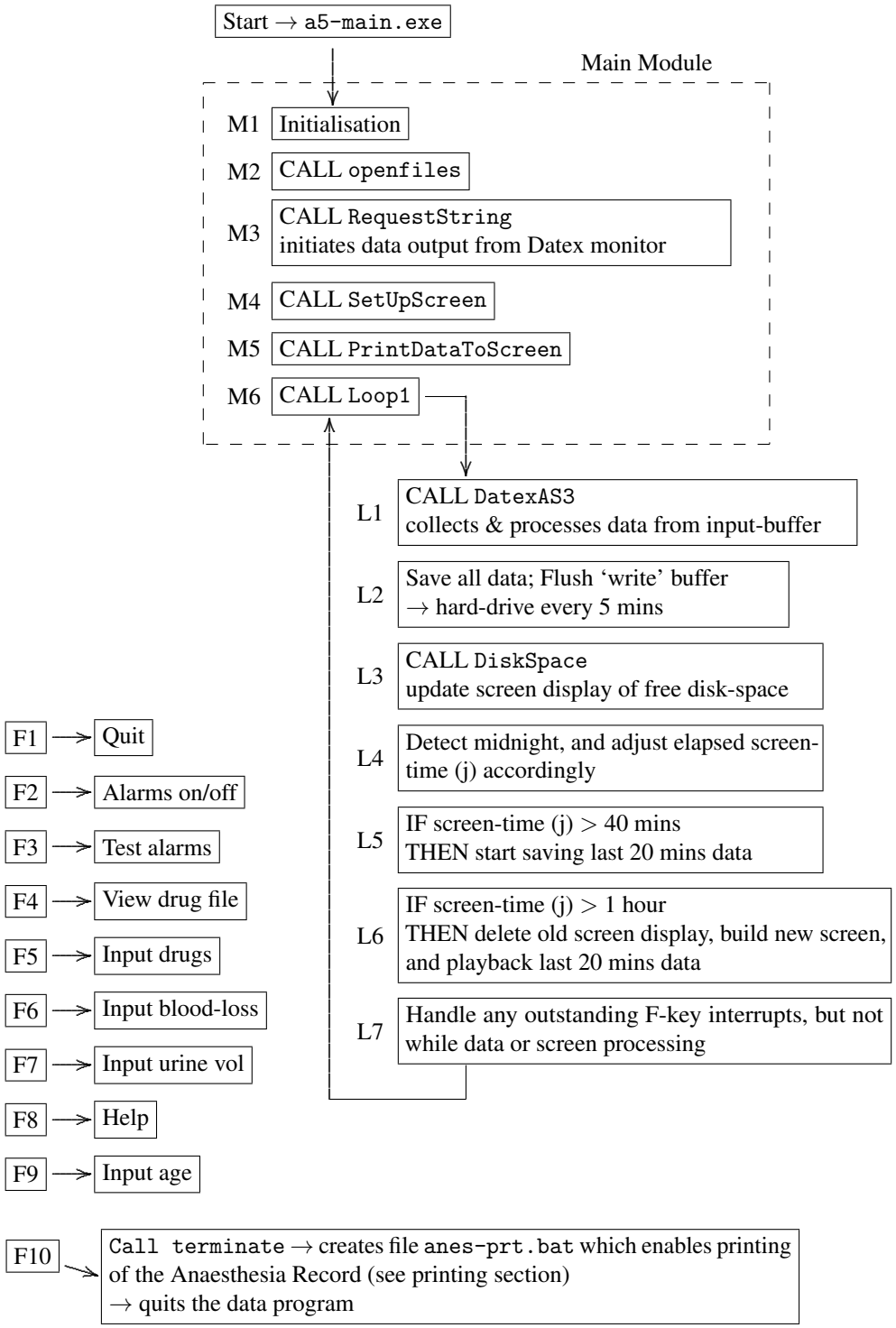


Figure 10.2: Overview of the data program. There are three components (a) the main module M1–M6 (Chapter 8, page 82), (b) the data-collecting loop L1–L7 (Chapter 9, page 90), and (c) F-key actions F1–F10 (Chapter 10, page 96). The program is terminated by either pressing F1 (quit), or pressing F10 (prints out the Anaesthesia Record, and then quits).

Chapter 11

Main module—a5-main.pb

ch-main.tex

```
REM ------TOP line--------------
REM a5-main.pb
REM Datex AS/3 monitor ONLY via COM1
REM PowerBASIC 3.5 DOS version
REM
REM Copyright RWD Nickalls November 21, 1999
REM Department of Anaesthesia, City Hospital, Nottingham, UK
REM EMAIL: dicknickalls@compuserve.com
REM
REM --------NOTES----------------
REM   adjustments for theatre (ie PC simulator --> theatre use)
REM   change N --> E in the serial protocol (in SUB openfiles)
REM   --> re compile with >pbc /ce a5-anes.bas
REM
REM -------List of all the INCLUDE files--- put after SHARED statements)-----
REM     a5-draw.bas    (contains all the screen drawing/writing routines)
REM     a5-lib.bas     (the library = diskspace/screenBON/ScreenBOFF/..)
REM     a5-keyf.bas    (all the KEY subs for /drugs/blood/urine/scroll)
REM     a5-plot.bas    (all the screen plot routines)
REM     a5-DXas3       (Datex AS/3 routines)
REM -------------------------
REM file for REPLAY fast data (last 20 mins) = lasthour.dat
REM use Screen9Save  and Screen9Restore  commands
REM -------------------------------------------
REM screensavers called by keyhandler sub
    DIM STATIC  ScreenTopBuf%(28001)
    DIM STATIC  ScreenBotBuf%(28001)
    SHARED ScreenTopBuf%(), ScreenBotBuf%()
REM ----- variables used in a4-SPLAB.bas module----------
SHARED  datatime$, nibpNowTime$, nibpOldTime$
SHARED ecgcol%, artcol%, cvpcol%, spo2col%, nibpcol%, etco2col%
SHARED ecgstartcol%, ecgendcol%, ecglen%, artstartcol%, artendcol%
```

```
SHARED artlen%, cvpstartcol%
SHARED h1line$, h2line$, h3line$, h4line$, h5line$
REM ------- screen variables -------
SHARED bpx1, bpx2, bpy1, bpy2
SHARED bpnowx1, bpnowx2, bpnowy1, bpnowy2, bpnowwidth
SHARED co2x1, co2x2, co2y1, co2y2
SHARED vapourx1, vapourx2, vapoury1, vapoury2, deltavapour
SHARED tidalvolx1, tidalvolx2, tidalvoly1, tidalvoly2, deltatidalvol
SHARED deltatime, deltabp, deltaco2
REM -----LONG  integer variables------
REM make times and j all LONG integers   (as maybe greater than 32,000)
DIM  starttime AS SHARED LONG    :REM (4 bytes =32 bits = &)
DIM j AS SHARED LONG
SHARED  firstj%: REM firstj% is first j-value in lasthour.dat file
SHARED LastSaveDataTime&  :REM time of saving data to disk (looptwo)
SHARED lastHourStartTime, LastHourEndTime
SHARED oldGNUtime&, oldHKtime&
REM -----time params = floating point -----
SHARED Keyclock, timeaway, screentime :REM time away from trend screen - see KeyF subs
REM ----- colours---------------------------------
SHARED fio2colour, satcolour, bpcolour, nibpcolour, hrcolour
SHARED fico2colour, etco2colour
SHARED black, darkgrey, blue, lightblue, lightgreen
SHARED cyan, lightcyan, lightred, magenta, lightmagenta
SHARED brown, white, brightwhite
SHARED windowcolour, windowbordercolour
SHARED bpvline, bphline
SHARED green, red, yellow, screenbackcolour, screenforecolour
REM ------FLAGS---------
REM generally used to allow a new activity *only* when last
REM activity has finished
SHARED pl20mf$      :REM = "on"   Print last 20 mins fast
SHARED windowtime$  :REM  = "lasthour"  enables saving data to last hour
                     REM see SUB housekeeping
SHARED filemode$    :REM  = "file2open"  used in PrintNewScreen SUB
SHARED lasthourduration, lasthourendtime, lasthourstarttime
SHARED Quit$        :REM used in F1 to quit without printing
SHARED PrintRecord$ :REM used in F10 to quit *and* print
SHARED screenmode$ : REM (screenmode OFF (main scr is off etc or "ON"))
                     REM ? used to prevent doing things when screen is OFF etc
SHARED keymode$    : REM used by FKEYS to call KeyScreenHandler = one, two, three etc)
SHARED  alarmmode$  :REM this string is set (enabled) by SUB setupscreen
SHARED alarmtest$ :rem flag for calling the alarmtester SUB
SHARED  GNUnibp$  :REM enables saving the nibp value to G file
SHARED nibp$  : REM  used to control plotting NIBP (in datex module)
SHARED alarmNIBP$  : REM
REM ----------KEY variables--------
SHARED druginfo$, drugmode$     :REM used in SUB SaveDrugData
REM --------------------
SHARED currenthour%, timename$
```

```
REM ------- variables used with the modules-------
SHARED hrecg, hrnibp, hroxim, nibps, nibpm, nibpd
SHARED oldhrecg,oldhrnibp,oldhroxim,oldnibps,oldnibpm,oldnibpd
SHARED abp1s, abp1d, abp1m, cvp2s, cvp2d, cvp2m, cvp
SHARED oldabp1s, oldabp1m,oldabp1d, oldcvp
SHARED co2, etco2, fico2, fio2, fin2o, etn2o
SHARED oldetco2
SHARED oximsat, oldoximsat  ,sat, satDatex
SHARED oldfio2,oldsat
SHARED olds, oldd,oldhrg,oldhrox   :REM check in PLOT why these are needed too
SHARED sh, bmac
SHARED vapourin, vapourout,vapourcodemode$, vapourcode$
SHARED tidalvol, tvexp, tvinsp, rr, code$
SHARED  oldTVexp, oldrr
SHARED subname$
SHARED filetoscroll$, r1&, r2&
SHARED age%
REM -------------------------
REM NB datafile$ must NOT be shared (this is a datastring)
SHARED  timename$
REM ----------------------
REM now all the files and associated filenumbers
SHARED datexAS3comportfilenumber%, datexAS3buffer$
SHARED Ddatafilename$, Ddatafilenumber%
SHARED Gdatafilename$, Gdatafilenumber%
SHARED PrintAll$, PrintAllfilenumber%
SHARED drugfile$
SHARED LastHourDataFilename$, LastHourDataFilenumber%
SHARED errorfile$, errorfilenumber%
SHARED messageTime, messagetimeFlag% : REM set in message SUB
REM ----set the max COM PORT buffer size------------
 $COM 30000
REM -----------------------------------------------
SHARED version$
version$ = "5aPB": REM version$ is printed in SUB setupscreen
REM ------ define screen colours-----------------
black = 0: grey = 8: blue = 1: lightblue = 9: green = 2: lightgreen = 10
cyan = 3: lightcyan = 11: red = 4: lightred = 12: magenta = 5: lightmagenta = 13
brown = 6: yellow = 14: white = 7: brightwhite = 15
REM--------define screen colours-----------
     satcolour = red
     fio2colour = red
     bpcolour = green
     nibpcolour = green
     hrcolour = yellow
     fico2colour = red
     etco2colour = blue
     screenbackcolour = blue
     screenforecolour = white
     windowcolour = white
```

```
     windowbordercolour = brightwhite
     bphline = red
     bpvline = green: REM = time lines
     fio2line = blue
REM -------screen details------------------
REM optimum size = 1 pixel /per 10 secs (C04 string comes every 10 secs.)
REM hh= no of hrs in the BP window
hh = 1.3333: REM 1.33
REM define the size/position of the BP window
bpwindowheight = 140: REM pixels
bpx1 = 50: REM no of pixels from LHS
bpx2 = bpx1 + hh * 360: REM this gives 1.33 hrs at 1 pixel/10 secs
bpy1 = 20: REM = top of window in pixels (measured DOWN from top)
bpy2 = bpy1 + bpwindowheight
REM
REM define the separation of windows
hwindowsep = 45: REM in pixels
vwindowsep = 10: REM in pixels
REM
REM define the size/position of the BPNOW window
bpnowwidth = 50: REM in pixels
bpnowx1 = bpx2 + hwindowsep
bpnowx2 = bpnowx1 + bpnowwidth
bpnowy1 = bpy1: REM same as for BP window
bpnowy2 = bpy2
REM
REM define the size/position of the CO2 window
co2windowheight = 42: REM pixels  was 50
co2x1 = bpx1
co2x2 = bpx2: REM keeps the width the same as for the BP window
co2y1 = bpy2 + vwindowsep: REM = top of window in pixels (measured DOWN from top)
co2y2 = co2y1 + co2windowheight: REM = bottom of window
REM
REM define the size of the TIDALVOL window
tidalvolwindowheight = 42
tidalvolx1 = bpx1
tidalvolx2 = bpx2
tidalvoly1 = co2y2 + vwindowsep: REM top of window in pixels
tidalvoly2 = tidalvoly1 + tidalvolwindowheight: REM bottom of tidalvol window
REM
REM define the size/position of the VAPOUR window
vapourwindowheight = 42: REM was 50
vapourx1 = bpx1
vapourx2 = bpx2: REM keeps the width the same as for the BP window
vapoury1 = tidalvoly2 + vwindowsep: REM = top of window in pixels (DOWN from top)
vapoury2 = vapoury1 + vapourwindowheight: REM = bottom of vapour window
REM ============================
REM define the time range in SECS (ie = hhx60x60) hh = no of hrs
    REM deltatime = hh * 60 * 60: REM hh= 1.33 = 1h 20 mins = 4800 secs
    deltatime = 4800
```

```
REM define the BP range  (mm Hg)
    deltabp = 200
REM define the co2 range (kpa)
    deltaco2 = 8: REM from 0 to 8 KPa
REM define the vapour range (%)
    REM this may need moving if going to consider different ranges etc
    deltavapour = 4: REM from 0 to 4%
REM define the Tidal vol range
    deltatidalvol = 1000: REM from 0 to 1000 mls

REM ======= start of KEY commands ==========
KEY 1, "QUIT"
ON KEY(1) GOSUB key1line
KEY(1) ON

KEY 2, "ALARMS"
ON KEY(2) GOSUB key2line
KEY(2) ON

KEY 3, "TESTal"
ON KEY(3) GOSUB key3line
KEY(3) ON

KEY 4, "VIEW": REM use this to view/scroll the drugfile.dat
ON KEY(4) GOSUB key4line
KEY(4) ON

KEY 5, "DRUGS"
ON KEY(5) GOSUB key5line
KEY(5) ON

KEY 6, "BLOOD"
ON KEY(6) GOSUB key6line
KEY(6) ON

KEY 7, "URINE"
ON KEY(7) GOSUB key7line
KEY(7) ON

KEY 8, "HELP"
ON KEY(8) GOSUB key8line
KEY(8) ON

KEY 9, "AGE"
ON KEY(9) GOSUB key9line
KEY(9) ON

KEY 10, "PRINT"
ON KEY(10) GOSUB key10line
KEY(10) ON
```

```
REM ======== initialise mode strings/parameters ====================
REM to see list of strings and what they do see INFO sub
filemode$ = "file2open": REM ? used in lasthour and newscreen
windowtime$ = "": REM ie do NOT save lasthourdata yet (housekeeping)
currenthour% = 1: REM =1  determines G01 to G01 etc and timing of lasthour data saving
vapourcodemode$ = "notselected": REM used to print ?AA on screen by SUB setupscreen
gnucounter% = 0: REM no of lines in Gdatafile
alarmmode$ = "OFF": REM have alarm sound OFF
alarmtest$ = "": REM ? need to test alarms at the beginning too
age% = 40: REM is the default age for MAC
nibp$="old"  :REM nibp$ is set in the decode sub of Datexas3
REM
REM =========== now start the program====================

REM first get the start time
starttime = INT(TIMER): REM this is a LONG variable

ON ERROR GOTO errortrap
COLOR 15,1
CLS

CALL openfiles     : REM open the ports and files
PRINT
PRINT " serial port open OK"
PRINT
SLEEP 1

REM now trigger data output (every 10 sec) from Datex AS/3 monitor
CALL RequestString :REM in DatexAS3 module (A5-DXas3.bas)
REM  start timer and wait max 5 sec for data to arrive
thistime=timer
DO
  IF TIMER > thistime + 5 then
     PRINT
     BEEP
     PRINT " No data --- quiting program"
     SLEEP 1
     END
  END IF
  REM if data in buffer, then continue
  IF LOC(datexAS3comportfilenumber%) > 0 then
     PRINT " data output OK"
     SLEEP 1
     EXIT DO
  END IF
  SLEEP 1
  REM print dots ... while waiting
  PRINT ".";
LOOP
```

```
CALL setupscreen      : REM make the screen
CALL PrintDataToScreen: REM print text info to the screen

REM now start the loop
CALL LoopOne

REM ------------ errortrap routine ----------------
errortrap:
BEEP
REM the error file is #errorfilenumber% (timename$.err)
    PRINT #errorfilenumber%, "--------------------"
    PRINT #errorfilenumber%, "ErrorTrap event"
    PRINT #errorfilenumber%, DATE$, TIME$
    PRINT #errorfilenumber%, "sub name = "; subname$
    PRINT #errorfilenumber%, "dataline$ = "; dataline$
    PRINT #errorfilenumber%, z$; "("; LEN(z$); ")"; "error code="; ERR
    PRINT #errorfilenumber%, "--------------------"
    FLUSH #errorfilenumber%
errorcounter% = errorcounter% + 1
errorcounter$ = LTRIM$(STR$(errorcounter%))
COLOR red, screenbackcolour
LOCATE 20, 68: PRINT SPACE$(11)
LOCATE 20, 68: PRINT "ERROR"; ERR; "/" + errorcounter$
LOCATE 21, 68: PRINT SPACE$(11)
LOCATE 21, 68: PRINT subname$
COLOR yellow, screenbackcolour
LOCATE 24, 25: PRINT "SYSTEM ERROR - try <enter> to recover, else F1 to quit"
COLOR screenforecolour, screenbackcolour
DO
   keyy$ = INKEY$
LOOP WHILE LEN(keyy$) = 0
LOCATE 20, 68: PRINT SPACE$(11)
LOCATE 24, 25: PRINT SPACE$(54);
RESUME NEXT

REM ========= DEFINE ALL THE GOSUB/KEY  ROUTINES ============
key1line:
  BEEP
  REM empty keyboard buffer
  DO: LOOP UNTIL LEN(INKEY$) = 0
  quit$ = "SET"  :REM enables quitting *without* printing
  CALL terminate
  RETURN
REM --------------------------
key2line:
  DO: LOOP UNTIL LEN(INKEY$) = 0
  REM (alarms toggle) set flag to be picked up in the main loop
  REM AlarmMode$ is used to enable beep alarm (see alarm SUB)
  IF alarmmode$ = "ON" THEN
```

```
      alarmmode$ = "OFF"
      LOCATE 24, 11: PRINT SPACE$(3)
      COLOR red, screenbackcolour
      LOCATE 24, 11: PRINT "OFF"
  ELSE
      alarmmode$ = "ON"
      LOCATE 24, 11: PRINT SPACE$(3)
      COLOR white, screenbackcolour
      LOCATE 24, 11: PRINT "ON"
  END IF
  COLOR screenforecolour, screenbackcolour
RETURN

REM ----------------------------
key3line:
  DO: LOOP UNTIL LEN(INKEY$) = 0
REM tests the alarms on screen (see Looptwo calls alarmtester SUB)
    alarmtest$="SET"
    call message("Just testing the visible alarms")
RETURN

REM ----------------------------
key4line:
  REM scroll drug file
  DO: LOOP UNTIL LEN(INKEY$) = 0
  screenmode$ = "OFF"
  keymode$ = "FOUR"
  CALL KeyHandler
RETURN

REM ----------------------------
key5line:
  REM drug screen
  DO: LOOP UNTIL LEN(INKEY$) = 0
  drugmode$="ON" :REM enables printing fo drug list (in terminate SUB)
  REM note that drugmode$ is also set with F6(blood) and F7(urine)
  screenmode$ = "OFF"
  keymode$ = "FIVE"
  CALL KeyHandler
RETURN

REM ----------------------------
key6line:
  REM blood loss
  DO: LOOP UNTIL LEN(INKEY$) = 0
  drugmode$="ON"
  screenmode$ = "OFF"
  keymode$ = "SIX"
  CALL KeyHandler
RETURN
```

```
REM ---------------------------
key7line:
  REM urine vol
  DO: LOOP UNTIL LEN(INKEY$) = 0
  drugmode$="ON"
  screenmode$ = "OFF"
  keymode$ = "SEVEN"
  CALL KeyHandler
RETURN

REM ---------------------------
key8line:
  REM help - scrolls the help.dat file
  DO: LOOP UNTIL LEN(INKEY$) = 0
  SHARED helpfile$
  REM helpfile$ = "c:help.dat"
  screenmode$ = "OFF"
  keymode$ = "EIGHT"
  CALL KeyHandler
RETURN

REM ---------------------------
key9line:
REM   AGE
  DO: LOOP UNTIL LEN(INKEY$) = 0
  screenmode$ = "OFF"
  keymode$ = "NINE"
  CALL KeyHandler
  REM erase the ?age on screen
  LOCATE 24, 2: PRINT SPACE$(4)
  COLOR white, blue
  LOCATE 24, 1: PRINT "AGE"; age%
  COLOR green, blue
RETURN

REM ---------------------------
key10line:
REM printing
  BEEP
  printRecord$ = "SET" :REM enables printing the anes record
  CALL terminate
RETURN

REM ============= END OF MAIN MODULE ==========

REM --------------place all the INCLUDE files--------
REM insert a couple of SEGMENT commands to ensure
REM we don't overrun the segment boundaries
```

```
  $INCLUDE  "a5-draw.pb"
  $INCLUDE  "a5-lib.pb"
$SEGMENT
  $INCLUDE  "a5-keyf.pb"
  $INCLUDE  "a5-plot.pb"
$SEGMENT
  $INCLUDE  "a5-DXas3.pb"

REM -----------------------------------------------
```

Chapter 12

Subroutines

ch-subs.tex

This is an alphabetical list of all the subroutines EXCEPT those in the Datex AS/3 module (see Chapter 13, page 178).

12.1 Alarms

```
SUB alarms
subname$ = "alarms"
REM called when plotting dat to screen
REM ie called by SUBs PlotBPdataNow, PlotNIBPdataNow
REM only makes NOISE if alarmmode$="ON" (enabled by F2 key)
REM
    IF alarmnibp$ = "SET" THEN
      REM alarmnibp$ set in either SUB plotNIBPdataNow or in datexas3 sub
      REM not doing anything at present
       alarmnibp$ = ""
    END IF
REM -----------------------------
LOCATE 20, 68: PRINT SPACE$(12)
COLOR red, screenbackcolour
IF hrecg > 2 AND hrecg < 50 THEN
         LOCATE 20, 68: PRINT "Hr"
         subname$ = "alrm-Hrecg"
         IF hrecg < 45 THEN code = 1
         END IF
IF hroxim > 2 AND hroxim < 50 THEN
         LOCATE 20, 68: PRINT "Hr"
         subname$ = "alrm-Hrox"
         IF hroxim < 45 THEN code = 1
         END IF
IF abp1s > 2 AND abp1s < 95 THEN
         LOCATE 20, 70: PRINT "Bp"
         subname$ = "alrm-bp1s1"
         IF abp1s < 90 THEN code = 1
```

```
         END IF
IF abp1s > 2 AND abp1s >= 185 THEN
         LOCATE 20, 70: PRINT "Bp"
         subname$ = "alrm-bp1s2"
         IF abp1s >= 200 THEN code = 1
         END IF
IF nibps > 2 AND nibps < 95 THEN
         LOCATE 20, 70: PRINT "Bp"
         subname$ = "alrm-nibp1"
         IF nibps < 90 THEN code = 1
         END IF
IF nibps > 2 AND nibps >= 185 THEN
         LOCATE 20, 70: PRINT "Bp"
         subname$ = "alrm-nibp2"
         IF nibps >= 200 THEN code = 1
         END IF

IF etco2 > 8 THEN
         LOCATE 20, 72: PRINT "CO2"
         subname$ = "alrm-etco2"
         IF etco2 > 10 THEN code = 1
         END IF

IF tvexp > 2 AND tvexp < 150 THEN
         LOCATE 20, 75: PRINT "Tv"
         subname$ = "alrm-tvex"
         IF tvexp < 100 THEN code = 1
         END IF
IF rr < 7 THEN
         LOCATE 20, 77: PRINT "Rr"
         subname$ = "alrm-rr1"
         IF rr < 5 THEN code = 1
         END IF

REM --------- line 21--------------
LOCATE 21, 68: PRINT SPACE$(13)
IF fio2 > 2 AND fio2 < 28 THEN
         LOCATE 21, 68: PRINT "FIO2"
         subname$ = "alrm-fio2"
         IF fio2 < 25 THEN code = 1
         END IF

IF sat > 2 AND sat < 93 THEN
         LOCATE 21, 72: PRINT "Sat"
         subname$ = "alrm-sat"
         IF sat < 90 THEN code = 1
         END IF

REM  Min Vol
IF rr * tvexp > 2 AND rr * tvexp < 2000 THEN
```

```
         LOCATE 21, 75: PRINT "MinVol"
         subname$ = "alrm-MV"
         IF rr * tvexp < 1000 THEN code = 1
         END IF

COLOR screenforecolour, screenbackcolour
IF code = 1 AND alarmmode$ = "ON" THEN
REM alarmmode$ is toggled on/off by F2key (via alarmhandler sub)
        CALL alarmsound
        subname$ = "alrm-c"
        code = 0
        END IF
END SUB
```

12.2 AlarmSound

```
SUB alarmsound
subname$ = "al-sound"
BEEP
END SUB
```

12.3 AlarmTester

```
 SUB alarmtester
 REM subname$ = "alarmtster"
 REM ? activated in LoopOne
 REM called from F3key and/or loopOne/LoopTwo using alarmtest$ as flag
 REM    IF keyy$ = CHR$(0) + CHR$(30) then .....:REM = ALT-A
 REM  tests the alarm codes on screen
      REM first save those values which are not routinely saved
        oldetco2 = etco2: oldtvexp = tvexp: oldrr = rr
      REM now set some abnormal values to trigger the alarm
        hrecg = 25: hroxim = 25: abp1s = 50: sat = 50: tvexp = 50
        rr = 3: fio2 = 15: etco2 = 150
      CALL alarms: SLEEP 5
      REM reset alarmtest Flag
      alarmtest$ = ""
      REM clear the alarm window
      LOCATE 20, 68: PRINT SPACE$(12)
      LOCATE 21, 68: PRINT SPACE$(13)
      REM restore the true variables
      hrecg = oldhrecg: hroxim = oldhroxim
      sat = oldsat: tvexp = oldtvexp: rr = oldrr
      fio2 = oldfio2: etco2 = oldetco2
 END SUB
```

12.4 ClearNIBPnow

```
SUB clearNIBPnow
REM called by A5-DXas3 to erase the NIBP bar
REM locate the BPnow window
VIEW (bpnowx1, bpnowy1)-(bpnowx2, bpnowy2)
WINDOW (0, 0)-(640, deltabp)
t=500
FOR delta = -40 TO 40 STEP 10
   LINE (t + delta, oldnibpd)-(t + delta, oldnibps), white
NEXT delta
END SUB
```

12.5 DiskSpace (bytes#)

```
SUB diskspace (bytes#)
REM called by :-
REM this returns the amount of remaining hard disk space
REM ----------------
REM see the PB example under call interrupts (same)
REM -----------------
REM uses INTERRUPT 21h (=33); function 36h (54) in HIGH byte of AX (ie AH)
REM uses disk DRIVE code in LOW byte of DX (ie in DL)
REM 0 = default drive;  1 = A drive;  2 = B drive;  3 = C drive....etc
REM If no disk is in the drive, this locks computer, therefore
REM check presence of disk first
    REM this is PowerBASIC 3.5
    AX%=1:BX%=2:CX%=3:DX%=4
    REG AX%, &h3600           :REM AX
    REG DX%, 3                :REM DX
    CALL INTERRUPT &h21       :REM =33
    AXout%=REG(AX%)
    BXout%=REG(BX%)
    CXout%=REG(CX%)

 freespace#=CXout%*AXout%
 freespace#=freespace#*BXout%
 bytes#=freespace#
END SUB
```

12.6 DrawBPwindow

```
SUB drawbpwindow
REM called by SUB Setupscreen
REM ----- draw the window -------------
VIEW (bpx1, bpy1)-(bpx2, bpy2), windowcolour, windowbordercolour
WINDOW (0, 0)-(deltatime, deltabp)
REM ------draw vertical time lines------
```

```
REM calculate when the next quarter/hr marker is (nq secs)
nq = 15 * 60 - starttime MOD (15 * 60)
hq = 60 * 60 - starttime MOD (60 * 60)
REM draw the 1/4 hr markers
FOR g = nq TO 5400 STEP 900
   IF g = hq OR g = hq + 3600 THEN
        LINE (g - 5, 0)-(g - 5, deltabp), bpvline
        LINE (g, 0)-(g, deltabp), bpvline
        LINE (g + 5, 0)-(g + 5, deltabp), bpvline
    ELSE
       LINE (g - 5, 0)-(g - 5, deltabp), bpvline, , &H707
       LINE (g, 0)-(g, deltabp), bpvline, , &H707
       LINE (g + 5, 0)-(g + 5, deltabp), bpvline, , &H707
   END IF
NEXT g
REM draw the horizontal lines
CALL drawbpwindowhlines
END SUB
```

12.7 DrawBPwindowHlines

```
SUB drawbpwindowhlines
REM called by SUB drawBPwindow
REM draw the horizontal lines in the BPwindow
red = 4
LINE (0, 150)-(deltatime, 150), fio2line, , &H707
LINE (0, 100)-(deltatime, 100), bphline
LINE (0, 50)-(deltatime, 50), fio2line, , &H707
LINE (0, 33)-(deltatime, 33), fio2line, , &H707
REM blue 25% o2 line
LINE (0, 20)-(deltatime, 20), fio2line
END SUB
```

12.8 DrawCO2window

```
SUB drawco2window
REM called by SUB Setupscreen
REM create the window
VIEW (co2x1, co2y1)-(co2x2, co2y2), windowcolour, windowbordercolour
WINDOW (0, 0)-(deltatime, deltaco2)
REM
REM draw vertical time lines
REM calculate when the next quarter/hr marker is (nq secs)
nq = 15 * 60 - starttime MOD (15 * 60)
hq = 60 * 60 - starttime MOD (60 * 60)
REM draw the 1/4 hr markers
FOR g = nq TO 5400 STEP 900
   IF g = hq OR g = hq + 3600 THEN
```

```
        LINE (g - 5, 0)-(g - 5, deltaco2), bpvline
        LINE (g, 0)-(g, deltaco2), bpvline
        LINE (g + 5, 0)-(g + 5, deltaco2), bpvline
    ELSE
       LINE (g - 5, 0)-(g - 5, deltaco2), bpvline, , &H707
       LINE (g, 0)-(g, deltaco2), bpvline, , &H707
       LINE (g + 5, 0)-(g + 5, deltaco2), bpvline, , &H707
   END IF
NEXT g
REM draw the horizontal lines (range 0 - 8 KPa)
REM dotted lines (use &H707) / full lines = nothing
red = 4
LINE (0, 6)-(deltatime, 6), bphline, , &H707
LINE (0, 5)-(deltatime, 5), bphline
LINE (0, 4)-(deltatime, 4), bphline, , &H707
LINE (0, 2)-(deltatime, 2), bphline, , &H707
END SUB
```

12.9 DrawHlinesBPnow

```
SUB drawhlinesbpnow
REM called by SUB DrawNowWindow
REM locate the BP window
VIEW (bpnowx1, bpnowy1)-(bpnowx2, bpnowy2)
WINDOW (0, 0)-(640, deltabp)
REM
REM draw all the horiz  lines first
LINE (0, 150)-(640, 150), fio2line, , &H707
LINE (0, 100)-(640, 100), bphline
LINE (0, 50)-(640, 50), fio2line, , &H707
LINE (0, 33)-(640, 33), fio2line, , &H707
REM blue 25% o2 line
LINE (0, 21)-(640, 21), fio2line
END SUB
```

12.10 DrawNowWindow

```
SUB drawNowWindow
REM called by SUB Setupscreen
VIEW (bpnowx1, bpnowy1)-(bpnowx2, bpnowy2), windowcolour, windowbordercolour
WINDOW (0, 0)-(640, deltabp)
REM draw the horizontal lines (same as for BP window)
red = 4
LINE (0, 150)-(640, 150), fio2line, , &H707
LINE (0, 100)-(640, 100), bphline
LINE (0, 50)-(640, 50), fio2line, , &H707
LINE (0, 33)-(640, 33), fio2line, , &H707
REM blue 25% o2 line
```

```
LINE (0, 21)-(640, 21), fio2line
END SUB
```

12.11 DrawTimeMarks

```
SUB drawTimeMarks
REM called by SUB Setupscreen
REM writes the hour and half-hour times at top of screen
REM above BPwindow
REM ----------- put TIME markers on ------------------------
REM INTEGER division
REM remember that  INT(a/b) is the same as (a\b) in BASIC
REM therefore the intiger part is given by using \  and
REM   the REMAINDER is given by using MOD
REM -----------
REM delete the old times on line 1
LOCATE 1: PRINT SPACE$(80)
REM calculate when the next HOUR markers are  (hq secs)
hq = 60 * 60 - starttime MOD (60 * 60)
REM find the first exact hour
REM 89=79*(1.5/hh)
firsthourcol% = (4 + INT((hq / 79))): REM used to be 79 when hh=1.5 hrs 89
colB%=firsthourcol%  :REM this is the second col
LOCATE 1, firsthourcol%
COLOR green, blue
a$ = RIGHT$("00" + LTRIM$(STR$(((starttime \ 3600) + 1) MOD 24)), 2) + ":00"
PRINT a$
REM ----------------------
REM find the second exact hour
col% = firsthourcol% + 45
IF col% < 66 THEN
      LOCATE 1, col%
      B$ = RIGHT$("00" + LTRIM$(STR$(((starttime \ 3600) + 2) MOD 24)), 2) + ":00"
      PRINT B$
      REM PRINT STR$(INT(starttime / 3600) + 2) + ":00"
END IF
REM ----------------
REM find the halfhour AFTER the first hr
col% = firsthourcol% + 23
IF col% < 66 THEN
     LOCATE 1, col%
     C$ = RIGHT$("00" + LTRIM$(STR$(((starttime \ 3600) + 1) MOD 24)), 2) + ":30"
     PRINT C$
     REM PRINT STR$(INT(starttime / 3600) + 1) + ":30"
END IF
REM -------------
REM find the halfhour BEFORE the first hr
col% = firsthourcol% - 23
colA%=col%  :REM this is the forst possible col
```

```
IF col% > 2 THEN
     LOCATE 1, col%
     d$ = RIGHT$("00" + LTRIM$(STR$(((starttime \ 3600)) MOD 24)), 2) + ":30"
     PRINT d$
     REM PRINT STR$(INT(starttime / 3600)) + ":30"
END IF
REM ---------
REM print the name of the Gdatafile to left if enough room
IF colA% >=14 OR (colA% <= 2 AND colB% >=14) then
   COLOR white, screenbackcolour
   LOCATE 1, 1: PRINT Gdatafilename$; : REM print filename to screen
   COLOR screenforecolour, screenbackcolour
ELSE
   COLOR white, screenbackcolour
   LOCATE 1, 69: PRINT Gdatafilename$; : REM print filename to screen
   COLOR screenforecolour, screenbackcolour
END IF
 REM ------------- end of time markers ------------------
END SUB
```

12.12 DrawTVwindow

```
SUB drawTVwindow
REM called by SUB Setupscreen
REM draw on the calibration values
REM create the window
VIEW (tidalvolx1, tidalvoly1)-(tidalvolx2, tidalvoly2), windowcolour, windowbordercolour
WINDOW (0, 0)-(deltatime, deltatidalvol)
REM
REM draw vertical time lines
REM calculate when the next quarter/hr marker is (nq secs)
nq = 15 * 60 - starttime MOD (15 * 60)
hq = 60 * 60 - starttime MOD (60 * 60)
REM draw the 1/4 hr markers
FOR g = nq TO 5400 STEP 900
   IF g = hq OR g = hq + 3600 THEN
        LINE (g - 5, 0)-(g - 5, deltatidalvol), bpvline
        LINE (g, 0)-(g, deltatidalvol), bpvline
        LINE (g + 5, 0)-(g + 5, deltatidalvol), bpvline
    ELSE
       LINE (g - 5, 0)-(g - 5, deltatidalvol), bpvline, , &H707
       LINE (g, 0)-(g, deltatidalvol), bpvline, , &H707
       LINE (g + 5, 0)-(g + 5, deltatidalvol), bpvline, , &H707
   END IF
NEXT g
REM draw the horizontal lines (range 0 - 1000)
red = 4
LINE (0, 750)-(deltatime, 750), bphline, , &H707: REM dotted line
LINE (0, 500)-(deltatime, 500), bphline: REM solid line
```

```
LINE (0, 250)-(deltatime, 250), bphline, , &H707: REM dotted line
END SUB
```

12.13 DrawVapourWindow

```
SUB drawvapourwindow
REM called by SUB Setupscreen
REM draw on the calibration values
REM create the window
VIEW (vapourx1, vapoury1)-(vapourx2, vapoury2), windowcolour, windowbordercolour
WINDOW (0, 0)-(deltatime, deltavapour)
REM
REM draw vertical time lines
REM calculate when the next quarter/hr marker is (nq secs)
nq = 15 * 60 - starttime MOD (15 * 60)
hq = 60 * 60 - starttime MOD (60 * 60)
REM draw the 1/4 hr markers
FOR g = nq TO 5400 STEP 900
   IF g = hq OR g = hq + 3600 THEN
        LINE (g - 5, 0)-(g - 5, deltavapour), bpvline
        LINE (g, 0)-(g, deltavapour), bpvline
        LINE (g + 5, 0)-(g + 5, deltavapour), bpvline
    ELSE
       LINE (g - 5, 0)-(g - 5, deltavapour), bpvline, , &H707
       LINE (g, 0)-(g, deltavapour), bpvline, , &H707
       LINE (g + 5, 0)-(g + 5, deltavapour), bpvline, , &H707
   END IF
NEXT g
REM draw the horizontal lines (range 0 - 4%)
red = 4
REM LINE (0, 8)-(deltatime, 5), bphline, , &H707
REM LINE (0, 6)-(deltatime, 6), bphline, , &H707
LINE (0, 3)-(deltatime, 3), bphline, , &H707: REM dotted line
LINE (0, 2)-(deltatime, 2), bphline, , &H707
LINE (0, 1)-(deltatime, 1), bphline: REM solid line
END SUB
```

12.14 Housekeeping

```
SUB housekeeping
subname$ = "housek1":call printsubname
REM  called from LoopOne
oldHKtime& = TIMER: REM resetting the HK timer

        REM ******** SAVING GNU data ******************************
        REM  save data (every 40 secs) in the GNU format in columns
        REM collect the first dataset only after at least 2 circuits to avoid
        REM having the first set with mainly ZEROS
```

```
subname$ = "housek2"
subname$ = "housek3"
        REM ---------------------
        REM if lasthour (ie last20mins)
            IF  windowtime$ = "lasthour" THEN CALL savelasthourdata
            REM this works OK because Housekeeping is called every 10 secs
            REM from LoopOne
       REM -------LAST 20 MINS DATA------------------
       REM now save data if in last 20 mins of the hour
       REM save the last 20 mins (less 100 secs) of the 1.33 hr period (hh)
       REM ie. we replay the last 1100 secs (= 20*60-100)
       REM The 100 secs is to take into account the fact that the window cycles
       REM 100 secs before the end of the window
       REM note that the full window (1.3333 hrs) = 4800 secs
subname$ = "housek4"
   REM trigger the opening of the lasthourdatafile
   IF currenthour% = 1 THEN
       IF j > (3600 - 1100) AND j <= 3600 AND windowtime$ = "" THEN
             windowtime$ = "lasthour"
             LastHourDataFilename$ = "lasthour.dat"
             LastHourDataFilenumber% = FREEFILE
             OPEN LastHourDataFilename$ FOR OUTPUT AS #LastHourDataFilenumber%
             lasthourstarttime = (INT(TIMER) - starttime): REM this =j
             END IF
   ELSE
        REM ie  for hours 2 and greater
        IF j > (4700 - 1100) AND j <= 4700 AND windowtime$ = "" THEN
             windowtime$ = "lasthour"
             LastHourDataFilename$ = "lasthour.dat"
             LastHourDataFilenumber% = FREEFILE
             OPEN LastHourDataFilename$ FOR OUTPUT AS #LastHourDataFilenumber%
             lasthourstarttime = (INT(TIMER) - starttime): REM this =j
             END IF
  END IF
  REM ********************************
  REM trigger the making of a new screen
  IF currenthour% = 1 THEN
      REM start new window when J exceeds 3600
      IF j > 3600 THEN
           REM print  a new screen
subname$ = "housek14"
           lasthourendtime = (INT(TIMER) - starttime)
subname$ = "housek15"
           CALL PrintNewScreen
           REM note the last20mins replay is called from  PrintNewScreen sub
     END IF
 ELSE
     REM ie for hours 2 and greater
     REM start new window when J exceeds 4700 (ie just before end of main window)
     IF j > 4700 THEN
```

```
           REM we have about 100 sec leeway before hitting the end
           REM print new screen and replay the last 20 mins
           REM note that the replay is also called from  PrintNewScreen sub
           lasthourendtime = (INT(TIMER) - starttime)
           CALL PrintNewScreen
     END IF
 END IF
REM ******************************
subname$ = "housek16"

END SUB
```

12.15 KeyHandler

```
SUB KeyHandler
subname$ = "KeyHand"
 REM called by pressing a Fkey (3,4,5,6,7,8,9)
  REM
  REM triggered from Fkeys via LoopOne/LoopTwo/Housekeeping
  REM this handles all the screen manipulation when changing screens
  REM start the clock to time how long we spend away from trend screen
  KeyClock = TIMER
  REM save the whole screen (subs in a4-lib.bas)
  CALL screen9SAVE
REM-----first, disable all the F-keys (AFTER saving the screen)-----
FOR keynum = 1 TO 10
   KEY(keynum) OFF
NEXT
REM ----------------------------------
REM now process the F-key service
IF keymode$<>"" THEN CALL KeyScreenHandler(keymode$)
REM ----------------------------------
REM on return replace stored whole screen
CALL screen9RESTORE
REM -----------
REM reset the screen and KEY flags
screenmode$="ON"  : Keymode$=""
REM-------enable the F-keys ---------
REM need to enable the F-keys as these were disabled
REM to stop us pressing the FKeys while playing with the screen
FOR keynum = 1 TO 10
   KEY(keynum) ON
NEXT
REM now switch on the KEY INFO BAR at bottom
REM COLOR green, screenbackcolour
REM KEY ON
REM COLOR screenforecolour, screenbackcolour
REM ----------------------------------
REM stop the clock (timeaway is the time spent away from the trend screen)
```

```
timeaway = TIMER - KeyClock
END SUB
```

12.16 KeyScreenHandler (k$)

```
SUB KeyScreenHandler (k$)
REM called from SUB KeyHandler
REM this used to be called screenKey in qb45 prog
subname$ = "KeyScrHa"
REM this handles which F key has been pressed
REM implement a time-out
    screentime = TIMER

SELECT CASE (UCASE$(k$))
      CASE "FOUR"
          REM used to scroll the drugfile.dat file
          REM -------------------------------
          REM clear only the bottom part of the screen
          VIEW (0, co2y1 - 1)-(639, 349), 1
          COLOR green, screenbackcolour
    LOCATE 13, 8: PRINT "  <ESC> to quit    <ARROW UP/DOWN> to scroll"
          COLOR screenforecolour, screenbackcolour
          REM  -------------------------------
          CALL scrollfile(drugfile$, 15, 24): REM  top/bottom =15,24


      CASE "FIVE"

fivetopline:
      REM = drugs
      REM try to clear only the bottom part of the screen
      VIEW (0, co2y1 - 1)-(639, 349), 1
      COLOR green, screenbackcolour
      LOCATE 13, 8: PRINT "enter appropriate key   <ESC> to quit"
      COLOR screenforecolour, screenbackcolour
      REM
      REM ------ make message box (narcotics)-----------
      REM   VIEW (50, 220)-(180, 300), 1, 15
      VIEW (20, 190)-(620, 330), 1, 15
      REM ---------------------------------------------
      REM VIEW PRINT 14 TO 24
      highlight = 14: REM defines the highlight colour (yellow)

      LOCATE 15, 5: PRINT " tomidate"
          COLOR highlight, screenbackcolour
          LOCATE 15, 5: PRINT "E"
          COLOR screenforecolour, screenbackcolour
      LOCATE 16, 5: PRINT " ropofol"
          COLOR highlight, screenbackcolour
```

```
          LOCATE 16, 5: PRINT "P"
          COLOR screenforecolour, screenbackcolour
      LOCATE 17, 5: PRINT "T iopentone"
          COLOR highlight, screenbackcolour
          LOCATE 17, 6: PRINT "H"
          COLOR screenforecolour, screenbackcolour
      LOCATE 18, 5: PRINT "mida olam"
          COLOR highlight, screenbackcolour
          LOCATE 18, 9: PRINT "Z"
          COLOR screenforecolour, screenbackcolour
      LOCATE 19, 5: PRINT "ephe rine"
          COLOR highlight, screenbackcolour
          LOCATE 19, 9: PRINT "D"
          COLOR screenforecolour, screenbackcolour
      LOCATE 20, 5: PRINT "metho amine"
          COLOR highlight, screenbackcolour
          LOCATE 20, 10: PRINT "X"
          COLOR screenforecolour, screenbackcolour
      LOCATE 21, 5: PRINT "adrena ine"
          COLOR highlight, screenbackcolour
          LOCATE 21, 11: PRINT "L"
          COLOR screenforecolour, screenbackcolour
      LOCATE 22, 5: PRINT "phen lephrine"
          COLOR highlight, screenbackcolour
          LOCATE 22, 9: PRINT "Y"
          COLOR screenforecolour, screenbackcolour

REM ------------------------
      LOCATE 15, 19: PRINT " orphine"
          COLOR highlight, screenbackcolour
          LOCATE 15, 19: PRINT "M"
          COLOR screenforecolour, screenbackcolour
      LOCATE 16, 19: PRINT " entanyl"
          COLOR highlight, screenbackcolour
          LOCATE 16, 19: PRINT "F"
          COLOR screenforecolour, screenbackcolour
      LOCATE 17, 19: PRINT " lfentanil"
          COLOR highlight, screenbackcolour
          LOCATE 17, 19: PRINT "A"
          COLOR screenforecolour, screenbackcolour
      LOCATE 18, 19: PRINT "sufentanil"
          COLOR highlight, screenbackcolour
        REM  LOCATE 18, 27: PRINT "I"
          COLOR screenforecolour, screenbackcolour
    LOCATE 19, 19: PRINT "at opine"
       COLOR highlight, screenbackcolour
       LOCATE 19, 21: PRINT "R"
       COLOR screenforecolour, screenbackcolour
    LOCATE 20, 19: PRINT " lycopyrrolate"
       COLOR highlight, screenbackcolour
```

```
       LOCATE 20, 19: PRINT "G"
       COLOR screenforecolour, screenbackcolour
    LOCATE 21, 19: PRINT " eostig/glyco"
       COLOR highlight, screenbackcolour
       LOCATE 21, 19: PRINT "N"
       COLOR screenforecolour, screenbackcolour


REM -----------------------------------------
    LOCATE 15, 35: PRINT " ecuronium"
       COLOR highlight, screenbackcolour
       LOCATE 15, 35: PRINT "V"
       COLOR screenforecolour, screenbackcolour
    LOCATE 16, 35: PRINT "a racurium"
       COLOR highlight, screenbackcolour
       LOCATE 16, 36: PRINT "T"
       COLOR screenforecolour, screenbackcolour
    LOCATE 17, 35: PRINT "m vacurium"
       COLOR highlight, screenbackcolour
       LOCATE 17, 36: PRINT "I"
       COLOR screenforecolour, screenbackcolour
    LOCATE 18, 35: PRINT "ro uronium"
       COLOR highlight, screenbackcolour
       LOCATE 18, 37: PRINT "C"
       COLOR screenforecolour, screenbackcolour
    LOCATE 19, 35: PRINT "panc ronium"
       COLOR highlight, screenbackcolour
       LOCATE 19, 39: PRINT "U"
       COLOR screenforecolour, screenbackcolour
    LOCATE 20, 35: PRINT " uxamethonium"
       COLOR highlight, screenbackcolour
       LOCATE 20, 35: PRINT "S"
       COLOR screenforecolour, screenbackcolour

       COLOR green, screenbackcolour
    LOCATE 22, 35: PRINT "other drugs/comments/info"
       COLOR highlight, screenbackcolour
       LOCATE 22, 35: PRINT "O"
       COLOR screenforecolour, screenbackcolour


REM -----------------------------------------

    LOCATE 15, 51: PRINT "1. Induction at ..."
       COLOR highlight, screenbackcolour
       LOCATE 15, 51: PRINT "1"
       COLOR screenforecolour, screenbackcolour
    LOCATE 16, 51: PRINT "2. Knife-to-skin"
       COLOR highlight, screenbackcolour
       LOCATE 16, 51: PRINT "2"
```

```
      COLOR screenforecolour, screenbackcolour
   LOCATE 17, 51: PRINT "3. Lung down"
      COLOR highlight, screenbackcolour
      LOCATE 17, 51: PRINT "3"
      COLOR screenforecolour, screenbackcolour
   LOCATE 18, 51: PRINT "4. Lung up"
      COLOR highlight, screenbackcolour
      LOCATE 18, 51: PRINT "4"
      COLOR screenforecolour, screenbackcolour
   LOCATE 19, 51: PRINT "5. End of surgery"
      COLOR highlight, screenbackcolour
      LOCATE 19, 51: PRINT "5"
      COLOR screenforecolour, screenbackcolour
   LOCATE 20, 51: PRINT "6. Transfer to ITU"
      COLOR highlight, screenbackcolour
      LOCATE 20, 51: PRINT "6"
      COLOR screenforecolour, screenbackcolour


REM ------------------
    DO
        keyy$ = INKEY$
         IF TIMER > (screentime + 10) THEN GOTO KeyScreenHandlerlastline

         SELECT CASE UCASE$(keyy$)
                CASE CHR$(27), CHR$(13)
                     REM <ESC>
                     GOTO KeyScreenHandlerlastline

                CASE "1"
                     CALL screendrug
                     LOCATE 19, 8: INPUT "enter INDUCTION time"; induction$
                     IF induction$ <> "" THEN
                         induction$ = "Induction at " + induction$
                         CALL SaveDrugData(induction$)
                         ELSE
                         induction$ = "Induction at " + TIME$
                         CALL SaveDrugData(induction$)
                         END IF
                     screentime = TIMER
                     GOTO fivetopline

                CASE "2"
                     CALL screendrug
                     LOCATE 19, 8: PRINT "KNIFE-TO-SKIN"
                     DELAY 1
                     knife$ = "Knife-to-skin"
                         CALL SaveDrugData(knife$)
                     screentime = TIMER
                     GOTO fivetopline
```

```
CASE "3"
     CALL screendrug
     LOCATE 19, 8: PRINT "LUNG DOWN"
     DELAY 1
     lungdown$ = "Lung down"
         CALL SaveDrugData(lungdown$)
     screentime = TIMER
     GOTO fivetopline


CASE "4"
     CALL screendrug
     LOCATE 19, 8: PRINT "LUNG UP"
     DELAY 1
     lungup$ = "Lung up"
         CALL SaveDrugData(lungup$)
     screentime = TIMER
     GOTO fivetopline

CASE "5"
     CALL screendrug
     LOCATE 19, 8: PRINT "END OF SURGERY"
     DELAY 1
     endd$ = "End of surgery"
         CALL SaveDrugData(endd$)
     screentime = TIMER
     GOTO fivetopline


CASE "S"
     CALL screendrug
     LOCATE 19, 8: INPUT "enter dose of SUXAMETHONIUM (mg)  "
                                                    ; suxamethonium$
     IF suxamethonium$ <> "" THEN
         suxamethonium$ = "Suxamethonium (mg) " + suxamethonium$
         CALL SaveDrugData(suxamethonium$)
     END IF
     screentime = TIMER
     GOTO fivetopline


CASE "N"
     CALL screendrug
     LOCATE 19, 8: PRINT "NEOSTIGMINE + GLYCOPYRROLATE"
     reversal$ = "Neostigmine + Glycopyrrolate"
         CALL SaveDrugData(reversal$)
     DELAY 1
     screentime = TIMER
     GOTO fivetopline
```

```
CASE "Y"
     CALL screendrug
     LOCATE 19, 8: INPUT "enter dose of PHENYLEPHRINE (mg)  "
                                                    ; phenylephrine$
     IF phenylephrine$ <> "" THEN
         phenylephrine$ = "Phenylephrine (mg) " + phenylephrine$
         CALL SaveDrugData(phenylephrine$)
     END IF
     screentime = TIMER
     GOTO fivetopline



CASE "L"
     CALL screendrug
     LOCATE 18, 8: PRINT "4mg in 50 mls = 8 microG/ml"
     LOCATE 20, 8: INPUT "enter dose of ADRENALINE (micro-gm)  "
                                                      ; adrenaline$
     IF adrenaline$ <> "" THEN
         adrenaline$ = "Adrenaline (micro-gm) " + adrenaline$
         CALL SaveDrugData(adrenaline$)
     END IF
     screentime = TIMER
     GOTO fivetopline


CASE "D"
     CALL screendrug
     LOCATE 19, 8: INPUT "enter dose of EPHEDRINE (mg)  "; ephedrine$
     IF ephedrine$ <> "" THEN
         ephedrine$ = "Ephedrine (mg) " + ephedrine$
         CALL SaveDrugData(ephedrine$)
     END IF
     screentime = TIMER
     GOTO fivetopline



CASE "X"
     CALL screendrug
     LOCATE 19, 8: INPUT "enter dose of METHOXAMINE (mg)  "
                                                    ; methoxamine$
     IF methoxamine$ <> "" THEN
         methoxamine$ = "Methoxamine (mg) " + methoxamine$
         CALL SaveDrugData(methoxamine$)
     END IF
     screentime = TIMER
```

```
        GOTO fivetopline


    CASE "H"
        CALL screendrug
        LOCATE 19, 8: INPUT "enter dose of THIOPENTONE (mg)  "
                                                    ; thiopentone$
        IF thiopentone$ <> "" THEN
            thiopentone$ = "Thiopentone (mg) " + thiopentone$
            CALL SaveDrugData(thiopentone$)
        END IF
        screentime = TIMER
        GOTO fivetopline


    CASE "E"
        CALL screendrug
        LOCATE 19, 8: INPUT "enter dose of ETOMIDATE (mg)  "; etomidate$
        IF etomidate$ <> "" THEN
            etomidate$ = "Etomidate (mg) " + etomidate$
            CALL SaveDrugData(etomidate$)
        END IF
        screentime = TIMER
        GOTO fivetopline



    CASE "M"
        CALL screendrug
        LOCATE 19, 8: INPUT "enter dose of MORPHINE (mg)  "; morphine$
        IF morphine$ <> "" THEN
            morphine$ = "Morphine (mg) " + morphine$
            CALL SaveDrugData(morphine$)
        END IF
        screentime = TIMER
        GOTO fivetopline

    CASE "F"
        CALL screendrug
        LOCATE 19, 8: INPUT "enter dose of FENTANYL (micro-gms)  "
                                                    ; fentanyl$
        IF fentanyl$ <> "" THEN
            fentanyl$ = "Fentanyl (mcg) " + fentanyl$
            CALL SaveDrugData(fentanyl$)
        END IF
        screentime = TIMER
        GOTO fivetopline

    CASE "A"
        CALL screendrug
```

```
            LOCATE 19, 8: INPUT "enter dose of ALFENTANIL (micro-gms)  "
                                                                ; alfentanil$
            IF alfentanil$ <> "" THEN
                 alfentanil$ = "Alfentanil (mcg) " + alfentanil$
                 CALL SaveDrugData(alfentanil$)
             END IF
            screentime = TIMER
            GOTO fivetopline

       CASE "P"
            CALL screendrug
            LOCATE 19, 8: INPUT "enter dose of PROPOFOL (mg)  "; propofol$
            IF propofol$ <> "" THEN
                 propofol$ = "Propofol (mg) " + propofol$
                 CALL SaveDrugData(propofol$)
             END IF
            screentime = TIMER
            GOTO fivetopline

       CASE "Z"
            CALL screendrug
            LOCATE 19, 8: INPUT "enter dose of MIDAZOLAM (mg)  "; midazolam$
            IF midazolam$ <> "" THEN
                 midazolam$ = "Midazolam (mg) " + midazolam$
                 CALL SaveDrugData(midazolam$)
             END IF
            screentime = TIMER
            GOTO fivetopline

     REM ----------------- relaxants---------
       CASE "V"
            CALL screendrug
            LOCATE 19, 8: INPUT "enter dose of VECURONIUM (mg)  "
                                                        ; vecuronium$
            IF vecuronium$ <> "" THEN
                vecuronium$ = "Vecuronium (mg) " + vecuronium$
                CALL SaveDrugData(vecuronium$)
            END IF
            screentime = TIMER
            GOTO fivetopline

       CASE "T"
            CALL screendrug
            LOCATE 19, 8: INPUT "enter dose of ATRACURIUM (mg)  "
                                                        ; atracurium$
            IF atracurium$ <> "" THEN
               atracurium$ = "Atracurium (mg) " + atracurium$
               CALL SaveDrugData(atracurium$)
             END IF
            screentime = TIMER
```

```
            GOTO fivetopline

        CASE "I"
            CALL screendrug
            LOCATE 19, 8: INPUT "enter dose of MIVACURIUM (mg)  "
                                                        ; mivacurium$
            IF mivacurium$ <> "" THEN
               mivacurium$ = "Mivacurium (mg) " + mivacurium$
               CALL SaveDrugData(mivacurium$)
             END IF
            screentime = TIMER
            GOTO fivetopline

        CASE "C"
            CALL screendrug
            LOCATE 19, 8: INPUT "enter dose of ROCURONIUM (mg)  "
                                                        ; rocuronium$
            IF rocuronium$ <> "" THEN
               rocuronium$ = "Rocuronium (mg) " + rocuronium$
               CALL SaveDrugData(rocuronium$)
             END IF
            screentime = TIMER
            GOTO fivetopline

        CASE "U"
            CALL screendrug
            LOCATE 19, 8: INPUT "enter dose of PANCURONIUM (mg)  "
                                                        ; pancuronium$
            IF pancuronium$ <> "" THEN
               pancuronium$ = "Pancuronium (mg) " + pancuronium$
               CALL SaveDrugData(pancuronium$)
             END IF
            screentime = TIMER
            GOTO fivetopline



REM -----------------------
        CASE "R"
            CALL screendrug
            LOCATE 19, 8: INPUT "enter dose of ATROPINE (mg)  "; atropine$
            IF atropine$ <> "" THEN
               atropine$ = "Atropine (mg) " + atropine$
               CALL SaveDrugData(atropine$)
             END IF
            screentime = TIMER
            GOTO fivetopline

        CASE "G"
            CALL screendrug
```

```
                    LOCATE 19, 8: INPUT "enter dose of GLYCOPYROLLATE (mg)  "
                                                                  ; glycopyrollate$
                    IF glycopyrollate$ <> "" THEN
                       glycopyrollate$ = "Glycopyrollate (mg) " + glycopyrollate$
                       CALL SaveDrugData(glycopyrollate$)
                    END IF
                    screentime = TIMER
                    GOTO fivetopline

               CASE "O"
                    CALL screendrug
                    LOCATE 20, 33: PRINT "[--------------------------------]"
                    LOCATE 19, 8: INPUT "enter DRUG/DOSE/MESSAGE "; drugdose$
                    IF drugdose$ <> "" THEN CALL SaveDrugData(drugdose$)
                    screentime = TIMER
                    GOTO fivetopline

             CASE ELSE
           END SELECT
      LOOP


    CASE "SIX"
         REM BLOOD loss
         CALL screendrug: REM this clears bottom half of screen
         LOCATE 19, 8: INPUT "enter TOTAL BLOOD LOSS (mls)  "; blood$
         IF blood$ <> "" THEN
             blood$ = "Total blood loss (mls)  " + blood$
             CALL SaveDrugData(blood$)
         END IF

    CASE "SEVEN"
         REM URINE output
         CALL screendrug
         LOCATE 19, 8: INPUT "enter URINE volume (mls)  "; urine$
         IF urine$ <> "" THEN
            urine$ = "Total urine (mls)  " + urine$
            CALL SaveDrugData(urine$)
         END IF


  CASE "EIGHT"
       REM help OPTION
   REM ------------------------------
   REM try to clear only the bottom part of the screen
   VIEW (0, co2y1 - 1)-(639, 349), 1
   COLOR green, screenbackcolour
   LOCATE 13, 8: PRINT "  <ESC> to quit    <ARROW UP/DOWN> to scroll"
   COLOR screenforecolour, screenbackcolour
   REM  ------------------------------
```

```
      CALL scrollfile(helpfile$, 15, 24)

       CASE "NINE"
            REM used to input the AGE of patient for use with MAC
            CALL screendrug
            lastage% = age%
            COLOR green, screenbackcolour
            LOCATE 17, 8: PRINT "Current age = "; lastage%
            LOCATE 21, 8: PRINT "press <enter> if OK"
            COLOR screenforecolour, screenbackcolour
    nineline:
            LOCATE 19, 8: INPUT "enter correct AGE (yrs) of patient to determine
                                                                    MAC  "; hisage
            IF hisage <> 0 THEN
                 IF (hisage < 1) OR (hisage > 150) then
                      beep
                      LOCATE 20, 8
                      COLOR yellow, screenbackcolour
                      Print "Age";hisage;"out of range ! (1-130)"
                      COLOR screenforecolour, screenbackcolour
                      locate 19,8:print space$(60)
                      goto nineline
                  END IF
                  age% = INT(hisage)

                  ELSE
                  age% = lastage%
            END IF
END SELECT
KeyScreenHandlerlastline:
END SUB
```

12.17 LoopOne

```
SUB LoopOne
subname$ = "loop1a": CALL printsubname
DO
REM --------------
REM clear message using timout
IF messagetimeFlag% > 0 THEN
     REM set the display time to 5 secs
     IF TIMER > (messageTime + 5) THEN
            LOCATE 24,33: print space$(40);
            messagetimeFlag%=0   :REM reset the Flag
     END IF
END IF
REM -------------------
REM get the j value and check for midnight
REM midnight Flag set in LoopTwo
```

```
  IF midnight$ = "SET" THEN
        j = (INT(TIMER + 86400) - starttime)
    ELSE
        j = (INT(TIMER) - starttime)
 END IF
 locate 24,69: print "j";j;

 REM print some info
rem -----TESTING-------

REM testing message
REM message string length = 35 chars only

call message(str$(nibplabel??)+space$(1)+nibpLabel$ + space$(1) + str$(nibpBIT8%)
                                          +space$(2)+str$(nibpS)+"/"+str$(nibpd))

rem  locate 22,69: print SPACE$(10)
rem  locate 22,69: print "COM2 ";LTRIM$(STR$(LEN(SpaceLabBuffer$),5) );
 locate 23,69: print SPACE$(10)
 locate 23,69: print "COM1 ";LTRIM$(STR$(LEN(datexAS3buffer$),5 ));

 REM ---- save GNU data every 40 secs-------
     REM keep the time variables as & as this is a fast loop
     IF TIMER > (oldGNUtime& + 40) THEN CALL saveGNUdata

     REM ----------------------
subname$ = "loop1b1": CALL printsubname
     REM this loop is VITAL - do not remove
     REM this loop gets the data strings from the buffer
     REM once the data strings have been processed, THEN go to loop2

REM --------DATEX AS/3 Monitor-----------------
subname$ = "loop1c": CALL printsubname
REM now pick up the datex input if present (via COM port)
 IF LOC(datexAS3comportfilenumber%) > 5  OR LEN(DatexAS3Buffer$) > 5
                                                    THEN CALL DatexAS3

subname$ = "loop1d": CALL printsubname

REM ---trap flags----
     IF screenmode$ = "OFF" THEN CALL KeyHandler
REM ------------------------

    CALL LoopTwo
subname$ = "loop1f": CALL printsubname

 IF TIMER > oldHKtime& + 10 THEN CALL housekeeping

subname$ = "loop1g": CALL printsubname
```

```
     IF screenmode$ = "OFF" THEN CALL KeyHandler
REM ------------------------

LOOP
END SUB
```

12.18 LoopTwo

```
SUB LoopTwo
REM called from LoopOne
subname$ = "loop2A": CALL printsubname
REM --------------------------
REM trap some key strokes

   IF alarmtest$ = "SET" THEN CALL alarmtester: REM set by Fkey3
   IF screenmode$ = "OFF" THEN CALL KeyHandler

REM -------save buffer data---(old saveDatatoDisk)----------
subname$ = "loop2C": CALL printsubname

REM save all buffer data every 5 mins (300 secs)
IF TIMER > LastSaveDataTime& + 300 THEN
subname$ = "loop2C1": CALL printsubname
      REM must include the #filenumber with FLUSH PB3.5
      FLUSH  #Ddatafilenumber%
      FLUSH  #Gdatafilenumber%
   REM need to flush the lasthour data also in in last 20 mins
     LastSaveDataTime& = TIMER :REM reset timer
subname$ = "loop2C2": CALL printsubname
      CALL diskspace(freebytes#)
 subname$ = "loop2C3": CALL printsubname
        LOCATE 24, 18: PRINT SPACE$(14)
    IF freebytes#>4000000 THEN
        COLOR white, screenbackcolour
        LOCATE 24, 18: PRINT "DISK"; freebytes#
        COLOR screenforecolour, screenbackcolour
     ELSE
        COLOR red, screenbackcolour
        LOCATE 24, 18: PRINT "DISK"; freebytes#
        COLOR screenforecolour, screenbackcolour
     END IF
END IF
LoopTwoLastLine:
REM-------print the clock---------------
   COLOR green, screenbackcolour
   LOCATE 13, 71: PRINT TIME$
   COLOR screenforecolour, screenbackcolour

REM ------detect midnight--------
```

```
       IF midnight$ = "" AND TIMER < 60 THEN midnight$ = "SET"
       REM (see LOOPone for midnight timing and j)
 END SUB
```

12.19 MAC (N2Opercent, vapourname$, etvapour, ageofpatient%, bmac)

```
SUB mac (n2opercent, vapourname$, etvapour, ageofpatient%, bmac)
subname$ = "MAC"
REM this sub is called from ?? where?? ? vapour sub?
REM ------
REM new MAC sub using etn2o
REM returns the value of BIGMAC (bmac)
REM this is the newMAC which works correctly
REM -------
REM problem with the SaveGNUdata sub making values -100 if are zero
REM this generates a NEGATIVE MAC problem
IF etvapour < 0 THEN etvapour = .001
REM -----------
REM Determines the current value of MAC
REM using the recent paper by Mapleson (BJA, 1996, vol 76, p 179-185)
REM Effect of age on MAC in humans: a meta-analysis
REM --------------
n2o = n2opercent
v$ = vapourname$
vap = etvapour
REM IF ageofpatient% < 1 THEN ageofpatient% = 1
A% = ageofpatient%
deltaage% = A% - 40
BB = -.00269
REM -------------
REM this MAC sub is called from the end of PLOTVAPOUR sub
REM vapour is on Datex Ultima BOO and C04 (13,3) data strings
REM vapourcode$= ISO, HAL etc  = "   " when not selected
IF v$ = "" THEN mac40 = 0
IF v$ = "HAL" THEN mac40 = .75
IF v$ = "ISO" THEN mac40 = 1.17
IF v$ = "ENF" THEN mac40 = 1.63
IF v$ = "SEV" THEN mac40 = 1.8
IF v$ = "DES" THEN mac40 = 6.6
REM mac40 for N2O = 104
REM ------------
REM do N2O calculation first
REM restrict n2o to zero or above
IF n2o < 0 THEN n2o = 0
REM eqn   mac=(mac40)*10^(-0.00269* deltaage%)
macn2o = 104 * 10 ^ (BB * deltaage%)
IF macn2o <= 0 THEN
    Fmacn2o = .01: REM changed from 0 to .01  check
    ELSE
```

```
    Fmacn2o = n2o / macn2o
END IF
REM -------------------
REM do VAPOUR calc next
REM eqn   mac=(mac40)*10^(-0.00269* deltaage%)
macvapour = mac40 * 10 ^ (BB * deltaage%)
IF macvapour <= 0 THEN
    totalFmac = Fmacn2o
    ELSE
    Fmacvapour = (vap / macvapour)
    totalFmac = Fmacvapour + Fmacn2o
END IF
REM ----------------------------------------
REM do not print to screen if printing last 20 mins fast data
IF pl20mf$ = "on" THEN GOTO MAClastline
REM ----------------------------------------
A = Fmacn2o
B = Fmacvapour
c = totalFmac
REM ----------------------------------------
   COLOR green, screenbackcolour
REM cannot print digits with PRINT USING and
REM strings in same PRINT statement, so therefore
REM we have to print them separately (red if vap mac=0)
LOCATE 18, 68: PRINT SPACE$(11)
LOCATE 18, 68: PRINT "MAC ";
    IF B <= 0 THEN
       COLOR red, screenbackcolour
       PRINT USING "#.##"; B;
       COLOR green, screenbackcolour
    ELSE
       PRINT USING "#.##"; B;
    END IF
    PRINT "/";
    PRINT USING "#.##"; A
REM --- print in red if bigmac less than .86
IF c < .86 THEN
          COLOR red, screenbackcolour
      ELSE
          COLOR green, screenbackcolour
END IF
LOCATE 19, 68: PRINT SPACE$(10)
LOCATE 19, 68: PRINT "bigMAC ";
   PRINT USING "#.##"; c
REM -----------------------------
REM now return to normal screen colours
COLOR screenforecolour, screenbackcolour
MAClastline:
bmac = c
END SUB
```

12.20 Message

```
SUB message (note$)
subname$ = "mesg"
REM prints message to screen
REM message string length = 35 chars max
REM display time is 5 sec (see LoopOne)
REM also called from errorsub,..
REM beep
messageTime=TIMER
messagetimeFlag%=3  :REM set  any positive  integer  >0
COLOR yellow, screenbackcolour
LOCATE 24, 33: PRINT SPACE$(35);
LOCATE 24, 33: PRINT note$;
COLOR screenforecolour, screenbackcolour
REM PRINT #errorfilenumber%, TIME$ + space$(2)+ note$
END SUB
```

12.21 OpenFiles

```
SUB openfiles
REM
REM --------first make date/time-encoded filename (= TimeName$)---------
REM this sub generates the filename follows
REM  <yy><month><dd><h><mm>
REM where <month> is 0-9, A,B,C for 0-12 (C=12)
REM where <h> is 0-9, A-N  (O=24)
REM this gives a filename equivalent to the time to the nearest minute
REM first read the current Date & Time
t$ = TIME$: d$ = DATE$
REM ********mins***<mm>**********
minute$ = MID$(t$, 4, 2)
fmin$ = RIGHT$("00" + minute$, 2)
REM ********* hours***<h>********
h$ = LEFT$(t$, 2)
IF VAL(h$) < 10 THEN
    fh$ = RIGHT$(h$, 1)
    ELSE
    IF h$ = "10" THEN fh$ = "A"
    IF h$ = "11" THEN fh$ = "B"
    IF h$ = "12" THEN fh$ = "C"
    IF h$ = "13" THEN fh$ = "D"
    IF h$ = "14" THEN fh$ = "E"
    IF h$ = "15" THEN fh$ = "F"
    IF h$ = "16" THEN fh$ = "G"
    IF h$ = "17" THEN fh$ = "H"
    IF h$ = "18" THEN fh$ = "I"
    IF h$ = "19" THEN fh$ = "J"
```

```
    IF h$ = "20" THEN fh$ = "K"
    IF h$ = "21" THEN fh$ = "L"
    IF h$ = "22" THEN fh$ = "M"
    IF h$ = "23" THEN fh$ = "N"
    IF h$ = "24" THEN fh$ = "O"
END IF
REM ******days***<dd>****
fd$ = MID$(d$, 4, 2)
REM ******month**<m>****
fm$ = LEFT$(d$, 2)
IF VAL(fm$) < 10 THEN
    fm$ = RIGHT$(fm$, 1)
    ELSE
    IF fm$ = "10" THEN fm$ = "A"
    IF fm$ = "11" THEN fm$ = "B"
    IF fm$ = "12" THEN fm$ = "C"
END IF
REM *****year**<yy>*****
fy$ = RIGHT$(d$, 2)
REM **********************
timename$ = fy$ + fm$ + fd$ + fh$ + fmin$
REM
REM ========= now open the serial and other ports ===============
REM need to try and rationalise the #n filenames
REM try and keep #1-#10 for COM ports
REM
REM  COM1 serial port  for DatexAS3
REM  the Dnn & Gnn files are kept OPEN; all others are OPENed and then closed
REM  lasthour.dat  (output - last 20 mins)) --- keep OPEN
REM  D files D01.. all raw data --- keep  OPEN
REM  G files G01.. data for printing anes record --- OPEN
REM  drugfile$ = timename.DRG  CLOSED
REM  #22 - #31 all the BPS, BPD, HR etc .dat files (printing section)
REM  ANES-PRT.BAT  CLOSED
REM  printall.bat  CLOSED (to collect the filenames to print at end)
REM  error file OPEN (to collect error messages)

REM -----------OPEN THE SERIAL PORTS----and DATA files-----------
REM the serial buffer size is set in main module $COM 30000
REM use FREEFILE command for COM ports too

REM ---datex AS/3---COM1-------------------------
REM datex AS3  runs using  "19200,E,8,1,CS,DS"
REM but use .. N,8,.. for testing with simulator
datexAS3protocol$ = "19200,E,8,1,CS,DS"
datexAS3comportfilenumber% = FREEFILE
OPEN "COM1:" + datexAS3protocol$ FOR INPUT AS #datexAS3comportfilenumber%
REM now create and open a file for saving raw Datex AS3 data
Ddatafilename$ = timename$ + ".D01"
Ddatafilenumber% = FREEFILE
```

```
OPEN Ddatafilename$ FOR OUTPUT AS #Ddatafilenumber%
REM print header to the Ddata file
PRINT #Ddatafilenumber%, "D0n DATA file collected from DATEX AS3 monitor"
PRINT #Ddatafilenumber%, "Program = "; version$, TIME$, DATE$, "USA FORMAT: m/d/yr"

REM --G00  data file---------
REM file to capture the data in GNU format for printing to PAPER
REM use timedependent filename (yymmddhh.G01 etc)
REM now add the .G01 extension etc (as this is the *first*  .Gnn file)
Gdatafilename$ = timename$ + ".G01"
Gdatafilenumber% = FREEFILE
OPEN Gdatafilename$ FOR OUTPUT AS #Gdatafilenumber%
REM this filename is passed to the PLOTANES.EXE prog using COMMAND$

REM -------Drugfile-----------------------
REM create the drugfile, and then close it (will need to read it later)
drugfile$ = timename$ + ".DRG": REM taken out the "c:" bit
    REM ------------
    filenumber% = FREEFILE: REM dont need specific name as file is closed
    OPEN drugfile$ FOR OUTPUT AS #filenumber%
    PRINT #filenumber%, "\begin{verbatim}"
    PRINT #filenumber%, SPACE$(2); "Drug filename = "; drugfile$
    PRINT #filenumber%, SPACE$(2); DATE$; SPACE$(2); "Code: "; Gdatafilename$
    PRINT #filenumber%, SPACE$(2); LEFT$(TIME$, 5); SPACE$(2); "START"
    PRINT #filenumber%, "  ---------------------------------"
    CLOSE #filenumber%
    REM we put the \end{verbatim} part in when QUIT/PRINT (see TERMINATE sub)
    REM -------------------

REM ---------write the PRINTALL.BAT file for LaTeX----------------------
REM  the PrintAll.bat file contains the G00 files to print out at the end
REM  This file is called from the Menu-nn.bat to do the printing
REM  use the PLOTAN3a.exe (upgraded from plotan8c) to give BP at the top
     REM ?? need to use the DRIVE as part of the namestring if
     REM going to use OPEN with it etc
     PrintAll$ = "c:PrintAll.bat"
     PrintAllfilenumber% = FREEFILE
     OPEN PrintAll$ FOR OUTPUT AS #PrintAllfilenumber%
     PRINT #PrintAllfilenumber%, "plotan3a.exe" + SPACE$(1) + Gdatafilename$
     CLOSE #PrintAllfilenumber%
     REM  SUBs which write to this file:- PrintNewScreen sub, terminate sub

REM ------help file--------------------
REM this file is scrolled from the screen (F8 key)
helpfile$ = "help.dat"

REM -------error file--------------
REM make a file to collect error info/variables
REM need to keep this open??
    errorfile$ = timename$ + ".ERR"
```

```
    errorfilenumber% = FREEFILE
    OPEN errorfile$ FOR OUTPUT AS #errorfilenumber%
    PRINT #errorfilenumber%, "ERROR file " + errorfile$
    PRINT #errorfilenumber%, TIME$, DATE$
REM -------------------------------------
END SUB
```

12.22 PlotBP

```
SUB plotBP
REM
REM abp1s, abp1d, CVP, HRecg, HRoxim
REM use some names locally only - BUT why?
hrg = hrecg
hrox = hroxim
t = j
REM----------------------------
REM locate the BP window
VIEW (bpx1, bpy1)-(bpx2, bpy2)
WINDOW (0, 0)-(deltatime, deltabp)
REM----------------------------
REM plot the BP values
REM if BP is artefact then use BLUE colour
REM this algorithm is taken from SUB plotBPnow
IF abp1d < abp1s * (90 / 250) OR abp1d > abp1s * (190 / 250) THEN
     BPcolour=blue
  ELSE
     BPcolour=green
END IF
s = abp1s
d = abp1d
LINE (t, s)-(t, d), BPcolour
LINE (t + 1, s)-(t + 1, d), BPcolour
REM----------------------------
REM plot the CVP
REM adjust value to fit in range (here is 0-200), equiv to 200 pixels
REM therefore let 1 cvp=1pixel
REM t = time
r = cvp
REM do not print aberrent flush cvp values  to trend screen
IF cvp > 30 THEN GOTO skipcvp
IF cvp2s - cvp2d > 15 THEN GOTO skipcvp
REM only print if syst and diast are different (ie if CVP is real)
IF cvp2s > cvp2d THEN
   IF cvp < 0 THEN
         GOTO skipcvp
   ELSE
         REM plot the cvp
         REM r = the plotted variable in pixel-land
```

```
         PSET (t, r + 1), lightcyan
         PSET (t, r), lightcyan
         PSET (t, r - 1), lightcyan
         REM
         PSET (t + 1, r + 1), lightcyan
         PSET (t + 1, r), lightcyan
         PSET (t + 1, r - 1), lightcyan
   END IF
ELSE
   GOTO skipcvp
END IF
skipcvp:
REM ----------------------------
REM plot HRoxim (= hrox)
REM plot the HRoxim before HRecg, as if AFib then see HRoxim OK
REM do not plot zero values (eg probe off etc)
IF hrox = 0 THEN GOTO SkipTrendHRoxim
PSET (t, hrox), blue
PSET (t, hrox + 1), blue
PSET (t, hrox + 2), blue
PSET (t, hrox + 3), blue
PSET (t + 1, hrox), blue
PSET (t + 1, hrox + 1), blue
PSET (t + 1, hrox + 2), blue
PSET (t + 1, hrox + 3), blue
SkipTrendHRoxim:
REM----------------------------
REM do not plot HRecg if =0
IF hrg = 0 THEN GOTO skiptrendhrecg
REM plot HRecg (= hrg)
PSET (t, hrg), yellow
PSET (t, hrg + 1), yellow
PSET (t, hrg + 2), yellow
PSET (t, hrg + 3), yellow
PSET (t + 1, hrg), yellow
PSET (t + 1, hrg + 1), yellow
PSET (t + 1, hrg + 2), yellow
PSET (t + 1, hrg + 3), yellow
skiptrendhrecg:
REM----------------------------
REM plot the BP=100 line again
PSET (t, 100), red
REM----------------------------
REM copy the current values to memory
olds = s
oldd = d
oldhrg = hrg
oldhrox = hrox
REM----------------------------
REM draw the horiz lines again
```

```
CALL drawbpwindowhlines
END SUB
```

12.23 PlotBPdataNow

```
SUB plotBPdataNow
REM plotting in NOW window
subname$ = "pltBPdnow"
REM
REM uses my algorithm to detect aberrent values.
REM art BP1, CVP, HRECG, FIO2
REM --------------------------------
REM locate the BPnow window
VIEW (bpnowx1, bpnowy1)-(bpnowx2, bpnowy2)
WINDOW (0, 0)-(640, deltabp)
CALL drawhlinesbpnow
REM ------------------FIO2----------------------
REM plot the FIo2 bar (green -->red if below 30%)
IF fio2 < 2 THEN GOTO endfio2
t = 0
FOR delta = 0 TO 350 STEP 10
   LINE (t + delta, oldfio2)-(t + delta, oldfio2 + 4), white
   LINE (t + delta, fio2)-(t + delta, fio2 + 4), green
NEXT delta
endfio2:
oldfio2 = fio2
REM ------------------CVP----------------------
REM plot the CVP bar (lightcyan)
IF cvp > 0 THEN
    t = 0
    FOR delta = 0 TO 200 STEP 10: REM 350
       LINE (t + delta, oldcvp)-(t + delta, oldcvp + 2), white
       LINE (t + delta, cvp)-(t + delta, cvp + 2), lightcyan
    NEXT delta
    oldcvp = cvp
 ELSE
    REM erase the last line
    t = 0
    FOR delta = 0 TO 200 STEP 10: REM 350
       LINE (t + delta, oldcvp)-(t + delta, oldcvp + 2), white
    REM    LINE (t + delta, cvp)-(t + delta, cvp + 2), lightcyan
    NEXT delta
    oldcvp = cvp
END IF

REM ---------------Arterial BP-------------------------
REM plot the BP
IF abp1s  < 0 THEN GOTO skipbp
REM ---------
```

```
REM erase and print the current one
t = 500
REM colour = red if systolic BP < 100 or >= 180
IF abp1s < 100 OR abp1s >= 180 THEN
         bpnowcolour = red
    ELSE
    bpnowcolour = lightblue
END IF
REM ----------
REM use my algorithm for aberrent values, and plot them in green
REM using the  two equations from plotting systolic & diastolic
IF abp1d < abp1s * (90 / 250) OR abp1d > abp1s * (190 / 250) THEN
          bpnowcolour = green
         END IF
REM --------------------------------
REM plot the arterial BP values
REM
FOR delta = -40 TO 40 STEP 10
   LINE (t + delta, oldabp1d)-(t + delta, oldabp1s), white
   LINE (t + delta, abp1s)-(t + delta, abp1d), bpnowcolour
NEXT delta
REM plot the BP=100 line again
FOR delta = -40 TO 40 STEP 5
     PSET (t + delta, 100), red
     NEXT delta
REM
oldabp1s = abp1s
oldabp1d = abp1d
skipbp:
REM -------HR ecg-----------
REM plot HRecg ( yellow) routine
IF hrecg = 0 THEN GOTO skiphrecg
IF hrecg < 50 THEN
        hrecgcolour = red
    ELSE
        hrecgcolour = yellow
END IF
t = 300
FOR delta = -40 TO 40 STEP 10
   REM first delete old NIBP hr value (only a problem when NIBP --> artline BP)
   REM should be a neater way of doing this.
      LINE (t + delta, oldhrnibp)-(t + delta, oldhrnibp + 10), white
   REM now delete last HRecg value
     LINE (t + delta, oldhrecg)-(t + delta, oldhrecg + 10), white
     LINE (t + delta, hrecg)-(t + delta, hrecg + 10), hrecgcolour
NEXT delta
REM
oldhrecg = hrecg
skiphrecg:
CALL drawhlinesbpnow
```

```
CALL alarms
END SUB
```

12.24 PlotCO2

```
SUB plotco2
REM
t = j
REM locate the CO2/RR window
VIEW (co2x1, co2y1)-(co2x2, co2y2)
WINDOW (0, 0)-(deltatime, deltaco2)
IF etco2 = 0 THEN GOTO skiptrendco2
REM plot the fico2 values if fico2>0.5%
REM ----------
LINE (t + delta, fico2)-(t + delta, etco2), cyan
REM plot etco2 values
pix = 8 / 50
PSET (t, etco2), etco2colour
PSET (t, etco2 + 1 * pix), etco2colour
PSET (t, etco2 - 1 * pix), etco2colour
PSET (t + 1 * pix, etco2), etco2colour
PSET (t + 1 * pix, etco2 + 1 * pix), etco2colour
PSET (t + 1 * pix, etco2 - 1 * pix), etco2colour
IF fico2 > .25 THEN
  pix = 8 / 50
  PSET (t, fico2), fico2colour
  PSET (t, fico2 + 1 * pix), fico2colour
  PSET (t, fico2 - 1 * pix), fico2colour
  PSET (t + 1 * pix, fico2), fico2colour
  PSET (t + 1 * pix, fico2 + 1 * pix), fico2colour
  PSET (t + 1 * pix, fico2 - 1 * pix), fico2colour
END IF
REM draw the horizontal lines (range 0 - 8 KPa)
red = 4
REM LINE (0, 8)-(deltatime, 5), bphline, , &H707
LINE (0, 6)-(deltatime, 6), bphline, , &H707
LINE (0, 5)-(deltatime, 5), bphline
LINE (0, 4)-(deltatime, 4), bphline, , &H707
LINE (0, 2)-(deltatime, 2), bphline, , &H707
skiptrendco2:
END SUB
```

12.25 PlotFast

```
SUB PlotFast
REM for Very fast plotting from the datafile for initial 20 min replay
REM this sub copied from the DEMO-2c.bas
REM called from printLast20minsFast
```

```
REM this SUB is a combination of all the plot subs
REM these subs are usually called from the DATEX program
REM ---------
REM start with top window
REM  plotBP
REM ====================BP  WINDOW=====================================
REM abp1s, abp1d, CVP, HRecg, HRoxim
hrg = hrecg
hrox = hroxim
t = j
REM----------------------------
REM locate the BP window
VIEW (bpx1, bpy1)-(bpx2, bpy2)
WINDOW (0, 0)-(deltatime, deltabp)
REM----------------------------
REM plot the art BP values (NIBP is saved as arterial BP in lasthour.dat)
s = abp1s
d = abp1d
LINE (t, s)-(t, d), green
LINE (t + 1, s)-(t + 1, d), green
REM ----------NIBP--------------

REM----------------------------
REM plot the CVP
REM adjust value to fit in range (here is 0-200), equiv to 200 pixels
REM therefore let 1 cvp=1pixel
REM t = time
r = cvp
REM do not print aberrent flush cvp values  to trend screen
IF cvp > 30 THEN GOTO plotfastskipcvp
IF cvp < 0 THEN
        GOTO plotfastskipcvp
   ELSE
        REM plot the cvp
        REM r = the plotted variable in pixel-land
        PSET (t, r + 1), lightcyan
        PSET (t, r), lightcyan
        PSET (t, r - 1), lightcyan
        REM
        PSET (t + 1, r + 1), lightcyan
        PSET (t + 1, r), lightcyan
        PSET (t + 1, r - 1), lightcyan
   END IF
plotfastskipcvp:
REM ----------------------------
REM plot HRoxim (= hrox)
REM do not plot zero values (eg probe off etc)
IF hrox = 0 THEN GOTO SkipPFTrendHRoxim
PSET (t, hrox), blue
PSET (t, hrox + 1), blue
```

```
PSET (t, hrox + 2), blue
PSET (t, hrox + 3), blue
PSET (t + 1, hrox), blue
PSET (t + 1, hrox + 1), blue
PSET (t + 1, hrox + 2), blue
PSET (t + 1, hrox + 3), blue
SkipPFTrendHRoxim:
REM----------------------------
REM do not plot HRecg if =0
IF hrg = 0 THEN GOTO skipPFtrendhrecg
REM plot HRecg (= hrg)
PSET (t, hrg), yellow
PSET (t, hrg + 1), yellow
PSET (t, hrg + 2), yellow
PSET (t, hrg + 3), yellow
PSET (t + 1, hrg), yellow
PSET (t + 1, hrg + 1), yellow
PSET (t + 1, hrg + 2), yellow
PSET (t + 1, hrg + 3), yellow
skipPFtrendhrecg:
REM --------------------HR--------------
REM done above with BP
REM ------------------
GOTO SkipPFTrendNibpHr
REM Plot HR-NIBP
REM  ?? do we need this too ??
REM plot the HR in the same colour as the HRecg
IF nihr = 0 THEN GOTO SkipPFTrendNibpHr
PSET (t, nihr), yellow
PSET (t, nihr + 1), yellow
PSET (t, nihr + 2), yellow
PSET (t, nihr + 3), yellow
PSET (t + 1, nihr), yellow
PSET (t + 1, nihr + 1), yellow
PSET (t + 1, nihr + 2), yellow
PSET (t + 1, nihr + 3), yellow
SkipPFTrendNibpHr:
REM----------------------------
REM plot fio2
PSET (t, fio2), fio2colour
PSET (t, fio2 + 1), fio2colour
PSET (t, fio2 + 2), fio2colour
PSET (t, fio2 + 3), fio2colour
PSET (t + 1, fio2), fio2colour
PSET (t + 1, fio2 + 1), fio2colour
PSET (t + 1, fio2 + 2), fio2colour
PSET (t + 1, fio2 + 3), fio2colour
REM ----------------------------
REM plot SAT
REM do not plot sat if =100% else it comes off the top
```

```
IF sat > 99 THEN sat = 99
REM do not plot SAT if = 0
IF sat = 0 THEN GOTO SkipPFTrendSat
REM adjust value to fit in range 150-200
thissat = 5 * sat - 300
PSET (t, thissat), red
PSET (t, thissat + 1), red
PSET (t, thissat + 2), red
PSET (t, thissat + 3), red
PSET (t + 1, thissat), red
PSET (t + 1, thissat + 1), red
PSET (t + 1, thissat + 2), red
PSET (t + 1, thissat + 3), red
SkipPFTrendSat:
REM----------------------------
REM plot the BP=100 line again
PSET (t, 100), red
REM----------------------------
REM draw the horiz lines again
CALL drawbpwindowhlines
REM ======================CO2 WINDOW=======================================
REM now CO2 window
REM plotco2
REM locate the CO2/RR window
VIEW (co2x1, co2y1)-(co2x2, co2y2)
WINDOW (0, 0)-(deltatime, deltaco2)
IF etco2 = 0 THEN GOTO skipPFtrendco2
REM plot the fico2 values if fico2>0.5%
REM ----------
LINE (t + delta, fico2)-(t + delta, etco2), cyan
REM plot etco2 values
pix = 8 / 50
PSET (t, etco2), etco2colour
PSET (t, etco2 + 1 * pix), etco2colour
PSET (t, etco2 - 1 * pix), etco2colour
PSET (t + 1 * pix, etco2), etco2colour
PSET (t + 1 * pix, etco2 + 1 * pix), etco2colour
PSET (t + 1 * pix, etco2 - 1 * pix), etco2colour
IF fico2 > .25 THEN
  pix = 8 / 50
  PSET (t, fico2), fico2colour
  PSET (t, fico2 + 1 * pix), fico2colour
  PSET (t, fico2 - 1 * pix), fico2colour
  PSET (t + 1 * pix, fico2), fico2colour
  PSET (t + 1 * pix, fico2 + 1 * pix), fico2colour
  PSET (t + 1 * pix, fico2 - 1 * pix), fico2colour
END IF
REM draw the horizontal lines (range 0 - 8 KPa)
red = 4
REM LINE (0, 8)-(deltatime, 5), bphline, , &H707
```

```
LINE (0, 6)-(deltatime, 6), bphline, , &H707
LINE (0, 5)-(deltatime, 5), bphline
LINE (0, 4)-(deltatime, 4), bphline, , &H707
LINE (0, 2)-(deltatime, 2), bphline, , &H707
skipPFtrendco2:
REM ==========================TVol Window============================
REM locate the TIDAL VOL window
VIEW (tidalvolx1, tidalvoly1)-(tidalvolx2, tidalvoly2)
WINDOW (0, 0)-(deltatime, deltatidalvol)
REM configure for plotting the EXP TVol (tvexp)
REM plot the TV as a vertical line
IF tvexp = 0 THEN GOTO skipPFTV
LINE (t, 0)-(t, tvexp), blue
skipPFTV:
REM draw horiz lines again (range 0-1000)
LINE (0, 750)-(deltatime, 750), bphline, , &H707: REM dotted line
LINE (0, 500)-(deltatime, 500), bphline, , &H707: REM dotted line
     REM LINE (0, 500)-(deltatime, 500), bphline: REM solid line
LINE (0, 250)-(deltatime, 250), bphline, , &H707: REM dotted line
REM------------------
REM plot RRate (yellow)
IF rr = 0 THEN GOTO skipPFrr
REM adjust value to fit in range 0-1000
GOTO skipPFrrline
REM ----------------
r = rr * 50
PSET (t, r), yellow
PSET (t, r + 1), yellow
PSET (t, r + 2), yellow
PSET (t, r + 3), yellow
PSET (t + 1, r), yellow
PSET (t + 1, r + 1), yellow
PSET (t + 1, r + 2), yellow
PSET (t + 1, r + 3), yellow
REM -------------------
skipPFrrline:
REM ? do not need this next section
REM GOTO skipPFrr
r2 = rr - .5
r = r2 * 50
PSET (t, r), yellow
PSET (t, r + 1), yellow
PSET (t, r + 2), yellow
PSET (t, r + 3), yellow
PSET (t + 1, r), yellow
PSET (t + 1, r + 1), yellow
PSET (t + 1, r + 2), yellow
PSET (t + 1, r + 3), yellow
skipPFrr:
REM =================VAPOUR==WINDOW================================
```

```
REM vapourin = inspired vapour conc
REM vapourout = expired vapour conc
REM ================================
REM locate the VAPOUR window
VIEW (vapourx1, vapoury1)-(vapourx2, vapoury2)
WINDOW (0, 0)-(deltatime, deltavapour)
IF vapourin = 0 AND vapourout = 0 THEN GOTO skipPFvapour
IF vapourin >= vapourout THEN
    REM plot the vapourin values as a vertical line
    IF vapourcodemode$ = "notselected" THEN
                    LINE (t, 0)-(t, vapourin), green
                    ELSE
                    LINE (t, 0)-(t, vapourin), blue
                    END IF
END IF
IF vapourin = vapourout THEN GOTO nextPFvapourline
REM plot vapourout values
pix = 4 / 50: REM calibheight/pixheight  of window
pix = pix * ABS(vapourin - vapourout)
vapouroutcolour = red
PSET (t, vapourout), vapouroutcolour
REM PSET (t, vapourout - 2 * pix), vapouroutcolour
PSET (t, vapourout - 1 * pix), vapouroutcolour
PSET (t + 1 * pix, vapourout), vapouroutcolour
PSET (t + 1 * pix, vapourout + 1 * pix), vapouroutcolour
PSET (t + 1 * pix, vapourout - 1 * pix), vapouroutcolour
nextPFvapourline:
IF vapourin < vapourout THEN
    REM plot the vapourin values as a vertical line
    IF vapourcodemode$ = "notselected" THEN
                    LINE (t, 0)-(t, vapourin), green
                    ELSE
                    LINE (t, 0)-(t, vapourin), blue
                    END IF
END IF
REM draw the horizontal lines (range 0 - 4%)
red = 4
REM LINE (0, 8)-(deltatime, 5), bphline, , &H707
LINE (0, 3)-(deltatime, 3), bphline, , &H707: REM dotted line
LINE (0, 2)-(deltatime, 2), bphline, , &H707
LINE (0, 1)-(deltatime, 1), bphline, , &H707: REM dotted line
skipPFvapour:
REM ------------------------------------------------------
REM bmac is saved in file lastHour.dat
REM CALL MAC(etn2o, vapourcodemode$, vapourout, age%, bmac)
REM this returns bmac so I can save it on G01 data file if necess
REM --------------
REM plot bigMAC (yellow)
REM adjust value to fit in range (here is 0-4%), equiv to 50 pixels
REM t = time
```

```
r = bmac
IF r < = 0 THEN GOTO skipBmac
REM r = the plotted variable in pixel-land
PSET (t, r), yellow
PSET (t, r + .05), yellow
    REM PSET (t, r + 2), yellow
    REM PSET (t, r + 3), yellow
PSET (t + 1, r), yellow
PSET (t + 1, r + .05), yellow
    REM PSET (t + 1, r + 2), yellow
    REM PSET (t + 1, r + 3), yellow
skipBmac:
REM ================================================================
END SUB
```

12.26 PlotFIO2

```
SUB plotfio2
REM
t = j
REM locate the BP window
VIEW (bpx1, bpy1)-(bpx2, bpy2)
WINDOW (0, 0)-(deltatime, deltabp)
REM plot fio2
PSET (t, fio2), fio2colour
PSET (t, fio2 + 1), fio2colour
PSET (t, fio2 + 2), fio2colour
PSET (t, fio2 + 3), fio2colour
PSET (t + 1, fio2), fio2colour
PSET (t + 1, fio2 + 1), fio2colour
PSET (t + 1, fio2 + 2), fio2colour
PSET (t + 1, fio2 + 3), fio2colour
END SUB
```

12.27 PlotNIBP

```
SUB plotnibp
REM
nis = nibps
nom = nibpm
nid = nibpd
nihr = hrnibp
t = j
REM----------------------------
REM locate the BP window
VIEW (bpx1, bpy1)-(bpx2, bpy2)
WINDOW (0, 0)-(deltatime, deltabp)
REM----------------------------
```

```
REM print NIBP in red if arterial BP is also being recorded
IF oldabp1s > 0 THEN nibpcolour = red
REM-----------------------------
REM plot the NIBP values
IF nis = 0 THEN GOTO SkipTrendNibp
LINE (t - 15, nis)-(t - 15, nid), nibpcolour
LINE (t - 10, nis)-(t - 10, nid), nibpcolour
LINE (t - 5, nis)-(t - 5, nid), nibpcolour
LINE (t, nis)-(t, nid), nibpcolour
LINE (t + 1, nis)-(t + 1, nid), nibpcolour
REM --------------------
REM save the nibp data to the GNU file
REM this is done by the Datex decode SUB
SkipTrendNibp:
END SUB
```

12.28 PlotNIBPdataNow

```
SUB plotNIBPdatanow
REM plotting in the NOW window
REM NIBPs, NIBPd, NIBPd
REM
REM locate the BPnow window
VIEW (bpnowx1, bpnowy1)-(bpnowx2, bpnowy2)
WINDOW (0, 0)-(640, deltabp)
REM plot the NIBP
REM erase and print the current one
t = 500
REM colour = red if systolic BP <100
IF nibps < 100 THEN
         bpnowcolour = red
    ELSE
        bpnowcolour = lightblue
END IF
REM ----------------------
REM  look for aberrent values, and plot them in green
REM using the  two equations from plotting systolic & diastolic
REM this is same as for art bp (see ...co3 sub)
IF nibpd < nibps * (90 / 250) OR nibpd > nibps * (190 / 250) THEN bpnowcolour = green
REM ----------
REM plot the BP values
IF oldabp1s > oldnibps THEN oldnibps = oldabp1s
IF oldabp1d < oldnibpd THEN oldnibpd = oldabp1d
REM
FOR delta = -40 TO 40 STEP 10
   LINE (t + delta, oldnibpd)-(t + delta, oldnibps), white
   LINE (t + delta, nibpd)-(t + delta, nibps), bpnowcolour
NEXT delta
REM
```

```
REM plot the BP=100 line again
FOR delta = -40 TO 40 STEP 5
     PSET (t + delta, 100), red
     NEXT delta
REM -----------------------
oldnibps = nibps
oldnibpd = nibpd
REM ----------------------
CALL alarms
END SUB
```

12.29 PlotOximDataNow

```
SUB PlotOximDataNow
REM = HRoxim, SAT
REM using the BPnow window
REM
REM locate the BPnow window
VIEW (bpnowx1, bpnowy1)-(bpnowx2, bpnowy2)
WINDOW (0, 0)-(640, deltabp)
REM *********
REM draw all the horiz  lines first
CALL drawhlinesbpnow
REM ------------HRoxim----------------------
REM plot HRoxim ( blue) routine
IF hroxim = 0 THEN GOTO skiphroxim
IF hroxim < 50 THEN
        hroximcolour = red
    ELSE
         hroximcolour = blue
END IF
t = 150
FOR delta = -50 TO 40 STEP 10
   LINE (t + delta, oldhroxim)-(t + delta, oldhroxim + 10), white
   LINE (t + delta, hroxim)-(t + delta, hroxim + 10), hroximcolour
NEXT delta
oldhroxim = hroxim
skiphroxim:
REM -------------SAT-------------------------
REM plot SAT
REM stop sat being plotted if =100%, and plot 99% to make it clear
REM -------? delete this if OK now---
REM this was to fix the shunt problem as sat=100 = only 99 etc
REM but I seem to be having problems here now - why??
REM so I have now put the shunt calculation before the sat
REM       IF sat > 99 THEN
REM            realsat = sat: sat = 99
REM       END IF
REM ------
```

```
IF sat > 99 THEN sat = 99
REM adjust value to fit in range 150-200
oximsat = 5 * sat - 300
oldoximsat = 5 * oldsat - 300
t = 0
FOR delta = 0 TO 350 STEP 10
   LINE (t + delta, oldoximsat)-(t + delta, oldoximsat + 3), white
   LINE (t + delta, oximsat)-(t + delta, oximsat + 3), red
NEXT delta
REM ---- IF realsat <> 0 THEN sat = realsat
oldsat = sat
skipsat:
CALL drawhlinesbpnow
END SUB
```

12.30 PlotRR

```
SUB plotRR
t = j
REM locate the TV/RR window
VIEW (tidalvolx1, tidalvoly1)-(tidalvolx2, tidalvoly2)
WINDOW (0, 0)-(deltatime, deltatidalvol)
REM plot RRate (yellow)
IF rr < = 0 THEN GOTO skiprr
REM adjust values(0-20/min) to fit in range 0-1000
REM when rr=20 then keep it visible on scale (by printing it as 19.5 etc)
r2 = rr - .5
r = r2 * 50
PSET (t, r), yellow
PSET (t, r + 1), yellow
PSET (t, r + 2), yellow
PSET (t, r + 3), yellow
PSET (t + 1, r), yellow
PSET (t + 1, r + 1), yellow
PSET (t + 1, r + 2), yellow
PSET (t + 1, r + 3), yellow
skiprr:
END SUB
```

12.31 PlotSAT

```
SUB plotsat
REM
REM locate the BP window
VIEW (bpx1, bpy1)-(bpx2, bpy2)
WINDOW (0, 0)-(deltatime, deltabp)
REM do not plot sat if =100% else it comes off the top
IF sat > 99 THEN sat = 99
```

```
REM do not plot SAT if = 0
IF sat = 0 THEN GOTO SkipTrendSat
t = j
REM plot SAT
REM adjust value to fit in range 150-200
thissat = 5 * sat - 300
PSET (t, thissat), red
PSET (t, thissat + 1), red
PSET (t, thissat + 2), red
PSET (t, thissat + 3), red
PSET (t + 1, thissat), red
PSET (t + 1, thissat + 1), red
PSET (t + 1, thissat + 2), red
PSET (t + 1, thissat + 3), red
REM----------------
SkipTrendSat:
END SUB
```

12.32 PlotTidalVol

```
SUB plotTidalVol
REM
REM tidal vol data from U03 string
t = j
REM locate the TIDAL VOL window
VIEW (tidalvolx1, tidalvoly1)-(tidalvolx2, tidalvoly2)
WINDOW (0, 0)-(deltatime, deltatidalvol)
REM configure for plotting the EXP TVol (tvexp)
REM plot the TV as a vertical line
IF tvexp = 0 THEN GOTO skipTV
LINE (t, 0)-(t, tvexp), blue
skipTV:
REM draw horiz lines again (range 0-1000)
LINE (0, 750)-(deltatime, 750), bphline, , &H707: REM dotted line
LINE (0, 500)-(deltatime, 500), bphline, , &H707: REM dotted line
     REM LINE (0, 500)-(deltatime, 500), bphline: REM solid line
LINE (0, 250)-(deltatime, 250), bphline, , &H707: REM dotted line
REM------------------
END SUB
```

12.33 PlotTrendData

```
SUB PlotTrendData
REM called from printLast20minsFast
REM these subs are usually called from the DATEX program
REM ---------
CALL plotBP
CALL plotfio2
```

```
CALL plotsat
CALL plotco2
CALL plotTidalVol
CALL plotRR
CALL plotvapour
REM ---------
END SUB
```

12.34 PlotVapour

```
SUB plotvapour
REM vapourin = inspired vapour conc
REM vapourout = expired vapour conc
REM ================================
REM vapour is on B00 and C04 (13,3) data strings
REM but note that every other DATEX vap string alternates "   " with "ISO" etc
REM vapourcode$= ISO, HAL etc  = "   " when not selected
IF vapourcode$ = "   " THEN vapourcodemode$ = "notselected"
IF vapourcode$ = "ISO" THEN vapourcodemode$ = "ISO"
IF vapourcode$ = "ENF" THEN vapourcodemode$ = "ENF"
IF vapourcode$ = "HAL" THEN vapourcodemode$ = "HAL"
IF vapourcode$ = "DES" THEN vapourcodemode$ = "DES"
IF vapourcode$ = "SEV" THEN vapourcodemode$ = "SEV"
SELECT CASE vapourcodemode$
        CASE "ISO"
                 COLOR green, screenbackcolour
                 LOCATE 21, 2: PRINT "ISO "
                 COLOR screenforecolour, screenbackcolour
                 LOCATE 20, 4: PRINT "4%"
                 REM LOCATE 23, 4: PRINT "1%"

   CASE "ENF"
         COLOR green, screenbackcolour
         LOCATE 21, 2: PRINT "ENF "
         COLOR screenforecolour, screenbackcolour
         LOCATE 20, 4: PRINT "4%"
         REM LOCATE 23, 4: PRINT "1%"
   CASE "HAL"
         COLOR green, screenbackcolour
         LOCATE 21, 2: PRINT "HAL "
         COLOR screenforecolour, screenbackcolour
         LOCATE 20, 4: PRINT "4%"
         REM LOCATE 23, 4: PRINT "1%"
   CASE "DES"
         COLOR green, screenbackcolour
         LOCATE 21, 2: PRINT "DES "
         COLOR screenforecolour, screenbackcolour
         LOCATE 20, 4: PRINT "4%"
         REM LOCATE 23, 4: PRINT "1%"
```

```
  CASE "SEV"
         COLOR green, screenbackcolour
         LOCATE 21, 2: PRINT "SEV "
         COLOR screenforecolour, screenbackcolour
         LOCATE 20, 4: PRINT "4%"
         REM LOCATE 23, 4: PRINT "1%"
     CASE "notselected"
                 COLOR red, screenbackcolour
                 LOCATE 21, 2: PRINT "? AA"
                 LOCATE 20, 4: PRINT "  "
                 REM LOCATE 23, 4: PRINT "  "
               COLOR screenforecolour, screenbackcolour
       CASE ELSE
END SELECT
REM ========================

t = j
REM locate the VAPOUR window
VIEW (vapourx1, vapoury1)-(vapourx2, vapoury2)
WINDOW (0, 0)-(deltatime, deltavapour)
IF vapourin = 0 AND vapourout = 0 THEN GOTO skipvapour
IF vapourin >= vapourout THEN
    REM plot the vapourin values as a vertical line
    IF vapourcodemode$ = "notselected" THEN
                     LINE (t, 0)-(t, vapourin), green
                     ELSE
                     LINE (t, 0)-(t, vapourin), blue
                     END IF
END IF
IF vapourin = vapourout THEN GOTO nextvapourline
REM plot vapourout values
pix = 4 / 50: REM calibheight/pixheight  of window
pix = pix * ABS(vapourin - vapourout)
vapouroutcolour = red
PSET (t, vapourout), vapouroutcolour
REM PSET (t, vapourout - 2 * pix), vapouroutcolour
PSET (t, vapourout - 1 * pix), vapouroutcolour
PSET (t + 1 * pix, vapourout), vapouroutcolour
PSET (t + 1 * pix, vapourout + 1 * pix), vapouroutcolour
PSET (t + 1 * pix, vapourout - 1 * pix), vapouroutcolour
nextvapourline:
IF vapourin < vapourout THEN
    REM plot the vapourin values as a vertical line
    IF vapourcodemode$ = "notselected" THEN
                     LINE (t, 0)-(t, vapourin), green
                     ELSE
                     LINE (t, 0)-(t, vapourin), blue
                     END IF
END IF
REM draw the horizontal lines (range 0 - 4%)
```

```
red = 4
REM LINE (0, 8)-(deltatime, 5), bphline, , &H707
LINE (0, 3)-(deltatime, 3), bphline, , &H707: REM dotted line
LINE (0, 2)-(deltatime, 2), bphline, , &H707
LINE (0, 1)-(deltatime, 1), bphline, , &H707: REM dotted line
REM -------------
CALL MAC(etn2o, vapourcodemode$, vapourout, age%, bmac)
REM this returns bmac so I can save it on G01 data file if necess
REM --------------
REM plot bigMAC (yellow)
REM adjust value to fit in range (here is 0-4%), equiv to 50 pixels
REM t = time
r = bmac
IF r = 0 THEN GOTO skipvapour
REM r = the plotted variable in pixel-land
PSET (t, r), yellow
PSET (t, r + .05), yellow
    REM PSET (t, r + 2), yellow
    REM PSET (t, r + 3), yellow
PSET (t + 1, r), yellow
PSET (t + 1, r + .05), yellow
    REM PSET (t + 1, r + 2), yellow
    REM PSET (t + 1, r + 3), yellow
REM -------------------
skipvapour:
END SUB
```

12.35 PrintDataToScreen

```
SUB PrintDataToScreen
REM called from Main Module/LoopOne/
REM ------------------
COLOR green, screenbackcolour
REM print the CVP
IF cvp < -15 THEN GOTO CVPlastline
REM check that CVP systolic > CVP diastolic
REM (ie that we have a waveform)
IF cvp2s > cvp2d THEN
   IF cvp < 0 THEN
         COLOR red, screenbackcolour
         LOCATE 14, 69: PRINT "CVP" + SPACE$(9)
         LOCATE 14, 74: PRINT USING "###"; cvp
         COLOR green, screenbackcolour
         ELSE
         LOCATE 14, 69: PRINT "CVP" + SPACE$(9)
         LOCATE 14, 74: PRINT USING "###"; cvp
   END IF
ELSE
   LOCATE 14, 69: PRINT SPACE$(11)
```

```
END IF
CVPlastline:
REM ---------------------
REM print the etCO2
IF etco2 >= 0 THEN
   LOCATE 15, 69: PRINT "CO2" + SPACE$(9)
   LOCATE 15, 74: PRINT USING "##.#"; etco2
END IF
REM --------------------
REM print the resp rate (rr)
IF rr >= 0 THEN
   LOCATE 16, 69: PRINT "Resp" + SPACE$(8)
   LOCATE 16, 74: PRINT USING "##"; rr
END IF
REM --------------------
REM print the tidal volume (TV)
IF tvexp >= 0 THEN
    LOCATE 17, 69: PRINT "TVex" + SPACE$(8)
    LOCATE 17, 74: PRINT USING "####"; tvexp
END IF
COLOR screenforecolour, screenbackcolour
END SUB
```

12.36 PrintLast20minsFast

```
SUB printLast20minsFast
subname$ = "pl20mf"
REM this sub is called from end of PrintNewScreen sub
REM ----------------
REM set flag to indicate that we are now printing last 20 mins data
pl20mf$ = "on"
REM ----------------
REM this plays back the last 1100 secs ( a bit less than 20 mins!)
REM disable  data-processing from the buffer
REM ? need to put this much earlier ie before the start of Prt-anesSheet?
REM filemode$ = "file2closed" ? not being used
REM close the lasthour.dat file
CLOSE #LastHourDataFilenumber%: REM this is opened in Housekeeping SUB
windowtime$ = "": REM disables saving data to the lasthour file
REM --------------------------
REM now read in the lasthourdata file and display the data
LastHourDataFilenumber% = FREEFILE
OPEN LastHourDataFilename$ FOR INPUT AS #LastHourDataFilenumber%
linecount% = 0
DO UNTIL EOF(LastHourDataFilenumber%)
     linecount% = linecount% + 1
     LINE INPUT #LastHourDataFilenumber%, LastHourdata$
     REM read the first j value (6 digits) from the lasthour file
     IF linecount% = 1 THEN
```

```
         firstj% = VAL(LEFT$(LastHourdata$, 6))
     END IF
     REM -----------------------------------
     REM now translate strings to values /pass the LastHourdata$
     REM this section used to be a separate subroutine in the QB version
    REM  get it to read the first j from line 1
   REM convert strings to values
   REM jfirst% is the first j value from the file
   j = VAL(LEFT$(LastHourdata$, 6)) - firstj%
   REM make sure that j cannot be negative
   IF j < = 0 THEN j = 1
   REM ------------------------
   REM make sure that any division/multiplication is *opposite* to
   REM that performed in SUB savelasthourdata
   REM ------------------------
   nibps = VAL(MID$(LastHourdata$, 8, 3))
   nibpd = VAL(MID$(LastHourdata$, 12, 3))
   hrecg = VAL(MID$(LastHourdata$, 16, 3))
   hroxim = VAL(MID$(LastHourdata$, 20, 3))
   abp1s = VAL(MID$(LastHourdata$, 24, 3))
   abp1d = VAL(MID$(LastHourdata$, 28, 3))
   sat = VAL(MID$(LastHourdata$, 32, 3))
   fio2 = VAL(MID$(LastHourdata$, 36, 3))
   etco2 = VAL(MID$(LastHourdata$, 40, 3)) / 10
   fico2 = VAL(MID$(LastHourdata$, 44, 3)) / 10
   tvexp = VAL(MID$(LastHourdata$, 48, 4))
   rr = VAL(MID$(LastHourdata$, 53, 2))
   vapourin = VAL(MID$(LastHourdata$, 56, 3))/100
   vapourout = VAL(MID$(LastHourdata$, 60, 3))/100
   cvp = VAL(MID$(LastHourdata$, 64, 4))
   bmac = VAL(MID$(LastHourdata$, 69, 4))
   REM -------------------------------------
        REM now plot the data
        CALL PlotFast
LOOP
CLOSE #LastHourDataFilenumber%
REM re-enable the processing of data from the buffer
filemode$ = "file2open"
REM ----------------
REM disable the flag for this subroutine
pl20mf$ = ""
REM ----------------
END SUB
```

12.37 PrintNewScreen

```
SUB PrintNewScreen
subname$ = "PrtNuScr"
REM this is called from the Housekeeping subroutine
```

```
REM this sub coordinates making a new screen and FAST20min replay
REM and creating new G02, H02 files
REM ----------------------------
REM close the current datafiles
CLOSE #Gdatafilenumber%
CLOSE #Hdatafilenumber%
CLOSE #Ddatafilenumber%  :REM Datex
CLOSE #Sdatafilenumber%  :REM SpaceLabs
REM ---------------------------------
REM turn off the key at the bottom of screem
KEY OFF
REM ---------create new files for the next hour----------
    REM increment the file counter and open the next hour's file
    REM this counter is initialised (to 1) in main module
    currenthour% = currenthour% + 1

    REM now change the G01 -->G02 etc on the end of the filename
    Gdatafilename$ = LEFT$(Gdatafilename$, 10) + RIGHT$("00"
                                              + LTRIM$(STR$(currenthour%)), 2)
    OPEN Gdatafilename$ FOR OUTPUT AS #Gdatafilenumber%

    REM now change the H01-->H02 etc on the end of the filename
    Hdatafilename$ = LEFT$(Hdatafilename$, 10) + RIGHT$("00"
                                              + LTRIM$(STR$(currenthour%)), 2)
    OPEN Hdatafilename$ FOR OUTPUT AS #Hdatafilenumber%

    REM now change the S01-->S02 etc on the end of the filename
    Sdatafilename$ = LEFT$(Sdatafilename$, 10) + RIGHT$("00"
                                              + LTRIM$(STR$(currenthour%)), 2)
    Sdatafilenumber%=FREEFILE
    OPEN Sdatafilename$ FOR OUTPUT AS #Sdatafilenumber%

    REM now change the D01-->D02 etc on the end of the filename
    Ddatafilename$ = LEFT$(Ddatafilename$, 10) + RIGHT$("00"
                                              + LTRIM$(STR$(currenthour%)), 2)
    Ddatafilenumber%=FREEFILE
    OPEN Ddatafilename$ FOR OUTPUT AS #Ddatafilenumber%

REM ---------------------------------
    REM reset the GNUdata line-counter
    gnucounter% = 0: REM used in saveGNUdata sub = No of lines in G file
    REM ? rename this to Glinecounter%
    REM used to make sure that a Gdatafilename is not written
    REM to the PrintAll batch file unless there is at least one
    REM line of G data in the Gdatafile (else get TeX printing
    REM problems as can't find the text to typeset etc)
REM -----------------------------------------------------
REM  now print a copy of this filename to the PrintAll.bat file
REM  This file is the one called from the Menu batch file to do the printing
REM  This file was first opened when this program was started, and filename
```

```
REM  created by OpenFiles SUB
     filenumber% = FREEFILE
     OPEN PrintAll$ FOR APPEND AS #filenumber%
     PRINT #filenumber%, "plotan3A.exe " + Gdatafilename$
     CLOSE #filenumber%
REM -------------------------------------------------------
REM reset the starttime for the next screen
lasthourduration = (lasthourendtime - lasthourstarttime): REM in secs
IF lasthourduration > 1100 THEN
    starttime = INT(TIMER) - lasthourduration
    ELSE
    starttime = INT(TIMER) - 1100
END IF
REM -----------
REM now make new screen and print the last 20 mins (lasthour.dat)
REM first disable relevant Fkeys (? put trap in FAST plot SUB)
KEY(1) OFF: REM error problem if press key 1 while printing last 20 mins!!
CALL setupscreen
CALL printLast20minsFast
KEY(1) ON
END SUB
```

12.38 PrintSubname

```
SUB printsubname
REM to help locate errors
   COLOR red, screenbackcolour
   LOCATE 21, 55: PRINT SPACE$(10)
   LOCATE 21, 55: PRINT subname$
   COLOR screenforecolour, screenbackcolour
END SUB
```

12.39 SaveDrugData (druginfo$)

```
SUB SaveDrugData (druginfo$)
REM called from KeyScreenHandler
subname$ = "savDrugDat"
   REM use "VERBATIM" method with TeX to allow ANY keystrokes
   REM but restrict the STRING length to 33 characters to avoid going
   REM over the central Two-column line
   REM ----------------
   REM re-open the drugfile
   REM restrict string length
   IF LEN(druginfo$) > 33 THEN druginfo$ = LEFT$(druginfo$, 33)
   filenumber%=FREEFILE
   OPEN drugfile$ FOR APPEND AS #filenumber%
   PRINT #filenumber%, SPACE$(2); LEFT$(TIME$, 5); SPACE$(2); druginfo$
   CLOSE #filenumber%
```

```
END SUB
```

12.40 SaveGNUdata

```
SUB saveGNUdata
subname$ = "saveGNUd"
REM get correct value of j
j = (INT(TIMER) - starttime)
REM
REM the GNU data strings are spaced with CHR$(176)
REM ---------------
REM reset the counter oldGNUtime (used in the LoopOne sub)
oldGNUtime& = TIMER
REM ----------------
REM print the data in cols in GNU format: time (=J) in col 1
REM the GNU file must contain the true values
REM if screen is in the second & subsequent  hour then need to
REM account for the fact that we start real time data at j= 1100 secs
IF currenthour% = 1 THEN
       timej$ = RIGHT$("000000" + LTRIM$(STR$(j)), 6) + CHR$(176)
    ELSE
       timej$ = RIGHT$("000000" + LTRIM$(STR$(j - 1100)), 6) + CHR$(176)
END IF
REM-----------------------------------------------

REM adjust later for at least 5 spaces for each parameter to allow -1000
IF sat = 0 THEN sat = -99: REM needs adjusting to 5 spaces for -1000 etc
IF fio2 = 0 THEN fio2 = -99: REM only 3 spaces in G01 file
IF fin2o = 0 THEN fin2o = -99: REM only 3 spaces in the G01 file
IF etco2 = 0 THEN etco2 = -100
IF fico2 = 0 THEN fico2 = -100
IF tvexp = 0 THEN tvexp = -999
IF tvinsp = 0 THEN tvinsp = -999
IF rr = 0 OR rr > 50 THEN rr = -9
IF vapourin = 0 THEN vapourin = -100
IF vapourout = 0 THEN vapourout = -100
IF bmac = 0 THEN bmac = -99: REM  bigmac value
REM ------
REM ****  string formating went here originally
REM ------
REM adjust the data strings for BPS, BPD, HR, CVP if necessary
REM exclude certain values
REM ie if value = 0 then val = -100  etc
 IF nibps < 0 OR nibps = 0 THEN nibps = -100
 IF nibpd < 0 OR nibpd = 0 THEN nibpd = -100
 IF hrecg < 0 OR hrecg = 0 OR hrecg > 200 THEN hrecg = -100
 IF hroxim < 0 OR hroxim = 0 OR hroxim > 200 THEN hroxim = -100
REM if systolic = diastolic then exclude
IF abp1s = abp1d THEN
```

```
              abp1s = -100
              abp1d = -100
        REM       GOTO endBPnew
              END IF
IF abp1s > 150 AND abp1d < 60 THEN
            abp1s = -100
            abp1d = -100
            END IF

IF abp1s < 0 OR abp1s = 0 THEN
            abp1s = -100
            abp1d = -100
            END IF

IF abp1d < 0 OR abp1d = 0 THEN
          abp1d = -100
          abp1s = -100
          END IF

REM CVP  Do not plot if NOT monitoring CVP
IF cvp2s = 0 AND cvp2m = 0 AND cvp2d = 0 THEN cvp = -100
REM -------------------------------------
TIM$ = RIGHT$("000000" + LTRIM$(STR$(INT(TIMER))), 6) + CHR$(176)
nibps$ = RIGHT$("000" + LTRIM$(STR$(nibps)), 3) + CHR$(176)
nibpd$ = RIGHT$("000" + LTRIM$(STR$(nibpd)), 3) + CHR$(176)
hrecg$ = RIGHT$("000" + LTRIM$(STR$(hrecg)), 3) + CHR$(176)
hroxim$ = RIGHT$("000" + LTRIM$(STR$(hroxim)), 3) + CHR$(176)
abp1s$ = RIGHT$("000" + LTRIM$(STR$(abp1s)), 3) + CHR$(176)
abp1d$ = RIGHT$("000" + LTRIM$(STR$(abp1d)), 3) + CHR$(176)
sat$ = RIGHT$("000" + LTRIM$(STR$(sat)), 3) + CHR$(176)
REM use INT for the fio2 to accomodate both formats 32 and 32.2 etc
fio2$ = RIGHT$("000" + LTRIM$(STR$(INT(fio2))), 3) + CHR$(176)
fin2o$ = RIGHT$("000" + LTRIM$(STR$(INT(fin2o))), 3) + CHR$(176)
etco2$ = RIGHT$("0000" + LTRIM$(STR$(etco2)), 4) + CHR$(176)
fico2$ = RIGHT$("0000" + LTRIM$(STR$(fico2)), 4) + CHR$(176)
tvexp$ = RIGHT$("0000" + LTRIM$(STR$(tvexp)), 4) + CHR$(176)
rr$ = RIGHT$("00" + LTRIM$(STR$(rr)), 2) + CHR$(176)
vapourin$ = RIGHT$("00000" + LTRIM$(STR$(vapourin)), 5) + CHR$(176)
vapourout$ = RIGHT$("00000" + LTRIM$(STR$(vapourout)), 5) + CHR$(176)
cvp$ = RIGHT$("0000" + LTRIM$(STR$(cvp)), 4) + CHR$(176)
REM bmac is a calculated BigMac value (not extracted from a string)
bmac = (INT(bmac * 100)) / 100
bmac$ = RIGHT$("0000" + LTRIM$(STR$(bmac)), 4) + CHR$(176)
REM -------------------------------------
REM using the --- symbol stops GNUPLOT  plotting the data point
IF nibps < 0 THEN nibps$ = "---" + CHR$(176)
IF nibpd < 0 THEN nibpd$ = "---" + CHR$(176)
IF abp1s < 0 THEN abp1s$ = "---" + CHR$(176)
IF abp1d < 0 THEN abp1d$ = "---" + CHR$(176)
IF hrecg < 0 THEN hrecg$ = "---" + CHR$(176)
```

```
IF hroxim < 0 THEN hroxim$ = "---" + CHR$(176)
IF rr < 0 THEN rr$ = "--" + CHR$(176)
IF vapourin < 0 THEN vapourin$ = "----" + CHR$(176)
IF vapourout < 0 THEN vapourout$ = "----" + CHR$(176)


REM reject aberrent CVP values (see SUB plotBP)
IF cvp > 30 THEN cvp$ = "----" + CHR$(176)
IF cvp2s - cvp2d > 15 THEN cvp$ = "----" + CHR$(176)
REM
REM IF hrecg < 0 THEN hrecg$ = "---" + CHR$(176)
IF sat < 0 THEN sat$ = "---" + CHR$(176)
IF bmac < 0 THEN bmac$ = "----" + CHR$(176)
REM --------------------------------------
REM do not print any vapout values if vapour not selected
IF vapourcodemode$ = "notselected" THEN
     vapourin$ = "-----" + CHR$(176): REM 5 spaces
     vapourout$ = "-----" + CHR$(176): REM 5
     bmac$ = "-100" + CHR$(176): REM have bmac off scale if vapour not selected
END IF
REM--------------------------------------
REM print data to Gdatafile
gnu1$ = TIM$ + timej$ + hrecg$ + hroxim$ + abp1s$ + abp1d$
gnu2$ = sat$ + fio2$ + etco2$ + fico2$ + tvexp$ + rr$
gnu3$ = vapourin$ + vapourout$ + cvp$ + bmac$ + fin2o$
subname$ = "sGd#G-1"
PRINT #Gdatafilenumber%, gnu1$ + gnu2$ + gnu3$
subname$ = "sGd#G+1"
REM --------------------------------------
END SUB
```

12.41 SaveLastHourData

```
SUB savelasthourdata
subname$ = "saveLHD***":call printsubname
REM
REM needs to be slightly modified for use with DAtex AS/3 parameters
REM called from Housekeeping (every 10 secs during lasthour)
REM data spaced with CHR$(176)
REM need to save this data EXACTLY as output by DATEX monitor.
REM remember to work with STRINGS only and do NOT change variables
 timej$ = RIGHT$("000000" + LTRIM$(STR$(j)), 6) + CHR$(176)
REM ****************************************
REM need to save NIBP data in the art BP cols
IF abp1s < = 0 then abp1s = nibps
IF abp1d < = 0 then abp1d = nibpd

REM ------------
nibps$ = RIGHT$("000" + LTRIM$(STR$(nibps)), 3) + CHR$(176)
```

```
nibpd$ = RIGHT$("000" + LTRIM$(STR$(nibpd)), 3) + CHR$(176)
hrecg$ = RIGHT$("000" + LTRIM$(STR$(hrecg)), 3) + CHR$(176)
hroxim$ = RIGHT$("000" + LTRIM$(STR$(hroxim)), 3) + CHR$(176)
abp1s$ = RIGHT$("000" + LTRIM$(STR$(abp1s)), 3) + CHR$(176)
abp1d$ = RIGHT$("000" + LTRIM$(STR$(abp1d)), 3) + CHR$(176)
sat$ = RIGHT$("000" + LTRIM$(STR$(sat)), 3) + CHR$(176)
fio2$ = RIGHT$("000" + LTRIM$(STR$(fio2)), 3) + CHR$(176)
REM remove decimal point from the co2 values (3.8 --> 038 etc)
REM etco2 = INT(etco2 * 10)
REM fico2 = INT(fico2 * 10)
etco2$ = RIGHT$("000" + LTRIM$(STR$(INT(etco2 * 10))), 3) + CHR$(176)
fico2$ = RIGHT$("000" + LTRIM$(STR$(INT(fico2 * 10))), 3) + CHR$(176)
tvexp$ = RIGHT$("0000" + LTRIM$(STR$(tvexp)), 4) + CHR$(176)
rr$ = RIGHT$("00" + LTRIM$(STR$(rr)), 2) + CHR$(176)
REM remove decimal point from the vapour values (0.38 --> 038 etc)
REM vapourin = INT(vapourin * 100)
REM vapourout = INT(vapourout * 100)
vapourin$ = RIGHT$("000" + LTRIM$(STR$(INT(vapourin * 100))), 3) + CHR$(176)
vapourout$ = RIGHT$("000" + LTRIM$(STR$(INT(vapourout * 100))), 3) + CHR$(176)
REM -------------------------------------
REM include cvp and bmac: these are copied from SaveGNUdata sub
cvp$ = RIGHT$("0000" + LTRIM$(STR$(cvp)), 4) + CHR$(176)
REM bmac is my calculated BigMac value (ie not extracted from a DATEX string)
bmac = (INT(bmac * 100)) / 100
bmac$ = RIGHT$("0000" + LTRIM$(STR$(bmac)), 4) + CHR$(176)
REM ---------------------------------
REM print data to LastHourDataFile (opened in LoopTwo/?hkeeping)
REM save in same order as in SaveGNUdata sub
REM split up the strings to help the compiler
a$ = timej$ + nibps$ + nibpd$ + hrecg$ + hroxim$ + abp1s$ + abp1d$
B$ = sat$ + fio2$ + etco2$ + fico2$ + tvexp$
C$ = rr$ + vapourin$ + vapourout$ + cvp$ + bmac$
PRINT #LastHourDataFilenumber%, a$ + B$ + C$
END SUB
```

12.42 Screen9Restore

```
SUB Screen9Restore
REM modified from Don Schillian <d83@ath.forthnet.gr>
REM first clear CLS the window area before PUTing
CLS graphics: REM = cls graph
REM -------------------------
VIEW: REM this view must be just before the PUT
PUT (000, 000), ScreenTopBuf%, PSET
PUT (000, 175), ScreenBotBuf%, PSET
REM ------- define the array for the screen saver---------
REM    DIM STATIC  ScreenTopBuf%(28001)
REM    DIM STATIC  ScreenBotBuf%(28001)
REM    SHARED ScreenTopBuf%(), ScreenBotBuf%()
```

```
END SUB
```

12.43 Screen9Save

```
SUB Screen9Save
REM modified from Don Schillian <d83@ath.forthnet.gr>
 VIEW: REM first define the WHOLE screen area again BEFORE using GET
 WINDOW : REM disables any window mapping
 GET (000,000)-(639,174), ScreenTopBuf%
 GET (000,175)-(639,349), ScreenBotBuf%
REM ------- define the array for the screen saver---------
REM    DIM STATIC  ScreenTopBuf%(28001)
REM    DIM STATIC  ScreenBotBuf%(28001)
REM    SHARED ScreenTopBuf%(), ScreenBotBuf%()
END SUB
```

12.44 ScreenBOFF

```
SUB ScreenBOFF
REM saves the Bottom half of screen (screen 9)
BSx% = 0: BSy% = 168
VIEW: REM first define the WHOLE screen area again BEFORE using GET
GET (0, BSy%)-(639, 349), ScreenBotBuf%
REM --------------
REM now clear the bottom half of the screen in BLUE(1)
VIEW (0, BSy%)-(639, 349), 1
END SUB
```

12.45 ScreenBON

```
SUB ScreenBON
REM replaces ON the screen the Bottom half (screen 9)
BSx% = 0: BSy% = 168
REM first clear CLS the window area before PUTing
CLS graphics: REM = cls graph
REM -------------------------
VIEW: REM this view must be just before the PUT
PUT (0, BSy%), ScreenBotBuf%, PSET
END SUB
```

12.46 Screendrug

```
SUB screendrug
subname$ = "screendrug"
   REM called from ScreenKey routine (F5) to make a new screen
   REM clears lower half of screen for inputting drugs
   VIEW (0, co2y1 - 1)-(639, 349), 1
```

```
   COLOR green, screenbackcolour
   LOCATE 17, 8: PRINT "time = "; TIME$
   COLOR screenforecolour, screenbackcolour
END SUB
```

12.47 ScrollFile (filetoscroll$, r1&, r2&)

```
SUB scrollfile (filetoscroll$, r1&, r2&)
subname$ = "scrollfile"
REM ---------------------------------
REM modified for PowerBASIC3.5 by RWDN Oct 19 1998
REM my QB45 version is t-scroll.bas
REM works OK now
REM  scroll error fixed (Jan 25 1997)
REM note that PB3.5 does have a scroll UP and DOWN function in PB35 examples
REM
REM ----- define the array-------
Maxline& = 100  'However many lines you want as maximum
DIM startposition&(Maxline&)
REM -----------------------------
toprow& = r1&
bottomrow& = r2&
REM -----some info (see end of SUB for info from Rusty Angel)------
REM row1/col1 specify the upper left corner of the area to be scrolled.
REM row2/col2 specify the lower right corner of the area to be scrolled.
REM lines = specifies how far to scroll.
REM  +lines = scroll up (text up; arrow down),
REM  -lines = scroll down (text down, arrow up)
REM attr = color attribute for blank lines created by the scroll.
REM ----------------start of scroller----------------------------
REM COMMON SHARED topline&, bottomline&
REM COMMON SHARED toprow&, bottomrow&
REM ---------open the file to be printed--------------
filenumber% = FREEFILE
OPEN filetoscroll$ FOR INPUT AS #filenumber%
REM ------- First load the file into an array -----------------
DO UNTIL EOF(filenumber%) OR numlines& > Maxline&
    numlines& = numlines& + 1
    startposition&(numlines&) = SEEK(filenumber%)
    LINE INPUT #filenumber%, text$
LOOP
REM now catch the last line of the file
numlines& = numlines& + 1
startposition&(numlines&) = SEEK(filenumber%)
REM ----------setup scroll parameters-------------------
topline& = 1
bottomline& = topline& + (bottomrow& - toprow&)
REM  scroll (row1&, col1, row2&, col2, lines, attr)
REM scroll (toprow&, 1, bottomrow&, 80, 1, 1)
```

```
REM these parameters are used below in the INTERRUPT part
row1& = toprow&
col1 = 1
row2& = bottomrow&
col2 = 80
attr = 1
REM the lines parameter is + for scroll up/ - for scroll down
REM ---------------------------------------
REM display first part of file  (ie just that number of lines to fit window)
REM  (not needed as using LOCATE) VIEW PRINT toprow& TO bottomrow&
SEEK #filenumber%, startposition&(topline&)
FOR I = toprow& TO bottomrow& - 1
   IF NOT EOF(filenumber%) THEN
          LINE INPUT #filenumber%, text$
          linelen% = LEN(text$)
   ELSE
       REM if file shorter than display window
       COLOR 2, 1       :REM green,blue
       PRINT "End of File"; : REM -1
       COLOR 7, 1       :REM white, blue
       scrolltime& = TIMER
       DO
          REM limit time to 8 secs
          IF TIMER > scrolltime& + 8 THEN GOTO scrollfilelastline
       LOOP WHILE INKEY$ <> CHR$(27)
       GOTO scrollfilelastline:
    END IF
    REM now print the line
    LOCATE I, 1
    REM colour the relaxant lines (vecURONium, atracURONium, mivacURIUm)
    IF INSTR(1, text$, "uron") > 0 OR INSTR(1, text$, "uriu") > 0 THEN
              COLOR yellow, screenbackcolour
              PRINT text$
              COLOR screenforecolour, screenbackcolour
        ELSE
              PRINT text$;
    END IF
NEXT I
REM reset the file marker
SEEK #filenumber%, startposition&(topline&)
REM ------------now for the scrolling------------------
REM set the SOF flag to stop scrolling up at begining
SOFflag$ = "SET"
REM ---------------------------------------
start:
scrolltime& = TIMER

DO       ' wait for a keypress
   keyy$ = INKEY$
   IF TIMER > scrolltime& + 5 THEN GOTO scrollfilelastline
```

```
LOOP WHILE keyy$ = ""
IF keyy$ = CHR$(0) + CHR$(80) AND EOFflag$ = "SET" THEN GOTO start
IF keyy$ = CHR$(0) + CHR$(72) AND SOFflag$ = "SET" THEN GOTO start
SELECT CASE keyy$

   CASE CHR$(0) + CHR$(80)
       REM arrow down  (scroll up; text up: put new text on bottom line)
       IF SOFflag$ = "SET" THEN SOFflag$ = ""
       IF EOFflag$ = "SET" THEN
              GOTO start
       END IF
       IF updown$ = "UP" THEN
              topline& = topline& + 1
              bottomline& = bottomline& + 1
       END IF
       REM scroll toprow&, 1, bottomrow&, 80, 1, 1   ' scroll up (+) 1
       REM ------scroll up 1 line-----------
            lines = 1
            REM this is PowerBASIC 3.5
            AX%=1:BX%=2:CX%=3:DX%=4
            REG AX%, 256 * 6 + lines
            REG BX%, 256 * (attr MOD 8) * 16
            REG CX%, 256 * (row1& - 1) + (col1 - 1)
            REG DX%, 256 * (row2& - 1) + (col2 - 1)
            CALL INTERRUPT 16
       REM -----------------
       REM stop at the end of the file
       IF EOF(filenumber%) THEN
             COLOR 2, 1
             LOCATE bottomrow& - 1, 2: PRINT "End of File"; : REM -1
             EOFflag$ = "SET"
             updown$ = ""
             COLOR 7, 1
             GOTO start
       END IF
       REM -------------*
          IF bottomline& > numlines& THEN bottomline& = numlines&
       SEEK #filenumber%, startposition&(bottomline&)
       IF NOT EOF(filenumber%) THEN
            LINE INPUT #filenumber%, text$
            linelen% = LEN(text$)
          ELSE
             REM print some empty lines
             text$ = ""
       END IF
       LOCATE bottomrow& - 1, 1
           REM now colour the relaxant lines
           IF INSTR(1, text$, "uron") > 0 OR INSTR(1, text$, "uriu") > 0 THEN
                     COLOR yellow, screenbackcolour
                     PRINT text$
```

```
                    COLOR screenforecolour, screenbackcolour
          ELSE
          PRINT text$;
          END IF
       REM now reset the line numbering by 1
      topline& = topline& + 1
      bottomline& = bottomline& + 1
      REM set the updown flag
      updown$ = "DOWN"
      REM ------end of arrow DOWN-----------

  CASE CHR$(0) + CHR$(72)
       REM arrow up (scroll down; text down)
       IF EOFflag$ = "SET" THEN EOFflag$ = ""
       IF SOFflag$ = "SET" THEN
              GOTO start:
       END IF
       IF updown$ = "DOWN" THEN
              topline& = topline& - 1
              bottomline& = bottomline& - 1
       END IF
       REM scroll toprow&, 1, bottomrow&, 80, -1, 1   ' scroll down (-) 1
       REM ---------scroll down 1 line-----------------
            lines = -1
            REM this is PowerBASIC 3.5
            AX%=1:BX%=2:CX%=3:DX%=4

            REG AX%, 256 * 7 + (-lines)
            REG BX%, 256 * (attr MOD 8) * 16   ' was bug here but it's fixed now.
            REG CX%, 256 * (row1& - 1) + (col1 - 1)
            REG DX%, 256 * (row2& - 1) + (col2 - 1)
            CALL INTERRUPT 16
       REM -------------------------------
       REM stop at the START of the file
       IF topline& < 1 THEN
                  topline& = 0
                  COLOR 2, 1
                LOCATE toprow&, 2: PRINT "Start of File";
                COLOR 7, 1
                  SOFflag$ = "SET"
                  updown$ = ""
                GOTO start:
                 END IF

       REM -------------------------------
       REM now print the next line  at the top of window
       SEEK #filenumber%, startposition&(topline&)
       LINE INPUT #filenumber%, text$
       LOCATE toprow&, 1
           REM now colour the relaxant lines
```

```
           IF INSTR(1, text$, "uron") > 0 OR INSTR(1, text$, "uriu") > 0 THEN
                     COLOR yellow, screenbackcolour
                     PRINT text$
                     COLOR screenforecolour, screenbackcolour
           ELSE
           PRINT text$;
           END IF
       REM now reset the line numbering by 1
       topline& = topline& - 1
       bottomline& = bottomline& - 1
       updown$ = "UP"
    REM ------------------------------------

   CASE "Q", "q", CHR$(27), CHR$(13)
        GOTO scrollfilelastline

   CASE ELSE

END SELECT
GOTO start

REM --------info on the scroll SUB-----------------------
REM SCROLLING in QB
REM taken fom my RNscroll.bas
REM originally from qb-scr2.llg (kermit)
REM modified a bit by me RWDN - works OK
REM originally From: Rusty Angel <arangel@fast.net>
REM Newsgroups: comp.lang.basic.misc
REM Subject: Re: scrolling a file up & down
REM Date: Sat, 07 Sep 1996 15:48:22 -0400
REM Organization: FASTNET(tm) PA/NJ/DE Internet
REM Here is an example of using the Interrupt routine of QB 4.5 to
REM call BIOS interrupt 10H (16 decimal) to scroll the display
REM either up or down. (It's fast too.)
REM I didn't write the Scroll sub, it's from the user interface toolbox
REM that is included with PDS 7 although I did correct the bug that causes
REM the blank lines to always be black when scrolling down. (hope MS doesn't mind)

REM By specifying row1,col1 & row2,col2 you can scroll the whole screen or
REM just some portion of the screen.

REM NOTE: I Commented out a couple of lines in the Scroll sub that make sure
REM       that the rows & columns to be scrolled are within the screen boundries
REM       before calling the BIOS interrupt. You may want to put them back in.
REM -------------
REM Here is a description of the BIOS Interrupt 10h functions
REM  called by the Scroll sub.

REM Function 06h    Scroll Page Up
REM                 scroll up or initialize a display "window"
```

```
REM entry   AH      06h
REM         AL      number of lines blanked at bottom of page
REM         00h     blank entire window
REM         BH      attributes to be used on blank line
REM         CH      row    (Y) of upper left corner or window
REM         CL      column (X) of upper left corner of window
REM         DH      row    (Y) of lower right corner of window
REM         DL      column (X) of lower right corner of window
REM RETURN none
REM note 1) Push BP before scrolling, pop after (early IBM PCs didn't
REM         save BP).
REM      2) Affects current video page only.

REM -----------
REM Function 07h    Scroll Page Down
REM                 scroll down or clear a display "window"
REM entry   AH      07h
REM         AL      number of lines to be blanked at top of page
REM         00h     blank entire window
REM
REM         BH      attributes to be used on blank line
REM         CH      row    (Y) of upper left corner or window
REM         CL      column (X) of upper left corner of window
REM         DH      row    (Y) of lower right corner of window
REM         DL      column (X) of lower right corner of window
REM RETURN none
REM  --
REM Rusty Angel     arangel@fast.net
REM Bangor, Pa. USA
scrollfilelastline:
CLS
CLOSE #filenumber%
END SUB
```

12.48 SetupScreen

```
SUB setupscreen
REM -----------
SCREEN 9: REM this is 640 x 350 pixels
REM SCREEN 12 - not used
REM set general full screen colours
COLOR screenforecolour, screenbackcolour
REM paint screen blue (ie with the previously defined colours)
REM VIEW (0, 0)-(639, 349), 1, 15
VIEW ,1,15   :REM view with no parameters = whole screen
REM -----------------
REM print the Program version number
COLOR green, screenbackcolour
LOCATE 12, 68: PRINT version$
```

```
COLOR screenforecolour, screenbackcolour
REM -----------------
REM put the BP & HR  calibration on
LOCATE 2, 3: PRINT "200"
LOCATE 7, 3: PRINT "100"
LOCATE 9, 4: PRINT "50"
LOCATE 11, 4: PRINT "20"
LOCATE 12, 5: PRINT "0"
COLOR green, screenbackcolour
LOCATE 4, 2: PRINT "SAT"
LOCATE 5, 2: PRINT "BP"
LOCATE 8, 2: PRINT "HR"
LOCATE 10, 2: PRINT "FIO2"
COLOR screenforecolour, screenbackcolour
REM -----------------
REM put on the CO2 values
COLOR green, screenbackcolour
LOCATE 15, 2: PRINT "CO2"
COLOR screenforecolour, screenbackcolour
LOCATE 14, 4: PRINT "5%"
REM  LOCATE 16, 4: PRINT "0%"
REM -----------------
REM put the SAT calibration on
LOCATE 2, 68: PRINT "100%"
COLOR green, screenbackcolour
LOCATE 3, 68: PRINT "Sat"
COLOR screenforecolour, screenbackcolour
LOCATE 4, 68: PRINT "90%"
LOCATE 7, 68: PRINT "80%"
REM -----------------
REM put the Tidal vol  calibration on
COLOR green, screenbackcolour
LOCATE 18, 2: PRINT "TVol"
COLOR screenforecolour, screenbackcolour
REM LOCATE 17, 2: PRINT "1000"
REM LOCATE 20, 5: PRINT "0"
REM -----------------
REM put the VAPOUR calibration on
IF vapourcodemode$ = "notselected" THEN
        COLOR red, screenbackcolour
        LOCATE 21, 2: PRINT "? AA"
        COLOR screenforecolour, screenbackcolour
END IF
REM ----------------------
REM print the ?age reminder (age is set using F9-key)
COLOR red, screenbackcolour
LOCATE 24, 2: PRINT "?Age"
COLOR screenforecolour, screenbackcolour
REM ------------
REM print the correct alarm status (controlled by F2-key)
```

```
  IF alarmmode$ = "ON" THEN
      COLOR white, screenbackcolour
      LOCATE 24, 11: PRINT "ON"
  ELSE
      COLOR red, screenbackcolour
      LOCATE 24, 11: PRINT "OFF"
  END IF
  COLOR screenforecolour, screenbackcolour
REM -----------
REM now put the hour and half-hour time markers on the top of the screen
REM and also the current G-data filename on either L or R top corner
CALL drawTimeMarks
REM -----------------
REM  now draw the BP / CO2 / TV / Vapour windows
CALL drawbpwindow
CALL drawNowWindow
CALL drawco2window
CALL drawTVwindow
CALL drawvapourwindow
REM ----------
REM now switch on the KEY INFO BAR at bottom (in green)
COLOR green, screenbackcolour
KEY ON
COLOR screenforecolour, screenbackcolour
END SUB
```

12.49 Terminate

```
SUB terminate
subname$ = "terminate"
REM called from F1-KEY, F10-key
REM ------------------
REM first empty keyboard buffer
DO: LOOP UNTIL LEN(INKEY$) = 0
REM -------------------
REM flush the open  print buffers (to make sure all data is written
REM  to the hard drive)
   FLUSH  #Ddatafilenumber%
   FLUSH  #Gdatafilenumber%

REM -------------------
IF quit$ = "SET" THEN
    REM  Quit$ is set by pressing F1
    REM confirm the quit (do *not* use CRLF as we are on line 24)
    COLOR red, screenbackcolour
    LOCATE 24, 68: PRINT " Quit? (y/n) ";
    REM -------wait for new key press------------
    DO
       keyy$ = INKEY$
```

```
    LOOP WHILE keyy$ =""
    REM ------------------
    SELECT CASE keyy$
         CASE "Y", "y"
              BEEP
              REM close all open files
              CLOSE
              IF drugmode$ = "ON" THEN
                   REM drugfile$ is set when press F5, F6(blood), F7(urine)
                   REM write ending to the drugfile
                   filenumber%=FREEFILE
                   OPEN drugfile$ FOR APPEND AS #filenumber%
                     PRINT #filenumber%, "  ---------------------------------"
                     PRINT #filenumber%, SPACE$(2); LEFT$(TIME$, 5); SPACE$(2); "END"
                     PRINT #filenumber%, "\end{verbatim}"
                     REM now close the drug datafile so this is written to it
                   CLOSE
                   REM now enable the PRINTALL file to print the drugs
                   filenumber%=FREEFILE
                   OPEN PrintAll$ FOR APPEND AS #filenumber%
                     PRINT #PrintAllfilenumber%, "IF EXIST" + SPACE$(1) + drugfile$
                                    + SPACE$(1) + "CALL drug-11h.exe" + SPACE$(1)
                                                          + timename$ + ".DRG"
                   CLOSE
              END IF
              REM delete anes-prt.bat if it exists
              REM because with QUIT option we don't want to print
              SHELL "IF EXIST anes-prt.bat del anes-prt.bat"
              KEY OFF
              CLOSE : REM close ALL files
              END
         CASE ELSE
              REM return to main program
              COLOR green, screenbackcolour
              LOCATE 24, 69: PRINT SPACE$(11);
              z$ = INKEY$
              quit$ = ""
        END SELECT
END IF

REM --------------------------------------
IF PrintRecord$ = "SET" THEN
   REM print anes record and then exit
   REM  set when press F10
   REM idea is to create a new batch file which will call
   REM the PRINTALL.BAT file from MENU-5A.com
   REM confirm the print/quit option
   REM no CRLF as we are using line 24
   COLOR red, screenbackcolour
   LOCATE 24, 68: PRINT "Print? (y/n) ";
```

```
   REM -------wait for key press------------
   DO
      keyy$ = INKEY$
   LOOP WHILE keyy$ = ""
   REM -----------------------------------
   SELECT CASE keyy$
       CASE "Y", "y"
           BEEP
           REM close all open files
           CLOSE
           REM now create a batch-file (anes-prt.bat) to trigger printing
           REM by calling the PRINTALL.BAT file
           REM note the file anes-prt.bat is called by the menu program
           filenumber%=FREEFILE
           OPEN "anes-prt.bat" FOR OUTPUT AS #filenumber%
               PRINT #filenumber%, "IF EXIST printall.bat  CALL printall.bat"
               CLOSE
           IF drugmode$ = "ON" THEN
               REM write finishing lines to the drugfile
               filenumber%=FREEFILE
               OPEN drugfile$ FOR APPEND AS #filenumber%
               PRINT #filenumber%, "  ---------------------------------"
               PRINT #filenumber%, SPACE$(2); LEFT$(TIME$, 5); SPACE$(2); "END"
               PRINT #filenumber%, "\end{verbatim}"
               REM now close the file to make sure this is written
               CLOSE
               REM now make PRINTALL file print the drugs
               REM by a new command
               filenumber%=FREEFILE
               OPEN PrintAll$ FOR APPEND AS #filenumber%
               PRINT #filenumber%, "IF EXIST" + SPACE$(1) + drugfile$ + SPACE$(1)
                           + "CALL drug-11h.exe" + SPACE$(1) + timename$ + ".DRG"
           END IF
           KEY OFF
           CLOSE
           END

     CASE ELSE
          REM return to main program (LoopOne)
          COLOR green, screenbackcolour
          LOCATE 24, 68: PRINT SPACE$(12);
          PrintRecord$ = ""
   END SELECT
END IF
END SUB
```

Chapter 13

Datex AS/3 module

ch-dxmod02.tex

This is a separate program module (`a5-dxas3.pb`) containing all the SUBs for interfacing to the Datex AS/3 monitor—note that in this chapter any reference to 'Datex' is taken to relate only to the Datex AS/3 monitor. This module (`a5-dxas3.pb`) is loaded (by the Main module `a5-main.pb`) at compile time.

Each round of collecting Datex data from the computer's input buffer, checking the format, decoding the input data-string, saving the data, plotting the data to the screen display is initiated from the SUB `LoopOne` (page 133), and dealt with by initially CALLing the SUB `datexas3` (page 186). These and the CALLs to the SUBs `RequestString` and `DatexAS3` are shown schematically in Figure 5.2 (page 50).

13.1 Overview

A listing of all SUBs and FUNctions in this module and a brief description of their action is as follows:

`DatexAS3`	(p. 186) extracts Datex data from receive buffer → `ddata$`
`FixString$(bad$)`	(p. 188) removes added control codes and reconstitutes the `bad$`
`SaveAS3Data(ddata$)`	(p. 188) saves Datex data to harddrive (D-data)
`Decode(ddata$)`	(p. 189) decodes the Datex data string
`RequestString`	(p. 195) sends Transmission Request string to Datex monitor
`Short%(n1%,n2%)`	(p. 197) UNIX 2-byte Signed 'short' integer value → decimal
`Long&(n1%,n2%,n3%,n4%)`	(p. 198) UNIX 4-byte Signed 'long' integer → decimal
`Bytes2??(n1%,n2%)`	(p. 198) UNIX 2-byte Unsigned integer → decimal
`Bytes4???(n1%,n2%,n3%,n4%)`	(p. 198) UNIX 4-byte Unsigned integer → decimal
`Byte1?(n1%)`	(p. 198) UNIX 1-byte Unsigned integer → decimal
`Label??(n1%,n2%)`	(p. 198) UNIX 2-byte Unsigned integer → decimal
`Status???(n1%,n2%,n3%,n4%)`	(p. 199) UNIX 4-byte Unsigned integer → decimal

A Initiation of data output from the Datex monitor

The regular data-output (every 10 seconds) from the Datex AS/3 monitor is initiated by the Data Program main module (Chapter 8, page 82) by calling the SUB

RequestString (page 195)—see M3 in Figure 9.1—which builds and sends the so-called Transmission Request string to the Datex AS/3 monitor. This results in data being output evert 10 seconds.

```
REM Main module
...
REM now trigger data output (every 10 sec) from Datex AS/3 monitor
REM build and send the Transmission Request string.
CALL RequestString
REM  start timer and wait max 5 sec for data to arrive
thistime=timer
DO
  IF TIMER > thistime + 5 then
     PRINT
     BEEP
     PRINT " No data --- quiting program"
     SLEEP 1
     END
  END IF
  REM if data in buffer, then continue
  IF LOC(datexAS3comportfilenumber%) > 0 then
     PRINT " data output OK"
     SLEEP 1
     EXIT DO
  END IF
  SLEEP 1
  REM print dots ... while waiting
  PRINT ".";
LOOP
```

Note that each data-string output by the Datex AS/3 monitor is 321 bytes—see Chapter 3.

B Check whether data is present in the serial port receive buffer

The status of the receive buffer is checked in a loop by SUB loopone (page 133). If more than 5 characters (bytes) are in the input-buffer, then it CALLs the SUB DatexAs3 (page 186), as follows.

```
SUB LoopOne
...
IF LOC(datexAS3comportfilenumber%) > 5 OR LEN(DatexAS3Buffer$) > 5
                                                    THEN CALL DatexAS3
```

C Accessing data from receive buffer

Data in the serial port 'recieve' buffer (hardware) is moved into a temporary string datexas3buffer$ (i.e. a sort of 'software' buffer), which in PowerBasic can be up to 37K bytes. Since (a) the data flows into the receive buffer very fast, and (b), there is a 10 second interval between each new set of data from the Datex monitor, then we can just empty the buffer in a loop until there are no more characters, as follows.

```
SUB datexas3
...
REM first empty the serial port buffer
DO WHILE LOC(datexAS3comportfilenumber%) > 0
   datexAS3buffer$ = datexAS3buffer$ + INPUT$(LOC(datexAS3comportfilenumber%),
                                                #datexAS3comportfilenumber%)
LOOP
```

D Check the format of the data string

We make the following checks on the string `datexas3buffer$`.

- Since the Datex data string starts and ends with the ASCII 126 character, we check that the string extracted does not contain two ASCII 126 characters side by side. Once we have the string with a single start and stop ASCII 126 character then we rename the string → `ddata$`.
- We now check to see if the string `ddata$` contains any ASCII `<125><94>` or `<125><93>` character pairs, as these pairs are used by the Datex AS/3 monitor as ESCAPE codes in order to allow the use of the `<126>` character within the data string as well as a start/stop FLAG.

  Consequently such ESCAPE-code pairings need to be re-encoded as a single character as follows: `<125><94>` → `<126>`, and `<125><93>` → `<125>` This 'fixing' or repairing of the string is performed by the SUB `FixString$` (page 188) as follows.

  ```
  SUB datexas3
  ...
  t1$ = CHR$(125) + CHR$(94)
  t2$ = CHR$(125) + CHR$(93)
  IF INSTR(ddata$, t1$) > 0 OR INSTR(ddata$, t2$) > 0 THEN
     ddata$ = fixstring$(ddata$)
  END IF
  ...
  ```

Once the data string is 'fixed', then it is passed for decoding and processing by the SUB `decode` (page 189), as follows.

```
CALL decode(ddata$)
```

E Saving the data string `ddata$`

All the Datex data is written to the hard drive in a structured format, and is known as D-data (page 203). The data string `ddata$` is saved by the SUB `decode` (page 189) by the CALLing the SUB `SaveAS3Data` page 188) as follows (see also page 203).

```
SUB decode
...
CALL saveAS3data(ddata$)
..
```

The D-data is formated in a fixed-field block of 18 lines with 17 cols. There is a header line (date/time/name etc), and each line starts with a header code (e.g. AS3000 → AS3018), as follows.

```
AS300,09:36:49,05-03-1991,(m/d/y) Datex AS/3 monitor
AS301,126,062,001,111,005,000,000,166,052,241,058,000,000,000,000,000,000,000
AS302,000,001,000,074,255,097,220,044,000,000,000,044,000,000,000,189,189,032
...
AS317,000,013,000,002,128,002,128,002,128,001,128,000,000,000,000,014,000,002
AS318,128,002,128,002,128,001,128,000,000,000,064,081,000,222,126
```

The Datex AS/3 monitor outputs a lot of raw data, all of which must be saved for possible future use. This occupies a large amount of hard-drive space, and so system for on-the-fly compression (public domain & stable) may well be appropriate here. The D-data accumulates at approximately 200KB/hr, and the total data saved by the Anaesthesia system is approximately 500KB/hr.

F Decoding the data string

The details of the Datex AS/3 data string encoding is given in Chapter 3 (page 14).

In general, the data string is made up of many so-called 'sub-records', each of which contains a a number of numeric data (2-byte) values, as well as group of STATUS-bytes (usually 4 bytes), and a group of LABEL-bytes (usually two bytes). These status and label bytes are mostly bit-encoded, and indicate such things as the source of the particular measurement, or the existance of an error state etc—some of the encodings for the more important parameters are included in this list, but it is not comprehensive just now (see manual for full details).

For example, one of the invasive blood pressure subrecords—InvBP(1)—is decoded as follows.

```
SUB decode
...
REM  Invasive Blood Pressure BP (1) (s=systolic, d=diastolic, m = mean)
stat???=status???(62,63,64,65)
   REM   IF bit(stat???,1)=1 THEN m$="module OK"
   REM   IF bit(stat???,2)=1 THEN m$="asystole"
   REM   PRINT "status code = ";stat???;" binary ="; bin$(stat???), m$
lab??=label??(66,67)
   REM print "label code  =";lab??; "binary = ";bin$(lab??)
BP1s = short%(68,69)/100
BP1d = short%(70,71)/100
BP1m = short%(72,73)/100
HRbp1 = short%(74,75)
   REM convert to the variable name I usually use with existing ANES prog
   REM note that the final variable name is apt to be changed
   REM from time to time, with naming standards etc.
aBP1s=BP1s
aBP1m=BP1m
aBP1d=BP1d
...
```

Note that the ?? symbols associated with variables (above) relate to PowerBasic unsigned integer formats. Note also that the numeric blood pressure (1) values texttttBP1s, BP1d, BP1m (systolic, diastolic, and mean) are paired bytes in the sequence 68–75. As an example, Table 13.1 shows some particular values being decoded into the associated blood pressure values and correct units (the particular Hex values used for this example are those given in Chapter 3, page 25).

Table 13.1: Decoding invasive blood pressure 1 (bytes 62–75). The systolic, diastolic and mean blood pressure values $\times 100$ are encoded as Hex words (Unix). The decimal value therefore has to be divided by 100 to obtain the physiological value, and in this particular case the decoded values are: systolic BP 149.1, diastolic BP 73.99, mean BP 105.45. In practice we would only pass on the integer values for blood pressure. This decoding is performed by the `SUB decode` (page 189) of the Datex module.

	mean	diastolic	systolic
Byte number	73 72	71 70	69 68
Hex values	29h 31h	1Ch E7h	3Ah 3Eh
Hex word	2931h	1CE7h	3A3Eh
decimal	10545	7399	14910
BP = decimal/100	105.45	73.99	149.1

13.2 List of parameter names

This is a provisional list (`namelst.tex`) of parameter variable and string names (made in November 2001) for use in the Linux Anaesthesia AS/3 project (see also Section 3.4 (page 20) —*Output data-string format*.

Each parameter name has an associated number of Bytes, Unit, Divisor, and Description. The parameters have a fixed byte location (Position) in the output data string. Note that the first location is numbered 1, and hence this numbering corresponds exactly with the numbers used in the SUB `Decode` (Section 13.7, page 189).

The parameter names I have suggested in the following table consist mostly of combined upper-case and lower-case characters

Use of the Divisor (DIV): we first extract the numeric value, and then divide it by the DIV value to obtain the true value. (see example on Table 3.5 on page 21, for invas BP, for which DIV = 100). Use the numeric example given in brackets at the end of each line as a guide to the number of decimal places to use for the final parameter value (always either only the integer or 1 decimal place).

Note (a) that bytes 1–41 are part of the data header, (b) bytes 278–317 are not allocated at present, (c) the FirstWord (bytes 2-3) and LastWord (bytes 318–319) are always ?? the same number. (d) the checksum is an 8-bit unsigned integer of the sum of all bytes including start AND stop flags (need to check this).

Table 13.2: Parameter definitions for Datex AS/3

Position	bytes	Name	Unit	Div	Description
2–3	2	FirstWord			total bytes excluding <start> & <stop> & <checksum> (318)
4	1	stringNumber			gives the no of strings since start
5	1	AS3version			software interface version (2)
—					
42–45	4	TIMEabsolute	secs	1	Time in secs since 1/1/1970 (871550923)
46–49	4	ECGstatus	—	—	ECG status
50–51	2	ECGlabel	—	—	ECG label
52–53	2	HRecg	/min	1	ECG Heart Rate (63)
54–55	2	ST1	mm	100	ST depression 1 (1.6)
56–57	2	ST2	mm	100	ST depression 2 (1.6)
58–59	2	ST3	mm	100	ST depression 3 (1.6)
60–61	2	RRecg	/min	1	Respiratory rate (15)
62–65	4	BP1status	—	—	Inv Press 1 status
66–67	2	BP1label	—	—	Inv Press 1 label
68–69	2	BP1s	mmHg	100	Inv Press 1 systolic (120)
70–71	2	BP1d	mmHg	100	Inv Press 1 diastolic (80)
72–73	2	BP1m	mmHg	100	Inv Press 1 mean (90)
74–75	2	HRbp1	/min	1	Inv Press 1 heart rate (63)
76–79	4	BP2status	—	—	Inv Press 2 status
80–81	2	BP2label	—	—	Inv Press 2 label
82–83	2	BP2s	mmHg	100	Inv Press 2 systolic (120)
84–85	2	BP2d	mmHg	100	Inv Press 2 diastolic (80)
86–87	2	BP2m	mmHg	100	Inv Press 2 mean (90)
88–89	2	HRbp2	/min	1	Inv Press 2 heart rate (63)
90–93	4	BP3status	—	—	Inv Press 3 status
94–95	2	BP3label	—	—	Inv Press 3 label
96–97	2	BP3s	mmHg	100	Inv Press 3 systolic (120)
98–99	2	BP3d	mmHg	100	Inv Press 3 diastolic (80)
100–101	2	BP3m	mmHg	100	Inv Press 3 mean (90)
102–103	2	HRbp3	/min	1	Inv Press 3 heart rate (63)
104–107	4	BP4status	—	—	Inv Press 4 status
108–109	2	BP4label	—	—	Inv Press 4 label
110–111	2	BP4s	mmHg	100	Inv Press 4 systolic (120)
112–113	2	BP4d	mmHg	100	Inv Press 4 diastolic (80)
114–115	2	BP4m	mmHg	100	Inv Press 4 mean (90)
116–117	2	HRbp4	/min	1	Inv Press 4 heart rate (63)

Table 13.3: Parameter definitions for Datex AS/3

Position	bytes	Name	Unit	Div	Description
118–121	4	NIBPstatus	—	—	NIBP status
122–123	2	NIBPlabel	—	—	NIBP label
124–125	2	NIBPs	mmHg	100	NIBP systolic (120)
126–127	2	NIBPd	mmHg	100	NIBP diastolic (80)
128–129	2	NIBPm	mmHg	100	NIBP mean (90)
130–131	2	HRnibp	/min	1	NIBP heart rate (63)
132–135	4	TEMP1status	—	—	Temp1 status
136–137	2	TEMP1label	—	—	Temp1 label
138–139	2	TEMP1	degC	100	Temp1 (37.5)
140–143	4	TEMP2status	—	—	Temp2 status
144–145	2	TEMP2label	—	—	Temp2 label
146–147	2	TEMP2	degC	100	Temp2 (37.5)
148–151	4	TEMP3status	—	—	Temp3 status
152–153	2	TEMP3label	—	—	Temp3 label
154–155	2	TEMP3	degC	100	Temp3 (37.5)
156–159	4	TEMP4status	—	—	Temp4 status
160–161	2	TEMP4label	—	—	Temp4 label
162–163	2	TEMP4	degC	100	Temp4 (37.5)
164–167	4	SATstatus	—	—	Saturation status
168–169	2	SATlabel	—	—	Saturation label
170–171	2	SAT	%	100	Saturation (95)
172–173	2	HRsat	/min	1	Saturation heart rate
174–175	2	SATamp	%	1	Plethysmograph IR-amplitude
176–177	2	SATloc	—	100	Sat probe location (ven/art)
178–181	4	CO2status	—	—	CO2 status
182–183	2	CO2label	—	—	CO2 label
184–185	2	ETCO2	%	100	End Tidal CO2 conc (4.6)
186–187	2	FICO2	%	100	Frac Insp CO2 conc (1.3)
188–189	2	RRCO2	/min	1	Resp Rate CO2 (12)
190–191	2	AMBPCO2	mmHg	10	Ambient pressure
192–195	4	O2status	—	—	Oxygen status
196–197	2	O2label	—	—	Oxygen label
198–199	2	ETO2	%	100	End Tidal oxygen conc (37)
200–201	2	FIO2	%	100	Frac Insp oxygen conc (28)
202–205	4	N2Ostatus	—	—	N2O status
206–207	2	N2Olabel	—	—	N2O label
208–209	2	ETN2O	%	100	End Tidal N2O conc (66)
210–211	2	FIN2O	%	100	Frac Insp N2O conc (63)
212–215	4	AAstatus	—	—	Anaesthetic agent status
216–217	2	AAlabel	—	—	Anaesthetic agent label
218–219	2	ETAA	%	100	End Tidal Anaes agent (1.65)
220–221	2	FIAA	%	100	Frac Insp Anaes agent (1.65)
222–223	2	MAC	%	100	MAC total (1.25)

Table 13.4: Parameter definitions for Datex AS/3

Position	bytes	Name	Unit	Div	Description
224–227	4	FLOWVOLstatus	—	—	Flow/volume status
228–229	2	FLOWVOLlabel	—	—	Flow/volume label
230–231	2	RR	/min	1	Resp rate (12)
232–233	2	Ppeak	cmH2O	100	Peak Pressure (30)
234–235	2	PEEP	cmH2O	100	PEEP (5)
236–237	2	Pplat	cmH2O	100	Plateau pressure (12)
238–239	2	TVinsp	ml	10	Inspiratory Tidal volume (653)
240–241	2	TVexp	ml	10	Expiratory Tidal volume (653)
242–243	2	compliance	ml/cmH2O	100	compliance (23)
244–245	2	MVexp	ml	100	Inspiratory Tidal volume (653)
246–249	4	COstatus	—	—	Cardiac output status
250–251	4	COlabel	—	—	Cardiac output label
252–253	2	CO	ml/min	1	cardiac output (5000)
254–255	2	BLOODtemp	degC	100	Blood temperature (37.4)
256–257	2	EJfrac	%	1	Right heart ejection fraction (54)
258–259	2	pcwp	mmHg	100	Pulm cap wedge pressure (12)
260–263	4	NMJstatus	—	—	Neuromuscular J status
264–265	4	NMJlabel	—	—	Neuromuscular J label
266–267	2	NMJt1	%	10	Train of 4 T1 (34)
268–269	2	NMJtratio	%	10	Train of 4 ratio (34)
270–271	2	NMJptc	—	—	Post tetanic count (bit encoded)
272–273	2	HRecg2	/min	1	ECG Heart Rate (63)
274–275	2	HRmax2	/min	1	ECG Heart Rate (max) (63)
276–277	2	HRmin2	/min	1	ECG Heart Rate (min) (63)
—					
318–319	2	LastWord			total bytes excluding <start> & <stop> & <checksum> (318)
320	1	<checksum>			

13.3 Main module

```
REM a5-dxas3.pb
REM used by A5-main.pb
REM --------------
REM do not use OP2000 on open com command
REM ------------
REM used in Theatre 6
REM tested using Toshiba Laptop prog as3-sim1.bas (qb)
REM test as/3
REM ------------------------
DECLARE SUB requeststring ()
DECLARE SUB Decode (ddata$)
```

```
DECLARE SUB thedata (ddata$)
DECLARE SUB serialport ()
DECLARE SUB datexAS3 ()
DECLARE SUB saveAS3data(ddata$)
DECLARE FUNCTION short%(n1%,n2%)
DECLARE FUNCTION long&(n1%,n2%,n3%,n4%)
DECLARE FUNCTION bytes2??(n1%,n2%)
DECLARE FUNCTION bytes4???(n1%,n2%,n3%,n4%)
DECLARE FUNCTION byte1?(n1%)
DECLARE FUNCTION label??(n1%,n2%)
DECLARE FUNCTION status???(n1%,n2%,n3%,n4%)
REM ----------------
REM note that the array D$() is shared with all the FUNCTIONs
SHARED datexAS3buffer$, D$()
shared  nibpBIT8%, nibpLabel$, nibpLabel??, jNIBP
```

13.4 DatexAS3

```
SUB DatexAS3
subname$ = "datex1": CALL printsubname
REM called from LOOPone in 4A-anes.bas
REM assumes that data is already flowing every 10 secs from AS/3 monitor
REM
REM subroutine to get and process the buffer
REM and to extract the data string when start and end flags are the SAME
flagchar$ = CHR$(126)
datexAS3start:
REM -------------------------
REM first empty the serial port buffer
DO WHILE LOC(datexAS3comportfilenumber%) > 0
   datexAS3buffer$ = datexAS3buffer$ + INPUT$(LOC(datexAS3comportfilenumber%),
                                                    #datexAS3comportfilenumber%)
LOOP
REM -------------------------
REM now extract the data from the software buffer  datexAS3buffer$
IF  LEN(datexAS3buffer$) > 0 THEN
   subname$ = "datex2": CALL printsubname
   REM
   REM First locate single-flag and double-flags
   d1% = INSTR(1, datexAS3buffer$, flagchar$)
   d11% = INSTR(1, datexAS3buffer$, flagchar$ + flagchar$)

   SELECT CASE d1%
         CASE 0
              REM no flag visible
              datexAS3buffer$ = ""
              GOTO datexAS3start
```

```
         CASE 1
              REM is first char  - OK
              IF d1% = d11% THEN
                  datexAS3buffer$ = RIGHT$(datexAS3buffer$,
                                                  LEN(datexAS3buffer$) - 1)
              END IF

         CASE LEN(datexAS3buffer$)
              REM is last char
              datexAS3buffer$ = ""
              GOTO datexAS3start


         CASE ELSE
              IF d1% = d11% THEN
                  datexAS3buffer$ = RIGHT$(datexAS3buffer$,
                                           LEN(datexAS3buffer$) - d1%)
              ELSE
                 REM truncate the string so # is first char
                 datexAS3buffer$ = RIGHT$(datexAS3buffer$, LEN(datexAS3buffer$)
                                                                  - d1% + 1)
              END IF
   END SELECT

subname$ = "datex3": CALL printsubname
   REM the Flag<126> is now the first char of the string
   REM now find the next flag (end flag)
   REM remove the first char (flag) if more than 2 chars
   shortbuff$ = RIGHT$(datexAS3buffer$, LEN(datexAS3buffer$) - 1)
   d2% = INSTR(1, shortbuff$, flagchar$)
   IF d2% > 0 THEN
      REM a second flag exists
      ddata$ = LEFT$(datexAS3buffer$, d2% + 1)
      datexAS3buffer$ = RIGHT$(datexAS3buffer$, LEN(datexAS3buffer$) - LEN(ddata$))
      REM now check the format of the datastring

      REM ----------------------------------------
      REM  now detect and fix any added control characters
      t1$ = CHR$(125) + CHR$(94)
      t2$ = CHR$(125) + CHR$(93)
      IF INSTR(ddata$, t1$) > 0 OR INSTR(ddata$, t2$) > 0 THEN
          REM  PRINT  "string length = "; LEN(ddata$)
          REM  PRINT  "extra control codes detected"
          ddata$ = fixstring$(ddata$)
      END IF

      REM ----------------------------------------
      REM now free to decode the datastring
      CALL decode(ddata$)
   ELSE
```

```
      REM cant find a second flag, so exit and wait until more chars in buffer
      rem was exit DO
      EXIT IF
   END IF
END IF
datexas3Lastline:
END SUB
```

13.5 FixString$ (bad$)

```
FUNCTION fixstring$ (bad$)
subname$ = "d-fixst1": CALL printsubname
REM removes added control codes and reconstitutes the bad$
REM first detect <125><93> combination and replace with <125>
DO WHILE INSTR(bad$, CHR$(125) + CHR$(93)) > 0
  k1% = INSTR(bad$, CHR$(125) + CHR$(93))
  k2% = k1% + 1
  front$ = LEFT$(bad$, (k1% - 1))
  REM insert the correct character in the middle
  middle$ = CHR$(125)
  back$ = RIGHT$(bad$, LEN(bad$) - k2%)
  bad$ = front$ + middle$ + back$
LOOP
REM second detect <125><94> combination and replace with <126>
subname$ = "d-fixst2": CALL printsubname
DO WHILE INSTR(bad$, CHR$(125) + CHR$(94)) > 0
  k1% = INSTR(bad$, CHR$(125) + CHR$(94))
  k2% = k1% + 1
  front$ = LEFT$(bad$, (k1% - 1))
  REM insert the correct character into the middle
  middle$ = CHR$(126)
  back$ = RIGHT$(bad$, LEN(bad$) - k2%)
  bad$ = front$ + middle$ + back$
LOOP
fixstring$ = bad$
END FUNCTION
```

13.6 SaveAS3Data (ddata$)

```
SUB saveAS3Data(ddata$)
subname$ = "d-saved": CALL printsubname
REM called by SUB decode()
REM works OK
REM now save the datastring in a condensed format
REM using a fixed field format  block of 18 lines

PRINT #Ddatafilenumber%, "AS300," + time$ + "," + date$ + ", (m/d/y)
                                                    Datex AS/3 monitor"
```

```
REM ----lines 1 to 17-----
FOR KK=0 to 16
  nfirst%=(KK*18 +1)
  nlast%=nfirst% + 17
  FOR k= nfirst% to nlast%
     c%=asc(d$(k))
     AS3line$ = AS3line$ + ","+ RIGHT$("000" + LTRIM$(STR$(c%)),3)
  NEXT k
  print #Ddatafilenumber%, "AS3"+ RIGHT$("00" + LTRIM$(STR$(kk+1)),2) + AS3line$
  AS3line$=""
NEXT KK
REM ----last line-----
  FOR k= 307 to 321
     c%=asc(d$(k))
     AS3line$ = AS3line$ + ","+ RIGHT$("000" + LTRIM$(STR$(c%)),3)
  NEXT k
  print #Ddatafilenumber%, "AS318" + AS3line$
END SUB
```

13.7 Decode (ddata$)

```
SUB Decode (ddata$)
subname$ = "d-decode": CALL printsubname
REM called from   DatexAS3 sub
REM ----------------------------
REM note that ddata$ is a byte encoded data string (321 bytes)
REM with groups of bytes, WORDs, DWORDs etc coding for decimal parameters
REM note that all the byte groups are decoded, but many
REM are REMed out for the moment (as not needed just now)
REM note that each parameter has a STATUS and LABEL code
REM ------------
REM first put the data chars into an array
REM this array D$() is used by the FUNCTIONS to define the bytes
REDIM D$(LEN(ddata$))
FOR jd = 1 TO LEN(ddata$)
    D$(jd)= MID$(ddata$, jd, 1)
NEXT jd
REM --------------------------
REM now save all the data to disk
CALL saveAS3Data(ddata$)
REM --------------------------
REM now decode the variables
REM ----
REM check the number of characters in the total string
stringlength%=LEN(ddata$)
REM ----
REM get total number of bytes in the string (coded in bytes 2+3)
NumOfBytes??=label??(002,003)
REM -----
```

```
REM get interface software version (byte 5)
datexSoftwareCode%=byte1?(005)
REM -----
REM get the time (secs)
DatexAS3time???=status???(08,09,10,11)
REM -------
REM use something like message$ or errorMessage$ to write to screen
REM get ECG data
  stat???=status???(46,47,48,49)
       IF bit(stat???,1)=1 THEN m$="module OK"
       IF bit(stat???,2)=1 THEN m$="asystole"
       REM   IF bit(stat???,3)=1 THEN m$="measurement off"
       REM print "status code = ";stat???;" binary ="; bin$(stat???), m$
 lab??=label??(50,51)
      REM print "label code  =";lab??; "binary = ";bin$(lab??)
 hrecg=short%(52,53)
 ST1 = short%(54,55)
 ST2 = short%(56,57)
 ST3 = short%(58,59)
 RRecg= short%(60,61)
REM ------------------------
REM  Invasive Blood Pressure BP (1) (s=systolic, d=diastolic, m = mean)
 stat???=status???(62,63,64,65)
   REM   IF bit(stat???,1)=1 THEN m$="module OK"
   REM   IF bit(stat???,2)=1 THEN m$="asystole"
   REM   PRINT "status code = ";stat???;" binary ="; bin$(stat???), m$
 lab??=label??(66,67)
   REM print "label code  =";lab??; "binary = ";bin$(lab??)
 BP1s = short%(68,69)/100
 BP1d = short%(70,71)/100
 BP1m = short%(72,73)/100
 HRbp1 = short%(74,75)
 REM convert to the variable name I usually use with existing ANES prog
     aBP1s=BP1s
     aBP1m=BP1m
     aBP1d=BP1d
REM ------------------------
REM  Invasive Blood Pressure  BP (2)
  stat???=status???(76,77,78,79)
  REM  IF bit(stat???,1)=1 THEN m$="module OK"
  REM  IF bit(stat???,2)=1 THEN m$="asystole"
  REM  print "status code = ";stat???;" binary ="; bin$(stat???), m$
 lab??=label??(80,81)
      REM print "label code  =";lab??; "binary = ";bin$(lab??)
 BP2s = short%(82,83)/100
 BP2d = short%(84,85)/100
 BP2m = short%(86,87)/100
 HRbp2 = short%(88,89)
     REM convert to variable name I usually use with existing ANES prog
  cvps=BP2s
```

```
  cvpm=BP2m
  cvpd=BP2d
REM -----------------------------------
REM  Invasive Blood Pressure BP (3)
 stat???=status???(90,91,92,93)
     REM  IF bit(stat???,1)=1 THEN m$="module OK"
     REM  IF bit(stat???,2)=1 THEN m$="asystole"
     REM  PRINT "status code = ";stat???;" binary ="; bin$(stat???), m$
 lab??=label??(94,95)
     REM PRINT "label code  =";lab??; "binary = ";bin$(lab??)
 BP3s = short%(96,97)/100
 BP3d = short%(98,99)/100
 BP3m = short%(100,101)/100
 HRbp3 = short%(102,103)
REM -----------------------------------
REM  NIBP   Noninvasive Blood Pressure
 stat???=status???(118,119,120,121)
     REM   IF bit(stat???,1)=1 THEN m$="module OK"
     REM   IF bit(stat???,2)=1 THEN m$="asystole"
     REM  PRINT "status code = ";stat???;" binary ="; bin$(stat???), m$
 lab??=label??(122,123)
 nibpLabel??=lab??
     REM print "label code  =";lab??; "binary = ";bin$(lab??)
 nibpBIT8%=BIT(lab??,8)
 nibpBIT5%=BIT(lab??,5)
 nibpLabel$=BIN$(lab??)

 nibpS  = short%(124,125)/100
 nibpD  = short%(126,127)/100
 nibpM  = short%(128,129)/100
 hrNIBP = short%(130,131)

REM ---------------------------------------
REM  oximetery
  stat???=status???(164,165,166,167)
     REM    IF bit(stat???,1)=1 THEN m$="module OK"
     REM    IF bit(stat???,2)=1 THEN m$="asystole"
     REM    PRINT "status code = ";stat???;" binary ="; bin$(stat???)
 lab??=label??(168,169)
     REM   PRINT "label code  =";lab??; "binary = ";bin$(lab??)
 sat = short%(170,171)/100
 HRoxim = short%(172,173)
     REM amplitude
 oximetrySourceCode% = short%(176,177)
    REM oximetrySourceCode% gives the source (HR, ECG, BP) of the oximetry signal
REM -----------------------------------
REM  Carbon Dioxide
 stat???=status???(178,179,180,181)
    REM   IF bit(stat???,0)=1 THEN m$="module exists"
    REM   IF bit(stat???,1)=1 THEN m$="module active"
```

```
    REM  PRINT "status code = ";stat???;" binary ="; bin$(stat???)
 lab??=label??(182,183)
    REM PRINT "label code  =";lab??; "binary = ";bin$(lab??)
 ETCo2 =short%(184,185)/100
 FICo2 =short%(186,187)/100
 RRco2 =short%(188,189)
 Pambient = short%(190,191)/10
REM ---------------------------------------
REM   Oxygen
  stat???=status???(192,193,194,195)
     REM    IF bit(stat???,0)=1 THEN m$="module exists"
     REM    IF bit(stat???,1)=1 THEN m$="module active"
     REM PRINT "status code = ";stat???;" binary ="; bin$(stat???)
     REM label code not used here
 ETo2 = short%(198,199)/100
 FIo2 = short%(200,201)/100
REM ------------------------------------------
REM Nitrous oxide
  stat???=status???(202,203,204,205)
     REM    IF bit(stat???,0)=1 THEN m$="module exists"
     REM    IF bit(stat???,1)=1 THEN m$="module active"
     REM PRINT "status code = ";stat???;" binary ="; bin$(stat???)
     REM label code not used
 ETn2o = short%(208,209)/100
 FIn2o = short%(210,211)/100
REM ------------------------------------------------
REM  anaesthesia agent
  stat???=status???(212,213,214,215)
  IF bit(stat???,1)= 1 THEN aa$="OK"
  IF bit(stat???,2)= 1 THEN aa$="calibrating"
  IF bit(stat???,3)= 1 THEN aa$="measurement off"
       REM PRINT "status code = ";stat???;" binary ="; bin$(stat???), m$
 lab??=label??(216,217)
      IF lab??=1 THEN vapourcode$ = "NONE"
      IF lab??=2 THEN vapourcode$ = "HAL"
      IF lab??=3 THEN vapourcode$ = "ENF"
      IF lab??=4 THEN vapourcode$ = "ISO"
      IF lab??=5 THEN vapourcode$ = "DES"
      IF lab??=6 THEN vapourcode$ = "SEV"
 vapourout = short%(218,219)/100
      IF vapourout < 0 THEN vapourout = 0
 vapourin = short%(220,221)/100
      IF vapourin < 0 THEN vapourin = 0
 MAC40 = short%(222,223)/100
REM ---------------------------------------
REM  Gas Flow & volumes
   stat???=status???(224,225,226,227)
   REM    IF bit(stat???,0)=1 THEN m$="module exists"
   REM    IF bit(stat???,1)=1 THEN m$="module active"
   REM    PRINT "status code = ";stat???;" binary ="; bin$(stat???)
```

```
   REM label code not used

 RRflow = short%(230,231)
 Ppeak  = short%(232,233)/100
 peep   = short%(234,235)/100
 Pplateau = short%(236,237)/100
 TVinsp = short%(238,239)/10
 TVexp  = short%(240,241)/10
 compliance = short%(242,243)/100
 MVexp = short%(244,245)/100
REM --------------------------------
REM  redefine some variables
     rr=rrco2

REM use the integer INT() versions for some variables
REM as the decimal part not necessary (for printing to G01 file)
     TVexp = INT(Tvexp)
     sat   = INT(sat)
     Fio2  = INT(Fio2)
     Fin2o = INT(Fin2o)

REM ----------------
             REM -------TESTING------------------
             REM generate some artificial values for testing program
             REM   cvp=5:cvp2s=7:cvp2d=4 :REM need s>d to get plotted
             REM   abp1s = 105 + rnd(5,70) : abp1d = 60 + rnd(1,30)
             REM   fio2=40 +rnd(1,10)
             REM    locate 24,40:print space$(15)
             REM locate 24,40:print "vap out=";vapourout

REM ------------------------------------------------------------
REM now call the PLOT subs

REM ------------this nibp routine needs some more research!!---------
REM first sort out the nibp logic   (works OK)
REM nibp is only plotted once in the NOW window (stays displayed for 1 min)
REM this is a complicated routine, because of the way & sequence
REM that the bits are output from the Datex AS/3 monitor
REM note there is a ? BUG here as PC occasionally crashes here
REM ?? due to a strange bit sequence output under certain circumstances

    IF nibpBIT5%=1 THEN
         REM we are starting to measure the BP
         REM therefore set nibp Flag to OLD
         nibp$="old"
    END IF

REM continue to show NIBP in the NOW window for first minute only
    IF nibpBIT8%=0 AND nibps > 0 THEN
           REM means that the NIBP has been taken and we have a value
```

```
           CALL plotNIBPdatanow
           REM get the J value (current screen time secs) for the NIBP
          jNIBP=j
    END IF

    IF nibpBIT8%=1 and (j - jNIBP) > 10 THEN
            REM bit --> 1 means that it has been 1 min since measurement
            REM therefore clear the nibp bar from the NOW window
            CALL clearNIBPnow
    END IF

    IF nibpBIT8%=0 AND nibp$="old" then
          REM this must be the first *new/next* nibp reading
          REM so plot it in the NOW window
            CALL plotNIBPdatanow
            CALL plotnibp
          REM ----------
          REM rename last abp1* values to be old
            oldabp1s=abp1s
            oldabp1d=abp1d
          REM  make apb1s & d <-- nibp (as only abp1* is plotted to main screen)
            abp1s=nibps
            abp1d=nibpd
          REM now save the nibp data to the GOn and lasthour.dat files
          REM ? need to make a separate saveGNUnibp sub just for nibp
            CALL saveGNUdata
            IF windowtime$="lasthour" THEN CALL saveLastHourData
          REM reset BP values (to stop nibp being shown as red)
             abp1s=oldabp1s
             abp1d=oldabp1d
          REM -----------
          REM set the nibp Flag to NEW
             NIBP$="new"
    END IF

REM ------plot data to the NOW window-----------
    CALL plotBPdatanow
    CALL plotOximDataNow

REM ------plot data to main BP window-----------------
REM plot fio2 and SAT *after* BP
  CALL plotBP       :REM artBP, CVP, HRecg,HRoxim
  CALL plotfio2     :REM fio2
  CALL plotsat

REM ---plot the CO2 to the  CO2 window----
  CALL plotco2

REM -----TVol/RR window -----
REM plot the TV *before* the RR
```

```
  CALL plotTidalVol
  CALL plotRR   :REM use the RRco2 for RR

REM -------Vapour window------------
  CALL plotvapour

REM --------numbers to screen------------
  CALL printDataToScreen

REM ------------------------

GOTO decodeLastline

REM ------------TESTING-----------------------------------------
 CLS
locate 1,3: print "Datex Software code = ";DatexSoftwareCode%
locate 2,3: print time$  ,"string length = ";stringlength%, "Num of bytes = ";
                                                                  NumOfBytes??
locate 3,3:print "Datextime = ";DatexAS3time???
locate 4,3: print "HRecg = ";hrecg, "HRnibp = ";hrnibp, "HRoxim = ";hroxim
locate 5,3:print "nibpS = ";nibps, "nibpD = ";nibpd,

locate 7,3: print "sat = ";sat
locate 8,3: print "FIo2 = ";fio2, "ETo2 = ";eto2
locate 9,3: print "FIco2 = ";fico2, "ETco2 = ";etco2, "RRco2 = ";rrco2
locate 10,3: print "vapour = ";vapourcode$, "Vapinsp = ";vapourin, "Vapout = ";
                                                                    vapourout
locate 11,3: print "MAC40 = ";mac40, "RRecg = "rrecg
locate 12,3: print "TVinsp = ";TVinsp, "TVexp = ";tvexp, "MVol = ";MVexp
locate 13,3: print "Compliance = ";compliance, "peep = ";peep, "Ppeak = ";Ppeak
locate 14,3:print  "Pplat = ";Pplateau
locate 16,3: print "RR flow = ";RRflow, "oxim sou code = ";oximetrySourceCode%
REM ---------------------------------------------------

decodeLastline:
END SUB
```

13.8 RequestString

```
SUB requeststring
REM called by Main Module (data program)
REM sends string to the Datex AS/3 to trigger data output
REM see Chapter on Datex AS/3 monitor for full details of request string
REM -----
REM first build the request string
r0$ = CHR$(126): REM 7Eh
REM -----
r1$ = CHR$(49): REM 31h
r2$ = CHR$(0)
```

```
REM -----
r3$ = CHR$(0)
REM
r4$ = CHR$(0)
REM ------
r5$ = CHR$(0)
r6$ = CHR$(0)
REM ----
REM
r7$ = CHR$(0)
r8$ = CHR$(0)
r9$ = CHR$(0)
r10$ = CHR$(0)
REM ------
r11$ = CHR$(0)
r12$ = CHR$(0)
r13$ = CHR$(0)
r14$ = CHR$(0)
REM --------
r15$ = CHR$(0)
r16$ = CHR$(0)
REM ------
r17$ = CHR$(0)
r18$ = CHR$(0)
REM ------
r19$ = CHR$(0)
REM -----
r20$ = CHR$(0)
r21$ = CHR$(0)
r22$ = CHR$(255): REM FFh
REM -----
r23$ = CHR$(0)
r24$ = CHR$(0)
r25$ = CHR$(0)
REM ----
r26$ = CHR$(0)
r27$ = CHR$(0)
r28$ = CHR$(0)
REM -----
r29$ = CHR$(0)
r30$ = CHR$(0)
r31$ = CHR$(0)
REM -----
r32$ = CHR$(0)
r33$ = CHR$(0)
r34$ = CHR$(0)
REM ------
r35$ = CHR$(0)
r36$ = CHR$(0)
r37$ = CHR$(0)
```

```
REM ----
r38$ = CHR$(0)
r39$ = CHR$(0)
r40$ = CHR$(0)
REM -----
REM start of transmission subgroup
r41$ = CHR$(1)
REM -----
r42$ = CHR$(10): REM 0Ah
r43$ = CHR$(0)
REM -----
r44$ = CHR$(0)
r45$ = CHR$(0)
REM ----
r46$ = CHR$(0)
r47$ = CHR$(0)
REM --------
r48$ = CHR$(0)
r49$ = CHR$(0)
REM ----
r50$ = CHR$(59): REM 3Bh
r51$ = CHR$(126): REM 7Eh
REM --------
REM we assemble the string before sending it

    AA$ = r0$ + r1$ + r2$ + r3$ + r4$ + r5$ + r6$ + r7$ + r8$ + r9$
    BB$ = r10$ + r11$ + r12$ + r13$ + r14$ + r15$ + r16$ + r17$ + r18$ + r19$
    CC$ = r20$ + r21$ + r22$ + r23$ + r24$ + r25$ + r26$ + r27$ + r28$ + r29$
    DD$ = r30$ + r31$ + r32$ + r33$ + r34$ + r35$ + r36$ + r37$ + r38$ + r39$
    EE$ = r40$ + r41$ + r42$ + r43$ + r44$ + r45$ + r46$ + r47$ + r48$ + r49$
    FF$ = r50$ + r51$
    request$ = AA$ + BB$ + CC$ + DD$ + EE$ + FF$

REM print some info to the screen
PRINT
PRINT " sending REQUEST data string"
REM print to the serial port
REM do *NOT* add <CRLF> to end of string, so use the ; at the end
PRINT #1, request$;
PRINT " ......done"
PRINT
PRINT " waiting for datex reply"
PRINT
END SUB
```

13.9 Short% (n1%,n2%)

```
FUNCTION short%(n1%,n2%)
   REM n1, n2 in sequential string order
```

```
   REM returns the decimal 2-byte Signed integer value of reversed bytes
   short% = 256*asc(D$(n2%)) + asc(D$(n1%))
END FUNCTION
```

13.10 Long& (n1%,n2%,n3%,n4%)

```
FUNCTION long&(n1%,n2%,n3%,n4%)
    REM n1, n2, n3, n4 in sequential string order
    REM returns the decimal 4-bytes long Signed integer of reversed bytes
    long& = 16777216*asc(D$(n4%)) + 65536*asc(D$(n3%)) + 256*asc(D$(n2%))
                                                              + asc(D$(n1%))
END FUNCTION
```

13.11 Bytes2?? (n1%,n2%)

```
FUNCTION bytes2??(n1%,n2%)
   REM  n1, n2 in sequential string order
   REM returns the decimal 2-byte Unsigned integer value of reversed bytes
   bytes2?? = 256*asc(D$(n2%)) + asc(D$(n1%))
END FUNCTION
```

13.12 Bytes4??? (n1%,n2%,n3%,n4%)

```
FUNCTION bytes4???(n1%,n2%,n3%,n4%)
    REM n1, n2, n3, n4 in sequential string order
    REM returns the decimal 4-byte = DWORD  unsigned integer of reversed bytes
    bytes4???=16777216*asc(D$(n4%)) + 65536*asc(D$(n3%)) + 256*asc(D$(n2%))
                                                            + asc(D$(n1%))
END FUNCTION
```

13.13 Byte1? (n1%)

```
FUNCTION byte1?(n1%)
    byte1?=asc(D$(n1%))
END FUNCTION
```

13.14 Label?? (n1%,n2%)

```
FUNCTION label??(n1%,n2%)
   REM n1, n2 in sequential string order
   REM returns the decimal  2-bytes Unsigned integer of reversed bytes
   label?? =  bytes2??(n1%,n2%)
END FUNCTION
```

13.15 Status??? (n1%,n2%,n3%,n4%)

```
FUNCTION status???(n1%,n2%,n3%,n4%)
    REM n1, n2, n3, n4 in sequential string order
    REM returns the decimal 4-bytes Unsigned integer of reversed bytes
    status??? =  bytes4???(n1%,n2%,n3%,n4%)
END FUNCTION
```

Part V

Data storage

Chapter 14

Data storage, files and formats

ch-data.tex

14.1 Introduction

The Anaesthesia Record System (ARS) generates four separate groups of stored numeric data, known as D-data, G-data, drug-data, lasthour-data. In addition, two output-files are used for storing information required for coordinating the printing process (`printall.bat` and `anes-prt.bat`; and another output-file is used for storing error messages (`<filename>.err`). Each data-file has a time/date-encoded filename plus a data-specific file-extension, e.g. `91503835.G01`.

Numeric data is written to the various output-files on the hard-drive every 5 mins (default), and also whenever a NIBP measurement is processed (this is because NIBP measurements tend to be of a variable frequency—but usually 1–5 mins when used). Other files are written to as necessary. Pressing either the F1 or F10-key terminates the Data Program and closes all open files as part of the shutdown routine.

14.2 `Printall.bat`

This file is OPENed by the SUB `OpenFiles` (page 138) in order to write a single line giving the command which wil be used during the printing process to print the new G-data filename, e.g. something like `plotan3a.exe 00918K15.G01`. The file is then closed, as follows:

```
SUB openFiles
...
PrintAll$ = "c:PrintAll.bat"
PrintAllfilenumber% = FREEFILE
OPEN PrintAll$ FOR OUTPUT AS #PrintAllfilenumber%
PRINT #PrintAllfilenumber%, "plotan3a.exe" + SPACE$(1) + Gdatafilename$
CLOSE #PrintAllfilenumber%
REM  SUBs which write to this file:- PrintNewScreen sub, terminate sub
...
```

Subsequently at the beginning of each hour when the new screen is built the SUB `PrintNewScreen` (page 160) writes a new line to the file, once more indicating the next G-data filename.

Finally, when either F1 or F10 is eventually pressed (to terminate the Data Program), a final command (for processing the drug-file) using `drug-11h.pb` (page 250) is written to the file, before the Data Program terminates altogether, as follows.

```
SUB terminate
...
IF drugmode$ = "ON" THEN
   ...
   REM now enable the PRINTALL file to print the drugs
   filenumber% = FREEFILE
   OPEN PrintAll$ FOR APPEND AS #filenumber%
   PRINT #PrintAllfilenumber%, "IF EXIST" + SPACE$(1) + drugfile$  + SPACE$(1)
                            + "CALL drug-11h.exe" + SPACE$(1) + timename$ + ".DRG"
   CLOSE
END IF
```

Consequently, the final state of the file `printall.bat` may look something like the following.

```
plotan3a.exe 00918K15.g01
plotan3a.exe 00918K15.g02
plotan3a.exe 00918K15.g03
plotan3a.exe 00918K15.g04
drug-11h.exe 00918K15.drg
```

Each line in the above batch file `printall.bat` except the last, is a command to process the G-data file `00918K15.Gnn` using the program `plotan3a.exe` (Chapter 17, page 217). CALLing this batch file therefore coordinates the whole process of printing all the G-data files for an operation.

At present the file `printall.bat` is only used once at the end of an anaesthetic to coordinate the printing of the Anaesthesia Record, and it is overwritten during the next anaesthetic. However, it would actually be more useful to save the batch file in order to facilitate printing of the data of each anaesthetic, by simply renaming it so it has the same operation-specific date/time-encoded filename as the contained data files, e.g. something like `00918K15.bat`. The could be done from the outset, or manipulated at the end by the SUB `Terminate`.

14.3 `Anes-prt.bat`

This file is created by the SUB `terminate` (page 175) immediately prior to shutdown, as follows.

```
SUB terminate
...
OPEN "anes-prt.bat" FOR OUTPUT AS #filenumber%
PRINT #filenumber%, "IF EXIST printall.bat  CALL printall.bat"
CLOSE
```

The file `anes-prt.bat` therefore contains only the single line

```
IF EXIST printall.bat  CALL printall.bat
```

When the file `anes-prt.bat` is subsequently CALLed by the menu front-end program (Chapter 6, page 53)—since the Data Program has now terminated—it therefore initiates the printing process.

14.4 Filenames—time/date encoding

The data filename consists of 8 characters in a time/date encoded sequence [year][month][day][hour][minutes] and is created by SUB `OpenFiles` (page 138) as follows.

- Year: last two digits of the year (00–99)
- Month: one alpha-numeric character (0–9, A–C)
- Day: two digits (00–31)
- Hour: one alpha-numeric character (0–9, A–N)
- Minutes: two digits (00–59)

For example, the filename for a record started at 14.34 hrs on 26 November 2001 would be 01B26E34; i.e. 01 B 26 E 34 The filename is defined in the SUB `openfile`.

14.5 D-data.

This is the raw data from the Datex AS/3 monitor, and is written to the file by SUB `decode` (page 189). Each data record consistes of 321 bytes, and is saved as a comma separated string of 8-bit ASCII codes (000–255). The data is written to files having a `.Dnn` file extension, e.g. `D01, D02` etc. Each file contains one hour of data.

The format of the D-data is as follows (note that each 10 second data episode of 19 lines is indicated by the code sequence `AS3000` → `AS3018` at the beginning of each line).

```
....
....
AS316,141,067,000,067,000,067,000,000,000,000,000,189,189,001,128,000,000,000
AS317,000,013,000,002,128,002,128,002,128,001,128,000,000,000,000,014,000,002
AS318,128,002,128,002,128,001,128,000,000,000,064,081,000,076,126
AS300,09:36:49,05-03-1991,(m/d/y) Datex AS/3 monitor
AS301,126,062,001,111,005,000,000,166,052,241,058,000,000,000,000,000,000,000
AS302,000,001,000,074,255,097,220,044,000,000,000,044,000,000,000,189,189,032
AS303,000,189,189,032,000,166,052,241,058,019,048,000,000,000,034,067,000,021
AS304,000,001,128,001,128,001,128,003,000,000,000,001,000,062,058,231,028,049
AS305,041,067,000,003,000,000,000,002,000,247,008,244,005,044,007,067,000,000
AS306,000,000,000,011,000,002,128,002,128,002,128,001,128,000,000,000,000,003
AS307,000,002,128,002,128,002,128,001,128,003,000,000,000,003,001,001,128,001
AS308,128,001,128,001,128,003,000,000,000,011,000,210,013,003,000,000,000,012
```

```
AS309,000,004,128,000,000,000,000,013,000,001,128,000,000,000,000,014,000,001
AS310,128,003,000,000,000,000,000,222,038,068,000,108,000,001,128,003,000,000
AS311,000,009,000,138,001,000,000,012,000,102,029,003,000,000,000,000,000,113
AS312,014,165,015,003,000,000,000,000,000,007,023,241,022,003,000,000,000,004
AS313,000,000,000,000,000,058,000,003,000,000,000,000,000,012,000,010,015,008
AS314,002,192,013,130,022,229,020,244,006,126,002,000,000,000,000,007,000,001
AS315,128,001,128,001,128,001,128,032,000,000,000,000,000,001,128,001,128,255
AS316,141,001,128,067,000,066,000,000,000,000,000,189,189,001,128,000,000,000
AS317,000,013,000,002,128,002,128,002,128,001,128,000,000,000,000,014,000,002
AS318,128,002,128,002,128,001,128,000,000,000,064,081,000,222,126
AS300,09:36:52,05-03-1991,(m/d/y) Datex AS/3 monitor
AS301,126,062,001,112,005,000,000,176,052,241,058,000,000,000,000,000,000,000
AS302,000,001,000,074,255,097,220,044,000,000,000,044,000,000,000,189,189,032
....
....
```

14.6 G-data

G-data is a sampled subset of D-data, and is the data collected (every 40 seconds approximately) for printing the anaesthetic record. G-data is written to the file by the SUB `SaveGNUdata` (page 163). This data is written to files having a `.Gnn` file extension, e.g. `G01, G02` etc. G-data is sampled every 40 seconds, and each `.gnn` file contains one hour of data.

Each data record consists of 15 physiological parameters, and two time-stamps (seconds since midnight, and seconds since start of current hour), and is saved as a $<$ *SPACE* $>$ separated fixed-field decimal values as strings, as follows.

```
SUB saveGnudata
...
gnu1$ = TIM$ + timej$ + hrecg$ + hroxim$ + abp1s$ + abp1d$
gnu2$ = sat$ + fio2$ + etco2$ + fico2$ + tvexp$ + rr$
gnu3$ = vapourin$ + vapourout$ + cvp$ + bmac$ + fin2o$
PRINT #Gdatafilenumber%, gnu1$ + gnu2$ + gnu3$
...
```

From left-to-right the data are as follows:
(1) `TIM$`; time in seconds from midnight
(2) `timej$`; time in seconds since start of current hour
(3) `hrecg$`; HR (ecg)
(4) `hroxim$`; HR (oximeter)
(5) `abp1s$`; systolic BP (1)
(6) `abp1d$`; diastolic BP (1)
(7) `sat$`; saturation
(8) `fio2$`; Inspired O2 fraction
(9) `etco2$`; expired CO2 %
(10) `fico2$`; Inspired CO2 %
(11) `tvexp$`; Expired Tidal Volume
(12) `rr$`; Respiratory rate
(13) `vapourin$`; inspired vapour conc %
(14) `vapourout$`; expired vapour conc %

(15) cvp$; Central venous pressure
(16) bmac$; total MAC
(17) fin2o$; inspired N2O %

The format of the G-data is as follows.

```
from 91503835.G02
034610 000092 067 068 149 073 099 040 3.94 -100 0534 12 ----- ----- 0018 0.87 058
034664 000146 065 067 148 072 099 040 3.96 -100 0520 12 ----- ----- 0017 0.87 058
034704 000186 068 068 150 075 099 040 0004 -100 0509 12 00.88 00.54 0018 1.05 058
034746 000228 068 070 150 075 099 039 3.99 -100 0506 12 00.94 00.64 0018 1.12 058
034787 000269 067 066 148 074 099 039 4.01 -100 0531 12 00.96 00.66 0017 1.13 058
034827 000309 064 064 146 072 099 039 4.04 -100 0532 12 00.98 00.68 0018 1.15 059
034867 000349 064 064 140 068 099 039 4.02 -100 0542 12 00.99 00.71 0014 1.17 058
034908 000390 067 065 149 077 099 039 0004 -100 0535 12 00.99 00.72 0020 1.17 059
034948 000430 067 068 148 075 099 039 4.01 -100 0548 12 01.01 00.76 0019 01.2 059
034988 000470 070 071 151 078 099 039 3.99 -100 0541 12 01.02 00.76 0019 01.2 059
035028 000510 071 071 155 081 099 039 0004 -100 0542 12 01.03 00.77 0019 1.21 058
035068 000550 069 069 153 079 099 039 3.97 -100 0535 12 01.03 00.81 0019 1.23 058
035108 000590 069 068 155 078 099 039 3.93 -100 0543 12 01.03 000.8 0019 1.23 058
035148 000630 069 070 156 080 099 039 0004 -100 0544 12 01.04 00.94 0019 1.32 058
035188 000670 069 070 159 081 099 039 0004 -100 0541 12 01.05 00.83 0019 1.25 058
035228 000710 069 067 158 080 099 039 3.89 -100 0532 12 01.05 00.84 0021 1.25 058
035268 000750 068 068 157 080 099 039 3.52 -100 0507 12 01.05 00.88 0020 1.28 058
035309 000791 068 068 157 081 099 039 3.72 -100 0514 12 01.05 00.87 0020 1.27 058
035350 000832 068 069 160 082 099 039 3.93 -100 0520 12 01.05 00.85 0020 1.26 058
035391 000873 068 068 159 082 099 039 03.9 -100 0521 12 01.05 00.85 0020 1.26 058
035431 000913 068 068 158 081 099 039 3.96 -100 0525 12 01.05 00.86 0020 1.27 058
035472 000954 068 068 159 082 099 039 4.05 -100 0513 12 01.06 00.85 0020 1.26 058
035512 000994 068 067 159 082 099 039 3.94 -100 0519 12 01.06 00.88 0020 1.28 058
035552 001034 068 068 159 082 099 039 3.84 -100 0517 12 01.06 00.88 0021 1.28 058
035592 001074 068 068 157 079 099 039 3.76 -100 0518 12 01.05 00.89 0021 1.29 058
035632 001114 068 067 159 081 099 039 3.95 -100 0521 12 01.06 00.87 0021 1.28 058
035672 001154 068 068 159 081 099 039 3.98 -100 0521 12 01.06 00.87 0021 1.27 059
035713 001195 068 068 160 083 099 039 04.1 -100 0529 12 01.06 00.88 0021 1.28 058
035754 001236 068 069 161 083 099 039 3.84 -100 0513 12 01.06 00.89 0021 1.29 059
035797 001279 069 070 163 084 099 039 03.8 -100 0514 12 01.06 00.87 0021 1.27 058
035837 001319 068 068 163 084 099 039 4.03 -100 0529 12 01.06 00.87 0021 1.28 058
035878 001360 069 069 163 083 099 039 3.96 -100 0523 12 01.06 00.86 0021 1.27 058
035918 001400 070 069 165 084 099 039 3.89 -100 0516 12 01.06 00.88 0022 1.28 058
035959 001441 070 071 163 084 099 039 3.98 -100 0520 12 01.06 00.88 0021 1.28 058
036014 001496 070 070 165 085 099 039 3.94 -100 0515 12 01.06 000.9 0020 1.29 058
036054 001536 070 070 165 084 099 039 3.73 -100 0514 12 01.06 00.94 0020 1.32 057
036097 001579 068 068 161 082 099 039 3.55 -100 0507 12 01.07 00.92 0020 1.31 058
036138 001620 068 068 160 081 099 039 3.74 -100 0515 12 01.07 00.91 0020 01.3 058
036178 001660 068 068 158 079 099 039 3.85 -100 0516 11 01.06 000.9 0019 1.29 058
036218 001700 068 070 158 079 099 039 3.78 -100 0528 10 01.06 000.9 0018 1.29 058
036258 001740 069 069 155 077 099 039 3.86 -100 0527 10 01.07 00.91 0018 01.3 058
036299 001781 069 069 154 076 099 039 3.83 -100 0518 10 01.07 00.88 0018 1.28 058
....
....
```

14.7 Drug-data

This is all the data input via the keyboard, and is written to an plain text 'drug-file' having a `.drg` file-extension. Since this file is destined to be typeset by being printed 'in verbatim' by LaTeX 2ε then the `\begin{verbatim}` and `\end{verbatim}` need to be written exactly as shown below. The heading is written by SUB `OpenFiles` (page 138) when the file is originally created. The footer is written by SUB `terminate` (page 175) on closing the Data Program after pressing either F1 or F10.

The format of the drug datafile is as follows.

```
\begin{verbatim}
  Drug filename = 91503835.DRG
  05-03-1991  Code: 91503835.G01
  08:35  START
  ---------------------------------
  09:35  Remifent 50 mcg
  09:35  Etomidate (mg) 15 mg
  09:35  Atracurium (mg) 50
  09:35  Augmentin 1.2
  09:35  Atracurium (mg) 20
  10:00  Atracurium (mg) 20
  10:18  epid marcain .25% plain
  10:18  epidural 5 + 2 mls
  10:40  Atracurium (mg) 20
  10:41  Total urine (mls)  200
  11:12  Atracurium (mg) 10
  11:23  Temp = 35.2 C (Naso/ph)
  11:27  CVP = 15
  11:52  Atracurium (mg) 10
  11:56  CVP = 9
  11:59  epid + 7mls
  12:13  epid   + 2mls
  12:14  Atracurium (mg) 10
  12:29  Total urine (mls)  200
  12:32  epid + 3mls
  12:36  epid + 2.5mls
  12:48  CVP =14
  12:49  epid =2mls
  12:50  Atracurium (mg) 10
  13:12  epid = 2mls
  13:13  Atracurium (mg) 10
  ---------------------------------
  13:35  END
\end{verbatim}
```

14.8 Lasthour-data

During he last 20 minutes of each hour all data displayed on the screen is saved to the file `lasthour.dat` by SUB `savelastHourData` (page 165), in order for this data to

be read back and displayed as the first 20 minutes of the next screen display (the screen displays 1 hr 20 mins of data, but updates each hour).

The file consists of 17 data fields, of which the first is the time in seconds since the start of the 20 minute period. The sequence from left-to-right is indicated in the program extract below (the names correspond to the above G-data).

```
SUB saveLastHourData
...
REM print data to LastHourDataFile (opened in LoopTwo/?hkeeping)
REM save in same order as in SaveGNUdata sub
a$ = timej$ + nibps$ + nibpd$ + hrecg$ + hroxim$ + abp1s$ + abp1d$
B$ = sat$ + fio2$ + etco2$ + fico2$ + tvexp$
C$ = rr$ + vapourin$ + vapourout$ + cvp$ + bmac$
PRINT #LastHourDataFilenumber%, a$ + B$ + C$
END SUB
```

The format of the Lasthour datafile is as follows.

```
003613 163 087 069 068 144 068 098 041 043 000 0556 10 119 087 -328 1.55
003623 163 087 068 068 145 068 098 041 043 000 0561 10 119 087 -328 1.55
003633 163 087 069 068 143 068 098 041 043 000 0554 10 100 086 -328 1.53
003643 163 087 069 069 144 069 098 041 043 000 0549 10 119 087 -100 1.53
003653 163 087 069 069 144 068 098 041 044 000 0556 10 117 087 -328 1.54
003664 163 087 069 069 142 067 098 041 044 000 0552 10 119 087 -328 1.55
003675 163 087 068 069 142 067 098 041 044 000 0553 10 119 086 -328 1.53
003685 163 087 068 068 140 066 099 041 044 000 0559 10 119 088 -100 1.55
003695 163 087 069 069 140 067 099 041 044 000 0555 10 119 088 -328 1.56
003706 163 087 069 069 137 065 099 041 044 000 0550 10 119 087 -328 1.55
.....
.....
.....
004639 139 103 088 066 107 054 099 040 043 000 0566 10 117 088 -328 1.57
004650 139 103 065 065 108 054 099 040 043 000 0566 10 119 087 -328 1.55
004660 139 103 064 065 103 051 099 040 043 000 0573 10 117 087 -328 1.56
004670 139 103 063 064 101 051 099 040 044 000 0563 10 117 087 -328 1.55
004680 139 103 063 063 100 051 098 040 043 000 0572 10 117 089 -328 1.58
004690 139 103 063 063 098 050 099 040 043 000 0565 10 119 088 -328 1.57
004700 139 103 064 064 100 051 099 040 044 000 0572 10 119 088 -328 1.57
004710 139 103 064 064 100 051 099 040 043 000 0572 10 119 088 -328 1.56
```

This data is accessed and printed to the screen by SUB `printLast20minsFast` (page 159).

Part VI

The printer program

Chapter 15

Printing—an overview

ch-print.tex

15.1 Starting the printing process

When F10 is pressed to quit the Data Program the SUB `terminate` (page 175) creates a one-line batch file called `anes-prt.bat` (page 202) which holds a pointer to a second batch file (`printall.bat`, page 201) which would be similar to the following:

```
REM printall.bat
plotan3a.exe 00918K15.g01
plotan3a.exe 00918K15.g02
plotan3a.exe 00918K15.g03
plotan3a.exe 00918K15.g04
drug-11h.exe 00918K15.drg
```

Each line in the above batch file `printall.bat` except the last, is a command to process the G-data file `00918K15.Gnn` using the program `plotan3a.exe` (Chapter 17, page 217). CALLing this batch file therefore coordinates the whole process of printing all the G-data files for an operation. The final line is a command to process the drug-data file `00918K15.drg` using the program `drug-11h.exe` (page 250). Thus the above example `printall.bat` (written during the anaesthetic) contains all the information to produce the final Anaesthetic Record.

Once the Data Program has been quit (F10), then command returns to the Menu program (see above) which now looks to see if the batch file `anes-prt.bat` exists, and if so, then it CALLs this and starts the printing process. We will now look in turn at (a) the printing of the graphic trend data—coordinated by the program `plotan3a.exe` (page 217) , and (b) the printing of the drug-data—coordinated by the program `drug-11h.exe` (page 250).

15.2 Processing the graphic trend data (`plotan3a.pb`)

We shall now see how the first line of the above `printall.bat` batch file is processed by the program `plotan3a.exe` (Chapter 17, page 217). A typical graphic page from the Anaesthetic Record is shown in Figure 2.1 (page 13).

```
plotan3a.exe 00918K15.g01
```

A Accessing the filename

As the filename is the second parameter, it can therefore be accessed using the PowerBasic `Command$` variable, and we grab the whole filename (and extension) placing it into the filename `Gnudatafile$` as follows.

```
Gnudatafile$ = COMMAND$
```

B Get start time from the .Gnn file

This integer value is required in order to build the appropriate time scale (horizontal axis). Note that the start time (time since midnight in seconds) is the first value in the first row of the file, e.g. `034610` in the following example (see also page 204 for file format).

```
034610 000092 067 068 149 073 099 040 3.94 -100 0534 12 ----- ----- 0018 0.87 058
034664 000146 065 067 148 072 099 040 3.96 -100 0520 12 ----- ----- 0017 0.87 058
034704 000186 068 068 150 075 099 040 0004 -100 0509 12 00.88 00.54 0018 1.05 058
...
```

The code for grabbing this value and placing it into the integer variable `starttime#` is as follows (open the file, grab the first line, and then close the file).

```
REM plotan3a.pb
...
OPEN "c:" + Gnudatafile$ FOR INPUT AS #1
LINE INPUT #1, timeline$
CLOSE #1
REM  get the first time value from the first column
starttime# = VAL(LEFT$(timeline$, 6))
```

C Create separate time/data parameter files

Using the SUB `MakeNewFiles` (page 220) we now create a series of small separate files, one for each parameter, and write the relevant data as two vertical space-separated columns, the first one for time (seconds), and the second being the data-value (see Chapter 18 for all the other examples).

For example, the HR-oximetry data would be placed in a separate file as follows, with time on the left, and HR on the right. In this case the first line has no returned value for the HR, and so this is indicated with a `---`. In order to stop GNUplot from trying to plot points having one of the values missing (as for line 1) we therefore write a `#` symbol as the first character on this line which tells the program GNUplot not to plot this particular line, and skip to the next line (see GNUplot manual). The 337 seconds value on the last row indicates that this set of values represents approximately 5.6 minutes of data.

```
# 8           ---
 18            82
 58            82
```

```
98              82
138             83
178             83
218             83
227             83
268             83
271             83
282             83
323             84
337             84
```

D Write the GNUplot scripts for each graph

We now write a series of GNUplot scripts (one for each graph) which will plot the graphs. In practice we do this by CALLing the 6 SUBS `plotSAT` (page 226), `plotFO2` (page 225) `plotBP` (page 223) `plotVapour` (page 228) `plotTV` (page 227).

These SUBs each output the associated `.gnu` script file, e.g. `plot-bp.gnu`, examples of which are all shown in Chapter 19 (page 238). For example, a typical `plot-bp.gnu` script is as follows. Note that when these scripts are run through GNUplot they output graphs in LaTeX picture format, and use the 'latex' terminal mode. Each output file is given a `.pic` filename extension.

```
# plot-bp.gnu script output by SUB plotBP (in plotan3a.pb)
set terminal latex   # need to use 'latex' for LaTeX2e
set output "c:\emtex\texinput\plot-bp.pic"
# NB. full size = 5x3 inches:set x,y
set size 1.5,1
# starttime = 47585
# h = 14
# give deltahour (deltah) in secs from start (starttime)
deltah =  2815
q=900; qq=1800; qqq=2700; qqqq=3600
set xtics("13.00" deltah-qqqq,"13.15" deltah-qqq,"13.30" deltah-qq,
          "13.45" deltah-q,"14.00" deltah,"14.15" deltah+q,
          "14.30" deltah+qq,"14.45" deltah+qqq,"15.00" deltah-qqqq)
set ytics("0" 0,"20" 20,"50" 50,"100" 100,"150" 150,"200" 200)
set title"\hspace*{12cm} \sc{99B21D12.G01}"
set ylabel"BP sys $\bigcirc$\\BP dias $\circ$\\HR $\bullet$\\CVP {\bf ---}"-3
set nokey
set grid
xmin=0; xmax=3600
ymin=0; ymax=200
plot [xmin:xmax][ymin:ymax] 98 with lines 1,\
     20 with lines 1,\
     10 with lines 4,\
     "BPS.DAT" using 1:2 with linespoints 1 9,\
     "BPD.DAT" using 1:2 with linespoints 1 8,\
     "HRecg.DAT" using 1:2 with linespoints 1 10,\
     "HRoxim.DAT" using 1:2 with points 1 10,\
     "CVP.DAT" using 1:2 with lines 2
```

E Run GNUplot on all the `.gnu` scripts

We now SHELL out and run GNUplot on each of the `.gnu` script files. The code for this and the previous section is as follows.

```
REM plotan3a.pb
...
CALL plotSAT
CALL plotFO2
...
SHELL "gnuplot2 plot-sat.gnu"
SHELL "gnuplot2 plot-fo2.gnu"
...
```

F Typeset the graphs using LaTeX 2ε

We now typeset all the graphs by LaTeXing the LaTeX script file `prtanes2.tex` as shown in Chapter 20 (page 244). The code we use is as follows.

```
REM plotan3a.pb
...
SHELL "latex prtanes2.tex"
...
```

This results in a `.dvi` file (DeVice Independent file) called `prtanes2.dvi`, which can then be sent to the printer (assuming we have an appropriate printer driver), and will result in all the graphs being printed on a single sheet, as shown in Figure 2.1 (page 13).

G Send the `.dvi` file to the printer

We now print the graphic page by sending the `.dvi` file to the printer. I am currently using the EmTeX implementation of LaTeX 2ε which uses the printer driver `prthpdj` for a HP DeskJet 500 series printer, and requires the following code.

```
REM plotan3a.pb
...
SHELL "prthpdj prtanes2.dvi prn"
...
```

Note the equivalent command when using a HP LaserJet is `prthplj prtanes2.dvi prn`.

15.3 Processing the drug sheet (`drug-11h.pb`)

This is done simply by typesetting the `.drg` drug file 'in verbatim' using LaTeX 2ε. However, inorder to make best use of space on the page, the drug file is typeset either (a) in single column format (if few lines in the file), or (b) in a two column format if there are more lines than will fit on one page using single column.

So, we first count the number of lines, and decide accordingly—the cut off was about 46 lines. The following works well in practice (I used the 10 point tt verbatim font).

```
REM drug-11h.pb
...
REM first count the number of lines
DO WHILE NOT EOF(1)
     INPUT #1, dataline$
     n = n + 1
     REM PRINT "n= "; n, dataline$
LOOP
CLOSE 1
...
IF n - 2 < 46 THEN
     REM typeset in single col
     SHELL "latex prt1drug.tex"
     REM now print to DeskJet printer
     SHELL "prthplj /od+ prt1drug.dvi prn"
ELSE
     REM typeset using 2 columns as usual
     SHELL "latex prt2drug.tex"
     REM now print to DeskJet printer
     SHELL "prthplj /od+ prt2drug.dvi prn"
END IF
```

The two alternative `.tex` files (`prt1drg.tex`, and `prt2drg.tex`) and the program `drug-11h.pb` are all shown in Chapter 21 (page 250).

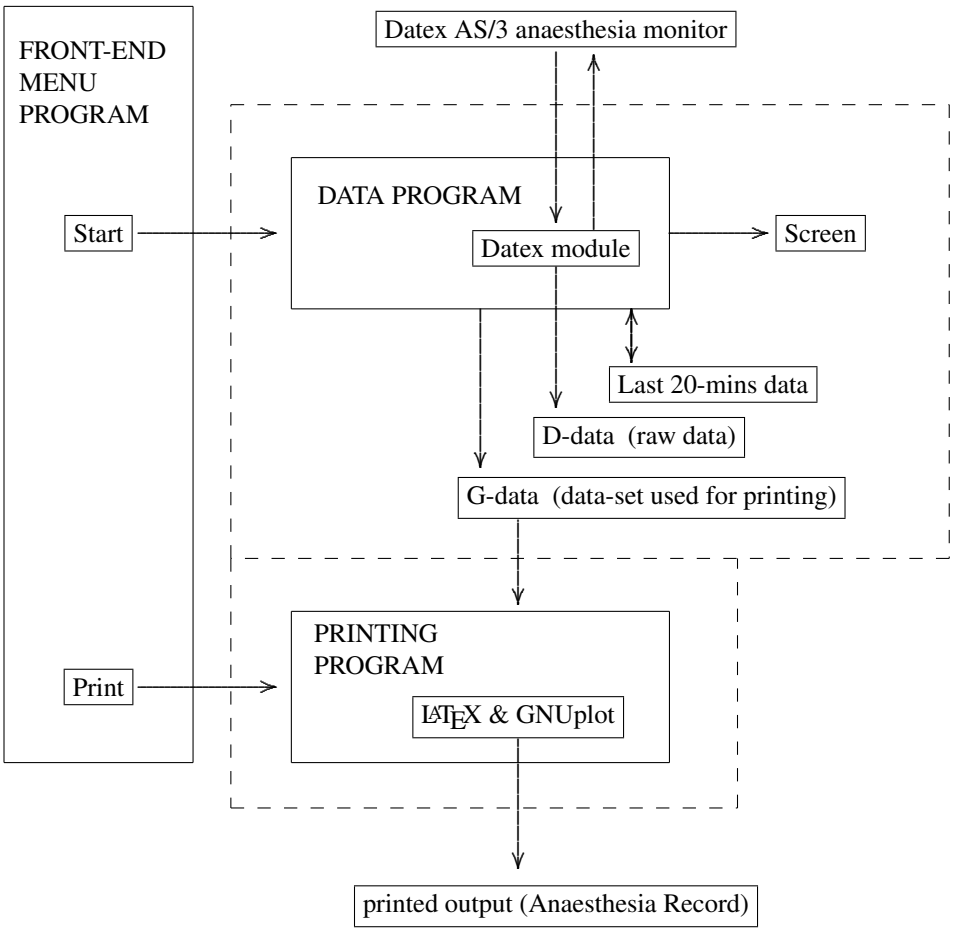


Figure 15.1: Overview of the Anaesthesia Record System programs. There are three main components (a) the front-end menu program, (b) the data program, and (c) the printing program. The data program is started by selecting from the pull-down menu. Printing proceeds either automatically after quitting the data program (by pressing F10 key), or by selecting a print option from the pull-down menu.

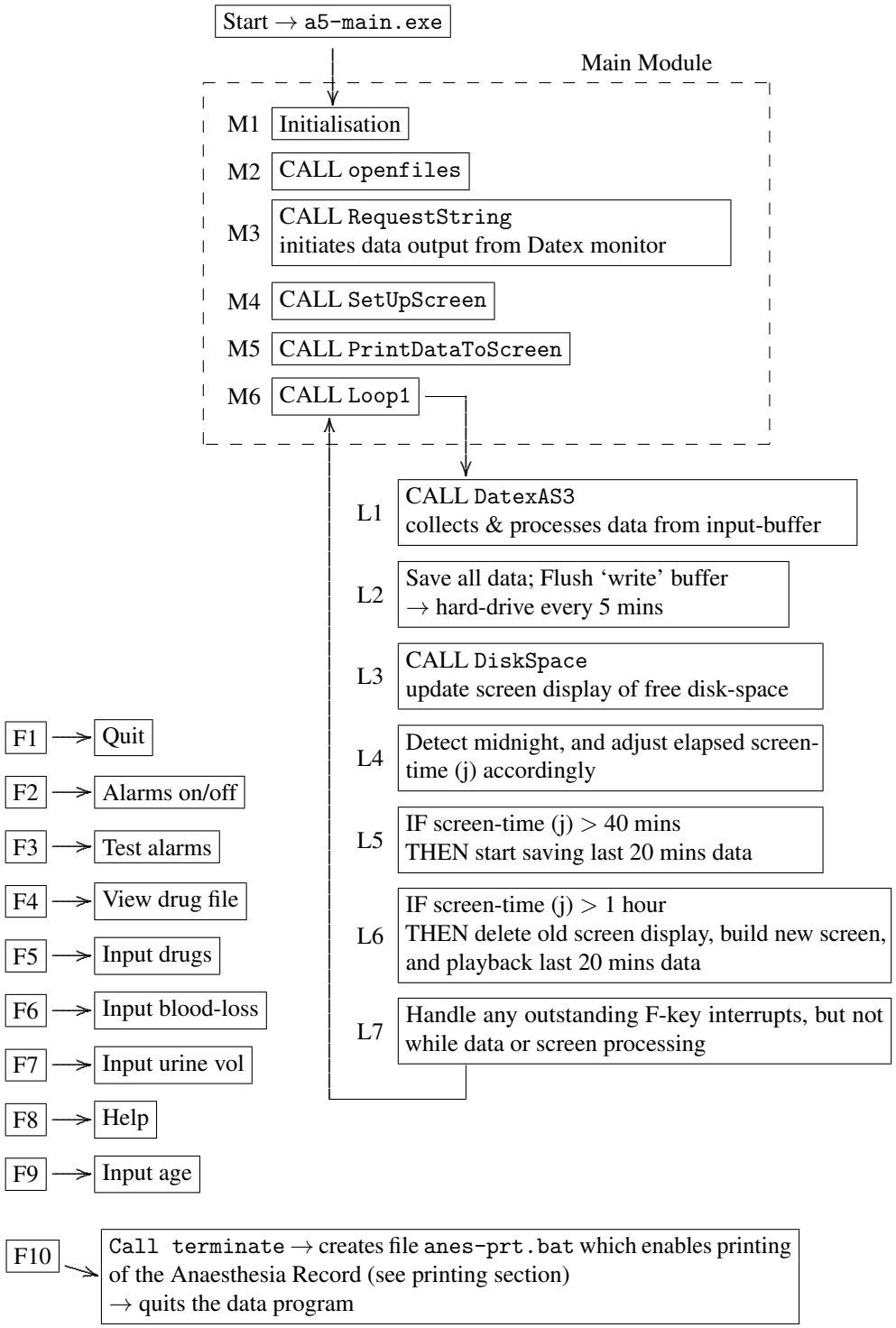


Figure 15.2: Overview of the data program. There are three components (a) the main module (M1–M6), (b) the data-collecting loop (L1–L7), and (c) F-key actions (F1–F10). The program is terminated by either pressing F1 (quit), or pressing F10 (prints out the Anaesthesia Record, and then quits).

Chapter 16

Printall.bat

ch-pbat.tex

The format of the `printall.bat` file is shown below.

```
plotan3a.exe 01522J28.G01
plotan3a.exe 01522J28.G02
plotan3a.exe 01522J28.G03
plotan3a.exe 01522J28.G04
plotan3a.exe 01522J28.G05
drug-11h.exe 01522J28.drg
```

Chapter 17

PlotAn3a.pb

ch-plot.tex

```
REM [PlotAn3a.bas]    (upgrade from plotAn8c.bas)
REM NEW version to accomodate the HRecg / HRoxim problem
REM and to add N2O
REM To be used to plot a single anaesthesia sheet from menu-3a.bat
REM  fixed the timebase problem OK
REM  uses Gnuplot2 (=3.2)
REM this produces the GNUPLOT progs [plot-bp.gnu/plot-co2.gnu etc]
REM (c) RWD Nickalls 1994-1998
REM last revised March 4 1998
REM adjusted to plot CVP (on the BP graph) and bigMAC (on vapour) graph
REM allows plotting of the NIBP by splitting the G01 file etc
REM uses COMMAND$ to access the filename
REM ------------------------------------------------------------
DECLARE SUB box1 (box$, t%, b%, s%)
DECLARE SUB box2 (box$, t%, b%, s%)
DECLARE SUB MakeNewFiles ()
DECLARE SUB letterquality ()
DECLARE SUB plotCO2 ()
DECLARE SUB plotSAT ()
DECLARE SUB plotFO2 ()
DECLARE SUB plotBP ()
DECLARE SUB plotVAPOUR ()
DECLARE SUB plotTV ()
DECLARE SUB printerstatus ()
DECLARE SUB timebase ()
COMMON SHARED starttime#, Gnudatafile$, Jnudatafile$
REM --------------------
REM blue screen with white text
COLOR 15, 1
```

```
REM ----------GNU-DATAFILE---------
REM use the command$ to pass the filename of the file to be processed
Gnudatafile$ = COMMAND$
REM    Gnudatafile$ = "97522A32.G04"
REM  Gnudatafile$ = "94TEST04.G01": REM ************************
REM get the starttime (secs) from line 1 of the GNUdatafile (.G01 etc)
OPEN "c:" + Gnudatafile$ FOR INPUT AS #1
LINE INPUT #1, timeline$
CLOSE #1
REM  get the first time value from the first column
starttime# = VAL(LEFT$(timeline$, 6))
REM ----now plot the data------------
CLS
PRINT
LOCATE 5: a$ = "creating graphic files": CALL box1(a$, 1, 1, 1)
REM first make all the temporary data files to be used by GNUPLOT
CALL MakeNewFiles
PRINT
a$ = "generating the GNUPLOT files": CALL box1(a$, 1, 1, 1)
PRINT "  making plot-sat.gnu"
CALL plotSAT
PRINT "  making plot-fo2.gnu"
CALL plotFO2
PRINT "  making plot-bp.gnu"
CALL plotBP
PRINT "  making plot-vap.gnu"
CALL plotVAPOUR
PRINT "  making plot-co2.gnu"
CALL plotCO2
PRINT "  making plot-tv.gnu"
CALL plotTV
PRINT "  ---- .gnu files all done ----"
REM ------------------------------------------------------------------------
PRINT
a$ = "plotting data plot-sat.gnu": CALL box1(a$, 1, 1, 1)
SHELL "gnuplot2 plot-sat.gnu"

CLS : LOCATE 5: a$ = "plotting data plot-fo2.gnu": CALL box1(a$, 1, 1, 1)
SHELL "gnuplot2 plot-fo2.gnu"

CLS :  LOCATE 5: a$ = "plotting data plot-bp.gnu": CALL box1(a$, 1, 1, 1)
SHELL "gnuplot2 plot-bp.gnu"

CLS : LOCATE 5: a$ = "plotting data plot-vap.gnu": CALL box1(a$, 1, 1, 1)
SHELL "gnuplot2 plot-vap.gnu"

CLS : LOCATE 5: a$ = "plotting data plot-co2.gnu": CALL box1(a$, 1, 1, 1)
SHELL "gnuplot2 plot-co2.gnu"

CLS : LOCATE 5: a$ = "plotting data plot-tv.gnu": CALL box1(a$, 1, 1, 1)
```

```
SHELL "gnuplot2 plot-tv.gnu"

REM-----------------------------------------------------------------------
REM now run the LaTeX  batch file
CLS : LOCATE 5
a$ = "calling LaTeX for typesetting": CALL box1(a$, 1, 1, 1)
SHELL "chdir c:\emtex\texinput"
REM normal laTeX 2.09 version of print format = prt-anes.tex
REM new laTeX2e version of print format = prtanes2.tex
    SHELL "latex prt-anes.tex"
    REM SHELL "latex prtanes2.tex"
REM ========================================
REM print the output file (.dvi file) to the HP-Deskjet-510
CLS : LOCATE 5
REM first isolate the filename extension (G01 etc)
extension$ = RIGHT$(Gnudatafile$, 3)
REM ---------------
REM now checkout the printer-status and set LETTER/ USLEGAL paper size etc
REM BUT note that we still need to write a subroutine here just for
REM   checking the printer status/setup/ and handle errors (eg if
REM   printer not switched on etc
REM need to select the correct printer (and have a default printer etc)
REM ----------------------------
REM --- so, assuming the printer is OK, then
REM  print the output file to the printer (HP-Deskjet-510)
CLS : LOCATE 5
a$ = "printing the Anaesthetic sheet to HP-DeskJet-510": CALL box1(a$, 1, 1, 1)
printdj$ = "prthplj /od+ prt-anes.dvi prn"
SHELL printdj$
REM ------------------
REM do *not* delete these files here, as can now print out all the current ones
REM if wish to
REM ========laserjet printer================================
     REM print it to the laserJet
     REM CLS : LOCATE 5
     REM a$ = "printing the record to LaserJet": CALL box1(a$, 1, 1, 1)
     REM SHELL "prthplj prt-anes.dvi prn"
     REM ========================================
REM return to the orig dir
SHELL "chdir c:\qb45\rs232\theatre"
END
```

17.1 Box1

```
SUB box1 (box$, t%, b%, s%)       ' =box with single line
L% = LEN(box$)                    ' REM  length of string
w% = 75                           ' REM width of the screen
leftmargin% = INT((w% - (L% + 2 * s%)) / 2)
PRINT STRING$(leftmargin%, 255); CHR$(218); STRING$((L% + 2 * s%), 196); CHR$(191)
```

```
PRINT STRING$(leftmargin%, 255); CHR$(179); STRING$(s%, 255); box$;
                                                    STRING$(s%, 255); CHR$(179)
PRINT STRING$(leftmargin%, 255); CHR$(192); STRING$((L% + 2 * s%), 196); CHR$(217)
END SUB
```

17.2 Box2

```
SUB box2 (box$, t%, b%, s%)
L% = LEN(box$)
w% = 75
margin% = INT((w% - (L% + 2 * s%)) / 2)
PRINT STRING$(margin%, 255); CHR$(201); STRING$((L% + 2 * s%), 205); CHR$(187)
PRINT STRING$(margin%, 255); CHR$(186); STRING$(s%, 255); box$; STRING$(s%, 255);
                                                                    CHR$(186)
PRINT STRING$(margin%, 255); CHR$(200); STRING$((L% + 2 * s%), 205); CHR$(188)
END SUB
```

17.3 Box1

```
SUB letterquality
REM enable LETTER quality ink-mode
REM LPRINT CHR$(27) + CHR$(40) + CHR$(115) + CHR$(50) + CHR$(81)
REM PRINT "returned to normal LETTER ink-mode"
REM CALL uslegal: PRINT "returned to uslegal OK"
PRINT "uslegal mode for DeskJet NOT set in PLOTAN8A.BAS (letterquality sub)"
END SUB
```

17.4 MakeNewFiles

```
SUB MakeNewFiles
REM to make 3 new files for BPs, BPd, and HR
REM BPs.G01 / BPd.G01 / HR.G01 etc
REM BPs is in cols 23-25
REM BPd is in cols 27-29
REM HRecg is in cols 15-17  (SpaceLabs output)
REM HRoxim is in cols 19-21  (Datex Cardiocap U03 output)
OPEN Gnudatafile$ FOR INPUT AS #20
OPEN "BPS.dat" FOR OUTPUT AS #22
OPEN "BPD.dat" FOR OUTPUT AS #23
OPEN "HRecg.dat" FOR OUTPUT AS #34
OPEN "HRoxim.dat" FOR OUTPUT AS #24
OPEN "TV.dat" FOR OUTPUT AS #25
OPEN "SAT.dat" FOR OUTPUT AS #26
OPEN "CO2.dat" FOR OUTPUT AS #27
OPEN "VAPIN.dat" FOR OUTPUT AS #28
OPEN "VAPOUT.dat" FOR OUTPUT AS #29
OPEN "FIO2.dat" FOR OUTPUT AS #30
OPEN "FIN2O.dat" FOR OUTPUT AS #35
```

```
OPEN "RR.dat" FOR OUTPUT AS #31
OPEN "CVP.dat" FOR OUTPUT AS #32
OPEN "MAC.dat" FOR OUTPUT AS #33
REM       #34 is used for HRecg.dat (see above)
REM      #35 is for FIN20.dat
REM ----------------------------
REM note problem of having empty lines at end of main data file
REM therefore check if there are any zeros in the time column
REM and add a # sign in front
REM --------------------------
DO WHILE NOT EOF(20)
    INPUT #20, line$
    TIME = VAL(MID$(line$, 8, 6))
    REM check whether time =0 or not
    REM if time=0 then put a # in its place to stop GNUPLOT plotting a value
    IF TIME < 1 THEN
         gnutime$ = "#"
         ELSE
         gnutime$ = STR$(TIME)
    END IF
    REM detect the --- characters
    REM for BPS
    IF MID$(line$, 23, 3) = "---" THEN
         PRINT #22, "#"; gnutime$, "---"
         ELSE
         BPS = VAL(MID$(line$, 23, 3))
              REM  IF abp1d < abp1s * (90 / 250) OR abp1d > abp1s * (190 / 250)
                                                          THEN bpnowcolour = green
         PRINT #22, gnutime$, BPS
         END IF
    REM for BPd
    IF MID$(line$, 27, 3) = "---" THEN
         PRINT #23, "#"; gnutime$, "---"
         ELSE
         BPD = VAL(MID$(line$, 27, 3))
         PRINT #23, gnutime$, BPD
         END IF
    REM for HRecg
    IF MID$(line$, 15, 3) = "---" THEN
         PRINT #34, "#"; gnutime$, "---"
         ELSE
         HRecg = VAL(MID$(line$, 15, 3))
         PRINT #34, gnutime$, HRecg
         END IF
    REM for HRoxim
    IF MID$(line$, 19, 3) = "---" THEN
         PRINT #24, "#"; gnutime$, "---"
         ELSE
         HRoxim = VAL(MID$(line$, 19, 3))
         PRINT #24, gnutime$, HRoxim
```

```
     END IF
IF MID$(line$, 49, 4) = "---" THEN
     PRINT #25, "#"; gnutime$, "---"
     ELSE
     TV = VAL(MID$(line$, 49, 4))
     PRINT #25, gnutime$, TV
     END IF
IF MID$(line$, 31, 3) = "---" THEN
     PRINT #26, "#"; gnutime$, "---"
     ELSE
     SAT = VAL(MID$(line$, 31, 3))
     PRINT #26, gnutime$, SAT
     END IF
IF MID$(line$, 39, 4) = "----" THEN
     PRINT #27, "#"; gnutime$, "----"
     ELSE
     CO2 = VAL(MID$(line$, 39, 4))
     PRINT #27, gnutime$, CO2
     END IF
REM vapours (vapin, vapout)
IF MID$(line$, 57, 5) = "---" THEN
     PRINT #28, "#"; gnutime$, "---"
     ELSE
     VAPIN = VAL(MID$(line$, 57, 5))
     PRINT #28, gnutime$, VAPIN
     END IF
IF MID$(line$, 63, 5) = "---" THEN
     PRINT #29, "#"; gnutime$, "---"
     ELSE
     VAPOUT = VAL(MID$(line$, 63, 5))
     PRINT #29, gnutime$, VAPOUT
     END IF
REM FIO2
IF MID$(line$, 35, 3) = "---" THEN
     PRINT #30, "#"; gnutime$, "---"
     ELSE
     FIO2 = VAL(MID$(line$, 35, 3))
     PRINT #30, gnutime$, FIO2
     END IF
REM RR (resp rate)
IF MID$(line$, 54, 2) = "--" THEN
     PRINT #31, "#"; gnutime$, "--"
     ELSE
     REM since RR is being plotted with TV, then scale RR to 0--1000
     REM by multiplying by 50 (20 * 50=1000)
     RR = 50 * VAL(MID$(line$, 54, 2))
     PRINT #31, gnutime$, RR
     END IF
REM CVP
IF MID$(line$, 69, 4) = "----" THEN
```

```
         PRINT #32, "#"; gnutime$, "---"
         ELSE
         CVP = VAL(MID$(line$, 69, 4))
         PRINT #32, gnutime$, CVP
         END IF
    REM MAC
    IF MID$(line$, 74, 4) = "----" THEN
         PRINT #33, "#"; gnutime$, "---"
         ELSE
         MAC = VAL(MID$(line$, 74, 4))
         PRINT #33, gnutime$, MAC
         END IF
    REM FIN2O
    IF MID$(line$, 79, 3) = "---" THEN
         PRINT #30, "#"; gnutime$, "---"
         ELSE
         FIN2O = VAL(MID$(line$, 79, 3))
         PRINT #35, gnutime$, FIN2O
         END IF
     REM SHUNT (virtual shunt)
     REM shunt not used- last used in anes-11J (shunt2) (SHUNT.DAT)
LOOP
CLOSE 20, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35
END SUB
```

17.5 PlotBP

```
SUB plotBP
REM -----------
REM plots BP, HRecg (SpaceLab), HRoxim, CVP
REM have a slight problem with how to select the appropriate HR
REM ------------
spac$ = CHR$(32)
spac5$ = CHR$(32) + CHR$(32) + CHR$(32) + CHR$(32) + CHR$(32)
quote$ = CHR$(34)
REM ------------
OPEN "c:plot-bp.gnu" FOR OUTPUT AS #1
PRINT #1, "# plot-bp.gnu prog made by [plotanes.bas]"
REM ==============================
PRINT #1, "set terminal emtex"
PRINT #1, "set output" + spac$ + quote$ + "c:\emtex\texinput\plot-bp.pic" + quote$
PRINT #1, "# NB. full size = 5x3 inches:set x,y"
PRINT #1, "set size 1.5,1": REM 1.5,1
REM ==============================
CALL timebase
REM-------------
REM  NB the set xtics is done by the TIMEBASE sub
   z0$ = quote$ + "0" + quote$ + spac$ + "0,"
  z20$ = quote$ + "20" + quote$ + spac$ + "20,"
```

```
  z50$ = quote$ + "50" + quote$ + spac$ + "50,"
 z100$ = quote$ + "100" + quote$ + spac$ + "100,"
 z150$ = quote$ + "150" + quote$ + spac$ + "150,"
 z200$ = quote$ + "200" + quote$ + spac$ + "200"
PRINT #1, "set ytics("; z0$ + z20$ + z50$ + z100$ + z150$ + z200$ + ")"
PRINT #1, "set title" + quote$ + "\hspace*{12cm} \sc{" + Gnudatafile$ + "}" + quote$
PRINT #1, "set ylabel" + quote$ + "BP sys $\bigcirc$\\BP dias $\circ$\\HR
                                    $\bullet$\\CVP {\bf ---}" + quote$ + "-3"
PRINT #1, "set nokey"
PRINT #1, "set grid"
PRINT #1, "xmin=0; xmax=3600"
PRINT #1, "ymin=0; ymax=200"
PRINT #1, "plot [xmin:xmax][ymin:ymax] 98 with lines 1,\"
REM draw a solid line  at 20, and the dotted line at 10 mm Hg for CVP
PRINT #1, spac5$ + "20 with lines 1,\"
PRINT #1, spac5$ + "10 with lines 4,\"
REM BPs = col 4  (lines points 1 9)
REM BPd = col 5  (lines points 4 8)
REM Pulse = col 3 (cols 2 & 3)  (lines points 1 10)
REM Shunt = col n (lines points 4 3)(squares)
REM plot the SYSTOLIC BP
PRINT #1, spac5$ + quote$ + "BPS.DAT" + quote$ + spac$ + "using 1:2 with
                                                          linespoints 1 9,\"
REM plot the DIASTOLIC BP
PRINT #1, spac5$ + quote$ + "BPD.DAT" + quote$ + spac$ + "using 1:2 with
                                                          linespoints 1 8,\"
REM plot the HRecg
PRINT #1, spac5$ + quote$ + "HRecg.DAT" + quote$ + spac$ + "using 1:2 with
                                                         linespoints 1 10,\"
REM plot the HRoxim
PRINT #1, spac5$ + quote$ + "HRoxim.DAT" + quote$ + spac$ + "using 1:2 with
                                                              points 1 10,\"
REM plot the CVP data (points only(3), no lines)
PRINT #1, spac5$ + quote$ + "CVP.DAT" + quote$ + spac$ + "using 1:2 with
                                                                   lines 2"
CLOSE
REM ------
END SUB
```

17.6 PlotCO2

```
SUB plotCO2
REM -----------
spac$ = CHR$(32)
spac5$ = CHR$(32) + CHR$(32) + CHR$(32) + CHR$(32) + CHR$(32)
quote$ = CHR$(34)
REM ------------
OPEN "c:plot-co2.gnu" FOR OUTPUT AS #1
PRINT #1, "# plot-co2.gnu prog made by [plotanes.bas]"
```

```
REM ==============================
PRINT #1, "set terminal emtex"
PRINT #1, "set output" + spac$ + quote$ + "c:\emtex\texinput\plot-co2.pic" + quote$
PRINT #1, "# NB. full size = 5x3 inches:set x,y"
PRINT #1, "set size 1.5,0.575": REM 1.5,.6
REM ==============================
CALL timebase
REM-------------
PRINT #1, "set ylabel" + quote$ + "CO$_2$ \% \\(exp\  $\diamond$\ )\\(insp  ---)"
                                                                    + quote$ + "-2"
PRINT #1, "set ytics 2, 2, 8"
PRINT #1, "set nokey"
PRINT #1, "set grid"
PRINT #1, "xmin=0; xmax=3600"
PRINT #1, "ymin=2; ymax=8"
PRINT #1, "plot [xmin:xmax][ymin:ymax] \"
REM CO2 = col 9  (lines points 4 1)
REM plot the CO2
PRINT #1, spac5$ + quote$ + "CO2.DAT" + quote$ + spac$ + "using 1:2 with
                                                          linespoints 4 1"
CLOSE
REM -----
END SUB
```

17.7 PlotFO2

```
SUB plotFO2
REM -----------
spac$ = CHR$(32)
spac5$ = CHR$(32) + CHR$(32) + CHR$(32) + CHR$(32) + CHR$(32)
quote$ = CHR$(34)
REM ------------
OPEN "c:plot-fo2.gnu" FOR OUTPUT AS #1
PRINT #1, "# plot-fo2.gnu prog made by [plotanes.bas]"
REM ==============================
PRINT #1, "set terminal emtex"
PRINT #1, "set output" + spac$ + quote$ + "c:\emtex\texinput\plot-fo2.pic"
                                                                    + quote$
PRINT #1, "# NB. full size = 5x3 inches:set x,y"
PRINT #1, "set size 1.5, 0.575"
REM ==============================
CALL timebase
REM-------------
REM use this if printing SHUNT
REM PRINT #1, "set ylabel" + quote$ + "\% \\FIO$_2$ $\bullet$\\21\% ....\\Shunt
                                                  {\bf ---}" + quote$ + "-2"
REM ------
PRINT #1, "set ylabel" + quote$ + "\% \\FIN$_2$O $\Box$\\FIO$_2$ $\bullet$\\
                                                 21\% ...." + quote$ + "-1"
```

```
PRINT #1, "set ytics 10,20,70"
PRINT #1, "set nokey"
PRINT #1, "set grid"
PRINT #1, "xmin=0; xmax=3600"
PRINT #1, "ymin=10; ymax=70"
PRINT #1, "plot [xmin:xmax][ymin:ymax] 21 with lines 4, \"
REM ----------
REM FIo2 = col 8  (lines points 4, 10 = small black dot)
REM plot the FIO2
PRINT #1, spac5$ + quote$ + "FIO2.DAT" + quote$ + spac$ + "using 1:2 with
                                                          linespoints 4 10,\"
REM ----------
REM FIN2O = col ??  (lines points 4, 3 = small square)
REM plot the FIN2O
PRINT #1, spac5$ + quote$ + "FIN2O.DAT" + quote$ + spac$ + "using 1:2 with
                                                             linespoints 4 3"
REM ----------
REM SHUNT = col 15  (with lines 2)
REM plot the SHUNT
REM remember to add the extra <, \ > at the end of the above line
REM PRINT #1, spac5$ + quote$ + "SHUNT.DAT" + quote$ + spac$ + "using 1:2 with
                                                                 lines 2"
REM -----------
CLOSE
REM ----------
END SUB
```

17.8 PlotSAT

```
SUB plotSAT
REM -----------
spac$ = CHR$(32)
spac5$ = CHR$(32) + CHR$(32) + CHR$(32) + CHR$(32) + CHR$(32)
quote$ = CHR$(34)
REM ------------
OPEN "c:plot-sat.gnu" FOR OUTPUT AS #1
PRINT #1, "# plot-sat.gnu prog made by [plotanes.bas]"
REM ==============================
PRINT #1, "set terminal emtex"
PRINT #1, "set output" + spac$ + quote$ + "c:\emtex\texinput\plot-sat.pic" + quote$
PRINT #1, "# NB. full size = 5x3 inches:set x,y"
PRINT #1, "set size 1.5,0.575"
REM ==============================
CALL timebase
REM-------------
PRINT #1, "set ylabel" + quote$ + "\% \\Sat $\circ$\\FIO$_2$ $\bullet$" + quote$
PRINT #1, "set ytics 80, 10, 100"
PRINT #1, "set nokey"
PRINT #1, "set grid"
```

```
PRINT #1, "xmin=0; xmax=3600"
PRINT #1, "ymin=80; ymax=100"
PRINT #1, "plot [xmin:xmax][ymin:ymax] \"
REM SAT = col 7  (lines points 4 8)
REM plot the SAT
PRINT #1, spac5$ + quote$ + "SAT.DAT" + quote$ + spac$ + "using 1:2 with
                                                          linespoints 4 8,\"
REM plot the FIO2 on same graph
PRINT #1, spac5$ + quote$ + "FIO2.DAT" + quote$ + spac$ + "using 1:2 with
                                                           linespoints 4 10"
CLOSE
REM ------
END SUB
```

17.9 PlotTV

```
SUB plotTV
REM -----------
spac$ = CHR$(32)
spac5$ = CHR$(32) + CHR$(32) + CHR$(32) + CHR$(32) + CHR$(32)
quote$ = CHR$(34)
REM ------------
OPEN "c:plot-tv.gnu" FOR OUTPUT AS #1
PRINT #1, "# plot-tv.gnu prog made by [plotanes.bas]"
REM ==============================
PRINT #1, "set terminal emtex"
PRINT #1, "set output" + spac$ + quote$ + "c:\emtex\texinput\plot-tv.pic" + quote$
PRINT #1, "# NB. full size = 5x3 inches:set x,y"
PRINT #1, "set size 1.5,0.575"
REM ==============================
CALL timebase
REM-------------
PRINT #1, "set ylabel" + quote$ + "tidal vol $\Box$\\(0--1000 mls)\\resp
                                  rate $\bullet$\\(0--20/min)" + quote$ + "-5"
PRINT #1, "set ytics 0, 250, 1000"
PRINT #1, "set nokey"
PRINT #1, "set grid"
PRINT #1, "xmin=0; xmax=3600"
PRINT #1, "ymin=0; ymax=1000"
PRINT #1, "plot [xmin:xmax][ymin:ymax] \"
REM exp Tidal Vol = col 11  (lines points 4 1)
REM squares = points 3
REM plot the Exp Tidal Vol  as squares (points 3: note must keep the place)
PRINT #1, spac5$ + quote$ + "TV.DAT" + quote$ + spac$ + "using 1:2 with
                                                          linespoints 4 3,\"
REM plot the Resp Rate (RR) as dotted lines (lines 4)
PRINT #1, spac5$ + quote$ + "RR.DAT" + quote$ + spac$ + "using 1:2 with
                                                           linespoints 4 10"
CLOSE
```

```
REM -----
END SUB
```

17.10 PlotVapour

```
SUB plotVAPOUR
REM -----------
spac$ = CHR$(32)
spac5$ = CHR$(32) + CHR$(32) + CHR$(32) + CHR$(32) + CHR$(32)
quote$ = CHR$(34)
REM ------------
OPEN "c:plot-vap.gnu" FOR OUTPUT AS #1
PRINT #1, "# plot-vap.gnu prog made by [plotanes.bas]"
REM ==============================
PRINT #1, "set terminal emtex"
PRINT #1, "set output" + spac$ + quote$ + "c:\emtex\texinput\plot-vap.pic" + quote$
PRINT #1, "# NB. full size = 5x3 inches:set x,y"
PRINT #1, "set size 1.5,0.575"
REM ==============================
CALL timebase
REM-------------
REM need to be able to be flexible wrt vapourname --> vapourname$ etc
PRINT #1, "set ylabel" + quote$ + "isoflurane \% \\(insp  ---)\\(exp  ....)\\total
                                  MAC $\diamond$\\(age corrected)" + quote$ + "-4"
REM xtics and ytics are START, INCREMENT, END
PRINT #1, "set ytics 0, 1, 4"
PRINT #1, "set nokey"
PRINT #1, "set grid"
PRINT #1, "xmin=0; xmax=3600"
PRINT #1, "ymin=0; ymax=4"
PRINT #1, "plot [xmin:xmax][ymin:ymax] \"
REM ------------
REM VAPOURin = col 13 / VAPOURout = col 14  (lines points 4 10)
REM MAC (bmac)= col 16 (lines points 4 1)(small rhomboids = 1; dotted line = 4)
REM -----------
REM plot the VAPOUR
REM single line = lines 1/bold line =2 / very bold line =3
REM dotted line = lines 4
REM PRINT #1, spac5$ + quote$ + gnudatafile$ + quote$ + spac$ + "using 2:13 with
                                                              linespoints 4 10"
REM
PRINT #1, spac5$ + quote$ + "VAPIN.DAT" + quote$ + spac$ + "using 1:2 with
                                                               lines 2,\"
PRINT #1, spac5$ + quote$ + "VAPOUT.DAT" + quote$ + spac$ + "using 1:2 with
                                                               lines 4,\"
REM print the MAC with points ONLY (rhomboids = points 1)
PRINT #1, spac5$ + quote$ + "MAC.DAT" + quote$ + spac$ + "using 1:2 with
                                                             points 4 1"
CLOSE
```

```
REM --------
END SUB
```

17.11 Printerstatus

```
SUB printerstatus
REM this is experimental - not working yet/not being used
REM
REM to check to see if problem with printer
REM set the errorstring variable (used in the errorhandler routine)
error$ = "printer"
REM read the STATUS register of parallel port
     PPStatusReg% = &H379
     p% = INP(PPStatusReg%)
IF p% <> 223 THEN
      BEEP: BEEP
      a$ = "Problem with printer "
      PRINT a$
      REM --------------------
      SELECT CASE p%
          CASE 79
             p$ = "printer is switched OFF"
          CASE 20000
             p$ = "printer is ON but is BUSY"
      END SELECT
      REM --------------------
      PRINT p$
      c$ = " Fix the printer and then press any key"
      PRINT b$
      DO
         p% = INP(PPStatusReg%)
         PRINT p%: SLEEP 1
         key$ = INKEY$
         IF UCASE$(key$) = "Q" THEN END
      LOOP UNTIL p% = 223
END IF
REM enable LETTER quality ink-mode
REM LPRINT CHR$(27) + CHR$(40) + CHR$(115) + CHR$(50) + CHR$(81)
REM PRINT "returned to normal LETTER ink-mode"
REM CALL uslegal: PRINT "returned to uslegal OK"
PRINT "uslegal mode for DeskJet NOT set in PLOTANES.BAS (printerstatus sub)"
DO
        p% = INP(PPStatusReg%)
        PRINT p%: SLEEP 1
        key$ = INKEY$
        IF UCASE$(key$) = "Q" THEN END
LOOP
END SUB
```

17.12 TimeBase

```
SUB timebase
REM this is to get the correct times for the graphs,
REM and line them up correctly
REM this is called from the subroutine PLOTANES
REM -------------
spac$ = CHR$(32)
spac5$ = CHR$(32) + CHR$(32) + CHR$(32) + CHR$(32) + CHR$(32)
quote$ = CHR$(34)
REM ------------
REM determine the time - deltah - (in secs) to next full hour h
deltah = 3600 - starttime# MOD 3600
REM ----------------
REM get next full hour (h)
REM use string manipulation to extract the integer
REM get the number up to the decimal point  using integer division with \
REM also make the clock cycle at midnight to 00.00 etc (MOD 24)
h = ((starttime# \ 3600) + 1) MOD 24
REM -------------
PRINT #1, "# starttime ="; starttime#
PRINT #1, "# h ="; h
PRINT #1, "# give deltahour (deltah) in secs from start (starttime)"
PRINT #1, "deltah = "; deltah
PRINT #1, "q=900; qq=1800; qqq=2700; qqqq=3600"
hminus1 = h - 1
hplus1 = h + 1
IF h = 23 THEN hplus1 = 0
IF h = 0 THEN hminus1 = 23
t1$ = quote$ + RIGHT$("00" + LTRIM$(STR$(hminus1)), 2) + ".00"
                                        + quote$ + spac$ + "deltah-qqqq" + ","
t2$ = quote$ + RIGHT$("00" + LTRIM$(STR$(hminus1)), 2) + ".15"
                                          + quote$ + spac$ + "deltah-qqq" + ","
t3$ = quote$ + RIGHT$("00" + LTRIM$(STR$(hminus1)), 2) + ".30"
                                           + quote$ + spac$ + "deltah-qq" + ","
t4$ = quote$ + RIGHT$("00" + LTRIM$(STR$(hminus1)), 2) + ".45"
                                            + quote$ + spac$ + "deltah-q" + ","
t5$ = quote$ + RIGHT$("00" + LTRIM$(STR$(h)), 2) + ".00"
                                              + quote$ + spac$ + "deltah" + ","
t6$ = quote$ + RIGHT$("00" + LTRIM$(STR$(h)), 2) + ".15"
                                            + quote$ + spac$ + "deltah+q" + ","
t7$ = quote$ + RIGHT$("00" + LTRIM$(STR$(h)), 2) + ".30"
                                           + quote$ + spac$ + "deltah+qq" + ","
t8$ = quote$ + RIGHT$("00" + LTRIM$(STR$(h)), 2) + ".45"
                                          + quote$ + spac$ + "deltah+qqq" + ","
t9$ = quote$ + RIGHT$("00" + LTRIM$(STR$(hplus1)), 2) + ".00"
                                              + quote$ + spac$ + "deltah-qqqq"
PRINT #1, "set xtics(" + t1$ + t2$ + t3$ + t4$ + t5$ + t6$ + t7$ + t8$ + t9$ + ")"
REM ---------
END SUB
```

Chapter 18

Example of data files output by `plotanes.pb`

ch-pexam.tex

In order for GNUplot to run easily, all the data for plotting is written to a separate time/data file for each parameter. The value in the left column is the time in seconds from the beginning of the current screen (i.e. from the beginning of the current 'hour'). The right column contains the parameter value. If the parameter value is `---` then the GBUplot comment symbol `#` is placed at the beginning of the line, which therefore prevents processing by GNUplot. The full list of such data files is as follows.

```
bps.dat
bpd.dat
co2.dat
cvp.dat
fio2.dat
fin2o.dat
hrecg.dat
hroxim.dat
mac.dat
rr.dat
sat.dat
tv.dat
vapin.dat
vapout.dat
```

18.1 bps.dat

```
# 8          ---
# 18         ---
# 58         ---
# 98         ---
# 138        ---
# 178        ---
# 218        ---
# 227        ---
# 268        ---
# 271        ---
# 282        ---
# 323        ---
  337        120
```

18.2 bpd.dat

```
# 8          ---
# 18         ---
# 58         ---
# 98         ---
# 138        ---
# 178        ---
# 218        ---
# 227        ---
# 268        ---
# 271        ---
# 282        ---
# 323        ---
 337          75
```

18.3 co2.dat

```
 8          -100
 18          4.24
 58          4.25
 98          4.26
 138         4.26
 178         4.42
 218         4.24
 227         4.27
```

```
268          4.25
271          4.29
282          4.31
323          4.31
337          4.35
```

18.4 cvp.dat

```
8           -100
18           5
58           5
98           5
138          5
178          5
218          5
227          5
268          5
271          5
282          5
323          5
337          5
```

18.5 fio2.dat

```
8           -99
18           75
58           75
98           75
138          75
178          75
218          75
227          75
268          75
271          75
282          75
323          75
337          75
```

18.6 fin2o.dat

```
 8          -99
 18          51
 58          51
 98          51
 138         51
 178         51
 218         51
 227         52
 268         51
 271         51
 282         51
 323         52
 337         52
```

18.7 hrecg.dat

```
# 8         ---
 18          82
 58          82
 98          82
 138         83
 178         83
 218         83
 227         83
 268         83
 271         83
 282         83
 323         84
 337         84
```

18.8 hroxim.dat

```
# 8         ---
 18          82
 58          82
 98          82
 138         83
 178         83
 218         83
 227         83
```

```
268          83
271          83
282          83
323          84
337          84
```

18.9 mac.dat

```
8           -100
18           1.12
58           1.15
98           1.14
138          1.15
178          1.15
218          1.16
227          1.15
268          1.16
271          1.15
282          1.17
323          1.18
337          1.17
```

18.10 rr.dat

```
# 8          --
 18          400
 58          400
 98          400
 138         400
 178         400
 218         400
 227         400
 268         400
 271         400
 282         400
 323         400
 337         400
```

18.11 sat.dat

```
# 8           ---
 18            96
 58            96
 98            96
 138           96
 178           96
 218           96
 227           96
 268           96
 271           96
 282           96
 323           96
 337           96
```

18.12 tv.dat

```
 8           -999
 18           591
 58           612
 98           608
 138          631
 178          599
 218          592
 227          610
 268          604
 271          603
 282          593
 323          593
 337          593
```

18.13 vapin.dat

```
 8            0
 18           1.37
 58           1.37
 98           1.37
 138          1.38
 178          1.38
 218          1.39
 227          1.39
```

```
268             1.39
271             1.4
282             1.39
323             1.4
337             1.4
```

18.14 vapout.dat

```
8               0
18              1.2
58              1.2
98              1.19
138             1.2
178             1.2
218             1.21
227             1.21
268             1.22
271             1.22
282             1.22
323             1.24
337             1.22
```

Chapter 19

GNUplot files

ch-pgnu.tex

The following are typical examples of the `.gnu` GNUplot scripts (each output by the associated SUB in the previous program `plotan3a.pb`, page 217) which are used to draw the graphs. Each GNUplot script is then processed by the free `GNUplot` utility program, and generates a single `.pic` file in LaTeX picture format. All the graphs, which are the same height and width, are then typeset by LaTeXing the file `prtanes2.tex` (page 244), which inputs all the `.pic` files.

```
SUB plotBP   --> plot-bp.gnu  --> plot-bp.pic
SUB plotSAT  --> plot-sat.gnu --> plot-sat.pic
SUB plotFO2  --> plot-fo2.gnu --> plot-fo2.pic
SUB plotCO2  --> plot-co2.gnu --> plot-co2.pic
SUB plotRR   --> plot-rr.gnu  --> plot-rr.pic
SUB plotVAP  --> plot-vap.gnu --> plot-vap.pic
```

19.1 Plot-BP.gnu

```
# plot-bp.gnu script made by SUB plotBP (in plotanes.bas)
set terminal emtex
set output "c:\emtex\texinput\plot-bp.pic"
# NB. full size = 5x3 inches:set x,y
set size 1.5,1
# starttime = 47585
# h = 14
# give deltahour (deltah) in secs from start (starttime)
deltah =  2815
q=900; qq=1800; qqq=2700; qqqq=3600
set xtics("13.00" deltah-qqqq,"13.15" deltah-qqq,"13.30" deltah-qq,
```

```
          "13.45" deltah-q,"14.00" deltah,"14.15" deltah+q,
          "14.30" deltah+qq,"14.45" deltah+qqq,"15.00" deltah-qqqq)
set ytics("0" 0,"20" 20,"50" 50,"100" 100,"150" 150,"200" 200)
set title"\hspace*{12cm} \sc{99B21D12.G01}"
set ylabel"BP sys $\bigcirc$\\BP dias $\circ$\\HR $\bullet$\\CVP {\bf ---}"-3
set nokey
set grid
xmin=0; xmax=3600
ymin=0; ymax=200
plot [xmin:xmax][ymin:ymax] 98 with lines 1,\
     20 with lines 1,\
     10 with lines 4,\
     "BPS.DAT" using 1:2 with linespoints 1 9,\
     "BPD.DAT" using 1:2 with linespoints 1 8,\
     "HRecg.DAT" using 1:2 with linespoints 1 10,\
     "HRoxim.DAT" using 1:2 with points 1 10,\
     "CVP.DAT" using 1:2 with lines 2
```

19.2 Plot-CO2.gnu

```
# plot-co2.gnu script made by SUB plotCO2 (in plotanes.bas)
set terminal emtex
set output "c:\emtex\texinput\plot-co2.pic"
# NB. full size = 5x3 inches:set x,y
set size 1.5,0.575
# starttime = 47585
# h = 14
# give deltahour (deltah) in secs from start (starttime)
deltah =  2815
q=900; qq=1800; qqq=2700; qqqq=3600
set xtics("13.00" deltah-qqqq,"13.15" deltah-qqq, "13.30" deltah-qq,
          "13.45" deltah-q, "14.00" deltah,"14.15" deltah+q,
          "14.30" deltah+qq,"14.45" deltah+qqq, "15.00" deltah-qqqq)
set ylabel"CO$_2$ \% \\(exp\  $\diamond$\ )\\(insp  ---)"-2
set ytics 2, 2, 8
set nokey
set grid
xmin=0; xmax=3600
ymin=2; ymax=8
plot [xmin:xmax][ymin:ymax] \
     "CO2.DAT" using 1:2 with linespoints 4 1
```

19.3 Plot-fo2.gnu

```
# plot-fo2.gnu script made by SUB plotFO2 (in plotanes.bas)
set terminal emtex
set output "c:\emtex\texinput\plot-fo2.pic"
```

```
# NB. full size = 5x3 inches:set x,y
set size 1.5, 0.575
# starttime = 47585
# h = 14
# give deltahour (deltah) in secs from start (starttime)
deltah =  2815
q=900; qq=1800; qqq=2700; qqqq=3600
set xtics("13.00" deltah-qqqq,"13.15" deltah-qqq,"13.30" deltah-qq,
          "13.45" deltah-q,"14.00" deltah,"14.15" deltah+q,
          "14.30" deltah+qq,"14.45" deltah+qqq,"15.00" deltah-qqqq)
set ylabel"\% \\FIN$_2$O $\Box$\\FIO$_2$ $\bullet$\\ 21\% ...."-1
set ytics 10,20,70
set nokey
set grid
xmin=0; xmax=3600
ymin=10; ymax=70
plot [xmin:xmax][ymin:ymax] 21 with lines 4, \
     "FIO2.DAT" using 1:2 with linespoints 4 10,\
     "FIN2O.DAT" using 1:2 with linespoints 4 3
```

19.4 Plot-sat.gnu

```
# plot-sat.gnu script made by SUB plotSAT (in plotanes.bas)
set terminal emtex
set output "c:\emtex\texinput\plot-sat.pic"
# NB. full size = 5x3 inches:set x,y
set size 1.5,0.575
# starttime = 47585
# h = 14
# give deltahour (deltah) in secs from start (starttime)
deltah =  2815
q=900; qq=1800; qqq=2700; qqqq=3600
set xtics("13.00" deltah-qqqq,"13.15" deltah-qqq,"13.30" deltah-qq,
          "13.45" deltah-q,"14.00" deltah,"14.15" deltah+q,
          "14.30" deltah+qq,"14.45" deltah+qqq,"15.00" deltah-qqqq)
set ylabel"\% \\Sat $\circ$\\FIO$_2$ $\bullet$"
set ytics 80, 10, 100
set nokey
set grid
xmin=0; xmax=3600
ymin=80; ymax=100
plot [xmin:xmax][ymin:ymax] \
     "SAT.DAT" using 1:2 with linespoints 4 8,\
     "FIO2.DAT" using 1:2 with linespoints 4 10
```

19.5 Plot-tv.gnu

```
# plot-tv.gnu script made by SUB plotTV (in plotanes.bas)
set terminal emtex
set output "c:\emtex\texinput\plot-tv.pic"
# NB. full size = 5x3 inches:set x,y
set size 1.5,0.575
# starttime = 47585
# h = 14
# give deltahour (deltah) in secs from start (starttime)
deltah =  2815
q=900; qq=1800; qqq=2700; qqqq=3600
set xtics("13.00" deltah-qqqq,"13.15" deltah-qqq,"13.30" deltah-qq,
          "13.45" deltah-q,"14.00" deltah,"14.15" deltah+q,
          "14.30" deltah+qq,"14.45" deltah+qqq,"15.00" deltah-qqqq)
set ylabel"tidal vol $\Box$\\(0--1000 mls)\\resp rate $\bullet$\\(0--20/min)"-5
set ytics 0, 250, 1000
set nokey
set grid
xmin=0; xmax=3600
ymin=0; ymax=1000
plot [xmin:xmax][ymin:ymax] \
     "TV.DAT" using 1:2 with linespoints 4 3,\
     "RR.DAT" using 1:2 with linespoints 4 10
```

19.6 Plot-vap.gnu

```
# plot-vap.gnu script made by SUB plotVAP (in plotanes.bas)
set terminal emtex
set output "c:\emtex\texinput\plot-vap.pic"
# NB. full size = 5x3 inches:set x,y
set size 1.5,0.575
# starttime = 47585
# h = 14
# give deltahour (deltah) in secs from start (starttime)
deltah =  2815
q=900; qq=1800; qqq=2700; qqqq=3600
set xtics("13.00" deltah-qqqq,"13.15" deltah-qqq,"13.30" deltah-qq,
          "13.45" deltah-q,"14.00" deltah,"14.15" deltah+q,
          "14.30" deltah+qq,"14.45" deltah+qqq,"15.00" deltah-qqqq)
set ylabel"isoflurane \% \\(insp  ---)\\(exp  ....)\\total
                                        MAC $\diamond$\\(age corrected)"-4
set ytics 0, 1, 4
set nokey
set grid
xmin=0; xmax=3600
ymin=0; ymax=4
plot [xmin:xmax][ymin:ymax] \
     "VAPIN.DAT" using 1:2 with lines 2,\
```

```
     "VAPOUT.DAT" using 1:2 with lines 4,\
     "MAC.DAT" using 1:2 with points 4 1
```

19.7 GNUplot and LaTeX picture format

The following example shows the code output by GNUplot in LaTeX picture format. Note the command `\tenrm` is used, and therefore we need to use the command

```
\newcommand{\tenrm}{\rmfamily\normalsize}
```

to redefine the old LaTeX 2.09 command.

```
%%--------------------------------------
%% RN-tv.pic  FINAL version July 19 1998
%% this file is required for RN-Fig1.tex
%%--------------------------------------
% GNUPLOT: LaTeX picture with emtex specials
\setlength{\unitlength}{0.240900pt}
\ifx\plotpoint\undefined\newsavebox{\plotpoint}\fi
\sbox{\plotpoint}{\rule[-0.175pt]{0.350pt}{0.350pt}}%
\special{em:linewidth 0.3pt}%
\begin{picture}(1800,476)(0,0)
\tenrm
\put(264,158){\special{em:moveto}}
\put(1736,158){\special{em:lineto}}
\put(264,158){\special{em:moveto}}
\put(264,363){\special{em:lineto}}
\put(264,158){\special{em:moveto}}
\put(1736,158){\special{em:lineto}}
...
...
\put(1736,158){\special{em:lineto}}
\put(1736,363){\special{em:lineto}}
\put(264,363){\special{em:lineto}}
\put(264,158){\special{em:lineto}}
\put(-65,260){\makebox(0,0)[l]{\shortstack{Tidal vol $\Box$\\(0--1000 mls)
                                          \\Resp rate $\bullet$\\(0--20/min)}}}
\sbox{\plotpoint}{\rule[-0.250pt]{0.500pt}{0.500pt}}%
\special{em:linewidth 0.5pt}%
\put(281,243){\usebox{\plotpoint}}
\put(281,243){\usebox{\plotpoint}}
...
...
\put(1588,292){\raisebox{-1.2pt}{\makebox(0,0){$\Box$}}}
\put(1624,283){\raisebox{-1.2pt}{\makebox(0,0){$\Box$}}}
\put(1656,281){\raisebox{-1.2pt}{\makebox(0,0){$\Box$}}}
\put(1689,283){\raisebox{-1.2pt}{\makebox(0,0){$\Box$}}}
\put(1722,286){\raisebox{-1.2pt}{\makebox(0,0){$\Box$}}}
\put(281,281){\usebox{\plotpoint}}
\put(281,281){\usebox{\plotpoint}}
```

```
\put(301,281){\usebox{\plotpoint}}
\put(322,281){\usebox{\plotpoint}}
...
...
\put(1588,260){\circle*{12}}
\put(1624,260){\circle*{12}}
\put(1656,260){\circle*{12}}
\put(1689,260){\circle*{12}}
\put(1722,260){\circle*{12}}
\end{picture}
```

Chapter 20

LaTeX files for typesetting the graphs (prtanes2.tex)

ch-pan2.tex

One of these `.tex` files is used for typesetting the output graphs (`.pic` files) on to one sheet of A4; one page per hour. It is called by the program `plotan3a.pb`. There are two forms, one for LaTeX 2_ε and the other for the old LaTeX 2.09 version.

20.1 LaTeX 2_ε

```
%% PRTanes2.TEX to print out the anaesthetic chart
%% LaTeX 2e version
%% copied from PRT-ANES.TEX = Latex2.09 version
%%--------------
%% June 2, 2001
%%
%%---------------
%% July 21 1998
%% making it latex-2e capable
%%----------------
%% March 4 1998
%% adjusted to include Shunt and total MAC
%%----------------
%% June 6 1996
%% adjusted to put BP at the top
%% old version = prt-anes.org
%%-----------------------------
%% June 3, 1996
%% adjusted to put Theatre 1 instead of 4
%% adjusted to put rr calibration on RHS of RR graph
%%------------
% for plotting the graphs called
% plot-bp.gnu  --> plot-bp.pic
% plot-sat.gnu --> plot-sat.pic
```

```
% plot-fo2 --> plot-fo2.pic
% plot-co2 --> plot-co2.pic
% plot-rr ---> plot-rr.pic
% plot-iso --> plot-iso.pic
%%-------------------------------------
%%
\documentclass[a4paper]{article}
\usepackage{latexsym}  % for the \Box symbol
%% redefine the \tenrm command output by GNUplot
%% in the .PIC files
\newcommand{\tenrm}{\rmfamily\normalsize}

%%
%%
%%
\voffset -2.2cm
\oddsidemargin -1.5cm
\textwidth 19.5cm
\textheight 26cm  %26 25.5
%%
\begin{document}
%%
%---------------------------------
%% note that all the empty lines are ESSENTIAL for layout
%% as \vspace{..} seems to require a preceeding free line
%%--------------------------------

\vspace{-1.8cm}
{\ } \hfill
\framebox{\rule[-10mm]{0cm}{3.3cm}\hspace{2.2cm}Patient
                         label\hspace{2.2cm}}\hspace{7mm} %2mm

\vspace{-3.6cm}
\noindent\hspace{3.1cm}{\LARGE ANAESTHETIC SHEET}

\vspace{3mm}
\hspace*{2.5cm} Theatre 1 -- City Hospital, Nottingham, UK.%

 %%----------------------------------------

\vspace{5mm}
\hspace*{1.54cm}\vbox{
\begin{tabular}{|ll|}
\hline
{\sc Date:} \rule{0pt}{12pt}  & \today   \\
{\sc Operation:}  &   \\
{\sc Anaesthetists:} & RWD Nickalls {\it et al \hspace{2cm}} \\
{\sc Surgeons:} &  \\
\hline
\end{tabular}
```

```
 }
%---------------------------------------

%****************
\vspace{-2mm}
\noindent \hspace{1.5mm}
   \input{plot-bp.pic}
%*************************

%******************
\vspace{-16mm}
\noindent\hspace{1.5mm}
      \input{plot-sat.pic}
%******************

%*************************
\vspace{-16mm}  %-20
\noindent\hspace{1.5mm}
    \input{plot-fo2.pic}
%******************


%*************************
\vspace{-16mm}
\noindent\hspace{1.5mm}
   \input{plot-co2.pic}
%******************

%*************************
\vspace{-16mm}
\noindent\hspace{1.5mm}
    \input{plot-tv.pic}
%%---------------------
%% now put on the right axis for Resp rate (0, 5,10,15,20).

 \vspace{-37mm} \noindent\hspace{188.5mm} 20 $\bullet$

 \vspace{1mm}\noindent\hspace{188.5mm} {15}

 \vspace{0.8mm}\noindent\hspace{188.5mm} {10}

 \vspace{1mm}\noindent\hspace{189mm} {5}

 \vspace{1mm}\noindent\hspace{189mm} {0}
%

\vspace{-4mm}
%*************************
%\vspace{-16mm}
\noindent\hspace{1.5mm}
```

```
     \input{plot-vap.pic}
%******************

  \vspace{-1cm}   %% keep the following empty line

  \rule{6.5cm}{.5pt}  %% keep the following empty line

 {\footnotesize
 Data from SpaceLabs Medical and Datex series-II monitors
  \hfill  {\sc anaesthesia record system--3a} \copyright\ RWD Nickalls,
1994--1998 \hspace{5mm}
  }
\end{document}
```

20.2 LaTeX 2.09

```
%% PRT-ANES.TEX to print out the anaesthetic chart
%% latex 2.09 version
%%----------------
%% plot-bp.gnu  --> plot-bp.pic
%% plot-sat.gnu --> plot-sat.pic
%% plot-fo2 --> plot-fo2.pic
%% plot-co2 --> plot-co2.pic
%% plot-rr ---> plot-rr.pic
%% plot-iso --> plot-iso.pic
%%-------------------------------------
\documentstyle[a4,miscrwdn]{article}
\voffset -2.2cm \oddsidemargin -1.5cm
\textwidth 19.5cm \textheight 26cm  %26 25.5
\begin{document}
%---------------------------------
% now draw the marks of the file-paper holes
%\vspace*{84.5mm}\noindent $\bigodot$  %130mm-40mm = 90mm
%\vspace{76.2mm}\noindent $\bigodot$  %2*40=80 (holes 80 mm apart)
% now return to begining again
%\vspace{-160mm}
%---------------------

\vspace{-1.8cm}
{\ } \hfill
\framebox{\rule[-10mm]{0cm}{3.3cm}\hspace{2.2cm}Patient
                        label\hspace{2.2cm}}\hspace{7mm} %2mm

\vspace{-3.6cm}  %-1.8cm -2.5
\noindent\hspace{3.1cm}{\LARGE ANAESTHETIC SHEET}

\vspace{3mm}
\hspace*{2.5cm} Theatre 1 -- City Hospital, Nottingham, UK.%
     \footnote{Data from Datex series-II monitors
```

```
      \hfill  {\sc anaesthesia record system-11h} \copyright\  RWD Nickalls,
      1994--1997 \hspace{5mm} }
%%-----------------------------------------
\vspace{5mm}
\hspace*{1.54cm}\vbox{
\begin{tabular}{|ll|}
\hline
{\sc Date:} \rule{0pt}{12pt}  & \today   \\
{\sc Operation:}  &   \\
{\sc Anaesthetists:} & RWD Nickalls {\it et al \hspace{2cm}} \\
{\sc Surgeons:} &  \\
\hline
\end{tabular}
}

%-----------------------
\vspace{-2mm}
\noindent \hspace{1.5mm}
   \input{plot-bp.pic}
%-----------------------

%-----------------------
\vspace{-16mm}
\noindent\hspace{1.5mm}
      \input{plot-sat.pic}
%-----------------------

%-----------------------
\vspace{-16mm}  %-20
\noindent\hspace{1.5mm}
    \input{plot-fo2.pic}
%-----------------------

%-----------------------
\vspace{-16mm}
\noindent\hspace{1.5mm}
   \input{plot-co2.pic}
%-----------------------

%-----------------------
\vspace{-16mm}
\noindent\hspace{1.5mm}
    \input{plot-tv.pic}
%%--------------------

%% now put on the right axis for Resp rate (0, 5,10,15,20).
%% *need* all the sapces between lines here

 \vspace{-37mm} \noindent\hspace{188.5mm} 20 $\bullet$
```

```
 \vspace{1mm}\noindent\hspace{188.5mm} {15}

 \vspace{0.8mm}\noindent\hspace{188.5mm} {10}

 \vspace{1mm}\noindent\hspace{189mm} {5}

 \vspace{1mm}\noindent\hspace{189mm} {0}
%--------

\vspace{-4mm}
%------------------------
\noindent\hspace{1.5mm}
     \input{plot-vap.pic}
%------------------------
\end{document}
```

Chapter 21

Typesetting the drug file

drug-11h.pb
prt1drug.tex
prt2drug.tex

ch-pdrug.tex

These two TEX files print out the drug file (.drg file) in either one column (prt1drug.tex), or two columns (prt2drug.tex). One column is the default, but if the .drg file is too big to fit on to one page using one column, then we use a two-column format.

21.1 drug-11h.pb

```
REM drug-11h.pb
REM May 18th 1997
REM RWD Nickalls
REM ----------------------
REM prints either SINGLE or DOUBLE cols depending on file length
REM this uses COMMAND$ to  get the filename
REM prog to print out the drug file
REM -------------------------
REM first determine the number of lines in the file
REM pass the filename  nnnnnn.drg from the main prog (anes-20n.bas)
REM -------------------------
OPEN COMMAND$ FOR INPUT AS #1
n = 0
CLS
DO WHILE NOT EOF(1)
     INPUT #1, dataline$
     n = n + 1
     REM PRINT "n= "; n, dataline$
LOOP
```

```
CLOSE 1
REM -------------------------
SHELL "chdir c:\qb45\rs232\theatre"
REM now copy the file to the \texinput dir
SHELL "copy" + SPACE$(1) + COMMAND$ + "c:\emtex\texinput\drugfile.dat"
REM  the file prt-drug.tex processes the file "drugfile.dat"
REM now change to \texinput dir and process the file
SHELL "chdir c:\emtex\texinput"
IF n - 2 < 46 THEN
     REM print in single col
     SHELL "latex prt1drug.tex"
     REM now print to DeskJet
     SHELL "prthplj /od+ prt1drug.dvi prn"
   ELSE
     REM print using 2 columns as usual
     SHELL "latex prt2drug.tex"
     REM now print to DeskJet
     SHELL "prthplj /od+ prt2drug.dvi prn"
END IF
REM now delete the file
SHELL "del drugfile.dat"
REM now return to the theatre dir
SHELL "chdir c:\qb45\rs232\theatre"
REM --------------------------------
END
```

21.2 prt1drug.tex

```
%% PRT1DRUG.TEX prints out the DRUG list in *one* column
%% using a SINGLE column (for short files)
%% called from DRUG-11g.BAS/exe  from the /theatre/ dir
%%------------------
%% February 23, 1997
%%------------------

%% note this is LaTeX 2.09  format
\documentstyle[a4,miscrwdn]{article}

\voffset -2.2cm
\oddsidemargin 0cm
\textwidth 16cm
\textheight 26cm

\begin{document}
%---------------------------------
% now draw the marks of the file-paper holes
%\vspace*{84.5mm}\noindent $\bigodot$  %130mm-40mm = 90mm
%\vspace{76.2mm}\noindent $\bigodot$  %2*40=80 (holes 80 mm apart)
% now return to begining again
```

```
%\vspace{-160mm}
%---------------------------------

\vspace{-1.8cm}
{\ } \hfill
\framebox{\rule[-10mm]{0cm}{3.3cm}\hspace{2.2cm}Patient
                        label\hspace{2.2cm}}\hspace{-0.5cm} %2mm

\vspace{-3.6cm}
\noindent\hspace{1.1cm}{\LARGE ANAESTHETIC SHEET}

\vspace{3mm}
\hspace*{1cm} Theatre 1 -- City Hospital, Nottingham, UK.%
     \footnote{  {\sc anaesthesia record system-11h}
               \copyright\  RWD Nickalls, 1994--1997. }

\vspace{5mm}
\hspace*{-1cm}\vbox{
\begin{tabular}{|ll|}
\hline
{\sc Date:} \rule{0pt}{12pt}  & \today   \\
{\sc Operation:}  &   \\
{\sc Anaesthetists:} & RWD Nickalls {\it et al \hspace{2cm}} \\
{\sc Surgeons:} &  \\
\hline
\end{tabular}
}
%-------------------------------
\vspace{4mm}

\begin{center}
   { \large {DRUG \& FLUID BALANCE}}

   \vspace{4mm}
   \begin{minipage}{9cm}
        \input{drugfile.dat}
    \end{minipage}
\end{center}
%%--------------------
\end{document}
```

21.3 prt2drug.tex

```
%% PRT2DRUG.TEX  prints out the DRUG list in two columns
%% using TWO columns
%%--------------------
%% February 23, 1997
%%--------------------

%% note this is LaTeX 2.09  format
\documentstyle[a4,multicols,miscrwdn]{article}

\voffset -2.2cm
\oddsidemargin 0cm
\textwidth 16cm
\textheight 26cm

\begin{document}
%---------------------------------
% now draw the marks of the file-paper holes
% (holes 80 mm apart)
%\vspace*{84.5mm}\noindent $\bigodot$
%\vspace{76.2mm}\noindent $\bigodot$
% now return to begining again
%\vspace{-160mm}
%---------------------

\vspace{-1.8cm}
{\ } \hfill
\framebox{\rule[-10mm]{0cm}{3.3cm}\hspace{2.2cm}Patient
                        label\hspace{2.2cm}}\hspace{-0.5cm} %2mm

\vspace{-3.6cm}
\noindent\hspace{1.1cm}{\LARGE ANAESTHETIC SHEET}

\vspace{3mm}
\hspace*{1cm} Theatre 1 -- City Hospital, Nottingham, UK.%
     \footnote{  {\sc anaesthesia record system-11h}
                 \copyright\  RWD Nickalls, 1994--1997.}
%%---------------------------------------
\vspace{5mm}
\hspace*{-1cm}\vbox{
\begin{tabular}{|ll|}
\hline
{\sc Date:} \rule{0pt}{12pt}  & \today   \\
{\sc Operation:}  &   \\
{\sc Anaesthetists:} & RWD Nickalls {\it et al \hspace{2cm}} \\
{\sc Surgeons:} &  \\
\hline
\end{tabular}
}
```

```
%------------------------------
\vspace{1cm}

\begin{center}
    \large {DRUG \& FLUID BALANCE}
\end{center}

\vspace{1cm}

\begin{multicols}{2}
\columnseprule = 1pt
    \input{drugfile.dat}
\end{multicols}
%%--------------------
\end{document}
```

Part VII

Appendix

Appendix A

List of files in `anes-5a.doc` directory

```
A1-LABEL TXT         5,289 20/06/01   19:38
A5-ANES2 TEX        24,891 02/05/01   18:07
A5-BOOK  TEX         6,582 21/06/01    8:41
A5-BOOK  DVI       772,724 21/06/01    8:26
A5-DRAW  TEX         9,473 23/05/01    4:03
A5-KEYF  TEX        22,762 23/05/01    4:10
A5-LIB   TEX        17,788 23/05/01    4:08
A5-PLOT  TEX        36,090 23/05/01    4:15
A5MAIN   M           7,814 09/06/01   15:06
A5MAIN   PIC         6,941 09/06/01   15:16
A5OVER   M           3,621 11/06/01   21:32
A5OVER   PIC         4,191 11/06/01   21:40
A5SCREEN M           5,218 17/06/01   14:31
A5SCREEN PIC         5,196 17/06/01   14:36
ANGRANT  TEX         8,874 25/05/01   13:53
AS3DATA  DAT       347,927 17/11/99   13:06
BIN      TEX         2,195 06/05/01   12:04
BOOKTOC  TEX        11,481 12/06/01    6:23
BPD      DAT           310 10/05/01    5:53
BPS      DAT           311 10/05/01    5:57
CH-DATA  TEX        10,139 20/06/01   17:38
CH-DPFK  TEX         8,872 19/06/01    4:21
CH-DPIN  TEX        15,672 20/06/01    9:07
CH-DPLOO TEX        12,168 18/06/01   18:17
CH-DPMM  TEX        16,421 18/06/01   14:11
CH-DXMOD TEX        33,625 20/06/01    9:13
CH-DXMON TEX        37,657 17/06/01   11:23
CH-HIST  TEX        11,850 10/06/01   20:34
CH-MAIN  TEX        15,146 19/06/01    5:25
CH-MENU  TEX        23,904 18/06/01    8:08
CH-PAN2  TEX         6,711 10/06/01   20:35
CH-PBAT  TEX           223 10/06/01   17:05
```

```
CH-PDRUG TEX         5,570 21/06/01    7:04
CH-PEXAM TEX         1,430 21/06/01    6:28
CH-PGNU  TEX         3,101 21/06/01    8:10
CH-PLOT  TEX        24,465 20/06/01   19:48
CH-PRINT TEX        11,403 21/06/01    8:02
CH-SPLAB TEX        27,759 10/06/01   20:38
CH-SUBS  TEX       110,359 19/06/01    5:49
CH-SYST  TEX         9,777 17/06/01   17:07
CH-TUG98 TEX         9,228 13/06/01   12:10
CITY     BAS         1,917 13/06/01   17:03
CITY     DAT         3,889 13/06/01   17:04
CITYTAB  TEX         1,789 14/06/01    6:17
CO2      DAT           346 10/05/01    5:58
COLOPHON TEX         4,882 21/06/01    8:20
COVER5A  TEX           627 13/06/01   12:26
CVP      DAT           310 10/05/01    5:58
DATA     DAT         2,116 13/05/01    9:22
DATAD01  TEX         2,111 14/05/01    7:14
DATADRG  TEX           994 13/05/01    9:26
DATAG01  TEX         3,615 14/05/01    7:16
DATALAST TEX         1,458 14/05/01    7:14
DIRLIST  TXT         4,396 11/06/01    7:35
DRUG-11H BAS         1,432 20/06/01   15:54
FILESLST TEX         4,452 21/06/01    8:45
FIN2O    DAT           323 10/05/01    6:00
FIO2     DAT           322 10/05/01    6:00
FULLCODE BAS         1,184 13/05/01    9:12
GNU-PIC  TEX         2,066 21/06/01    7:28
HELP     DAT         3,511 22/10/98   18:00
HRECG    DAT           323 10/05/01    5:59
HROXIM   DAT           324 10/05/01    6:01
IDEAS    TEX           534 02/05/01   15:38
LASTHOUR DAT         8,775 24/11/99   22:51
LOOP12HK TEX         6,776 05/05/01   18:18
MAC      DAT           346 10/05/01    6:01
MENU-501 PCX        10,386 05/05/01   12:59
MENU-502 PCX        10,829 05/05/01   12:59
MENU-503 PCX        12,027 05/05/01   12:59
MENU-504 PCX        12,427 05/05/01   12:59
MPIC21F2 EXE       154,259 16/05/01   16:57
PLOT-BP  GNU         1,148 09/05/01   18:01
PLOT-CO2 GNU           793 09/05/01   18:01
PLOT-FO2 GNU           872 09/05/01   17:58
PLOT-SAT GNU           835 09/05/01   17:59
PLOT-SAT PIC         1,160 02/06/01   18:35
PLOT-TV  GNU           861 09/05/01   17:59
PLOT-VAP GNU           913 09/05/01   18:00
PLOTAN2E BAS        22,473 21/07/98   19:39
PRT2DRUG TEX         1,707 10/05/01    5:24
RESTORE  BAT           201 12/06/01   18:20
```

```
RN-BP     PIC        23,498 19/07/98   13:51
RN-CO2    PIC         8,292 19/07/98   13:52
RN-FIG1   TEX         3,049 06/05/01   15:08
RN-FO2    PIC        14,824 19/07/98   13:51
RN-SAT    PIC         7,057 19/07/98   13:51
RN-TV     PIC        12,437 19/07/98   13:53
RN-VAP    PIC        10,233 19/07/98   13:53
ROYALCOL  TEX         6,779 02/05/01   11:54
RR        DAT           331 10/05/01    6:02
SAT       DAT           321 10/05/01    6:02
TEST-PIC  TEX         1,358 05/05/01   13:27
TUG98RPT  TEX         9,184 06/05/01   15:01
TV        DAT           333 10/05/01    6:02
VAPIN     DAT           343 10/05/01    6:03
VAPOUT    DAT           343 10/05/01    6:03
```

Appendix B

Colophon (paper version)

This booklet was typeset in Computer Modern Roman 10-point font using LaTeX 2ε, and printed using a HP DeskJet-520 (300 dpi) on A4 paper. The booklet structure for the original printed version (2001) was defined using the following LaTeX 2ε code.

```
\documentclass[a4paper]{book}
%
%\usepackage[frame]{crop}  %% for paper edge
%\usepackage{showfram}  %% for text margins
\usepackage{fancyvrb}    % for Verbatim
%%---------packages for TUG98 article-----------
\usepackage{latexsym} % for the \Box symbol for TUG98 figure \Box symbol
\usepackage{multicol}
\usepackage{miscrwdn} %% needed for the \anglebracket{} macro for
                      %% SpaceLabs and Datex
%-------------------
\usepackage{pictexwd}  % for \PiCTeX\ symbol
%\usepackage{amssymb}   % for symbols in Table 1
%
\setlength{\topmargin}{-0.5cm}
\setlength{\textwidth}{16cm}
\setlength{\textheight}{23.5cm}
\oddsidemargin=0.46cm  % = 21-2-3-16 - 2.54
\evensidemargin=-\oddsidemargin
\hoffset=-3.5mm
%%------ mathsPIC  macros -----------
\newcommand{\mathsPIC}{\textsf{mathsPIC}}
\newcommand{\MathsPIC}{\textsf{MathsPIC}}
\newcommand{\gap}{\protect\texttt{\char'40}\protect} %% the <space> symbol
\newcommand{\bs}{$\backslash$}

%%----------dunhill fonts for rwdn logo-------------
\newfont{\dunfont}{cmdunh10}
\newfont{\largedunfont}{cmdunh10 scaled\magstep2}
\newfont{\Largedunfont}{cmdunh10 scaled\magstep3}
```

```
\newfont{\LARGEdunfont}{cmdunh10 scaled\magstep4}

%%------------document macros for page refs--------
\newcommand{\page}[1]{page~\pageref{#1}}
\newcommand{\p}[1]{p.~\pageref{#1}}    %% a short p.55 format
\newcommand{\chaptercode}[1]{Chapter~\ref{#1}}
\newcommand{\figurecode}[1]{Figure~\ref{#1}}
\newcommand{\tablecode}[1]{Table~\ref{#1}}

%%============================================================
\begin{document}

\frontmatter
\pagenumbering{roman}
%----------------------
\thispagestyle{empty}    %% put *after* \maketitle is used
%% cover   printed separately
\vspace*{2cm}
\input{cover5a.tex}

\cleardoublepage
%%------------------
%% half-title page  (this is page i)

\vspace*{3cm}

\vspace{3cm}
\begin{center}
%\begin{minipage}{8cm}
\textsl{\Large The single biggest problem we face is that of
 visualisation}
%\begin{flushright}

\vspace{2\baselineskip}
{\large \hspace{3cm} Richard P.\ Feynman (1918--1988)
   }\footnote{The Mathematical Gazette (1996); \underline{80}, 267.}
%\end{flushright}
%\end{minipage}
\end{center}

%%-----------------------------
\cleardoublepage
%% title page

\vspace*{3cm}

\begin{center}
\Huge{Anaesthetic Record System 5a}
\end{center}
```

```
\vspace{3cm}

\begin{center}
\large
{\Large Richard W.\ D.\ Nickalls},\\
 {\ } \\
 Department of Anaesthesia,\\
 City Hospital, Nottingham, UK.\\
 {\ }\\
 \textit{dicknickalls@compuserve.com}\\
 Tel: +44--(0)115--9691169 Extn 45637\\
 Fax: +44--(0)115--9627713\\
\end{center}

\vspace{4cm}
\begin{center}
%% use Dunfont 10 pt for logo
{\dunfont \framebox{r}\framebox{w}\framebox{d}\framebox{n}}\\
{\ }\\
June 2001
\end{center}
%-------------------------
\cleardoublepage

\tableofcontents

\cleardoublepage

%---------------------
%%\section{Contributors}
\cleardoublepage

%% ? include a half-page colour screen shot

\section{Preface}
\cleardoublepage
\mainmatter
%---------------

\part{Background}
%\setcounter{chapter}{0} \setcounter{page}{1}   %% odd page

    \include{ch-hist}   % introduction & info
    \include{ch-tug98}  % Poland talk
    \include{ch-dxmon}    % Datex AS/3 monitor serial interface
    \include{ch-syst}   % system overview

%-----------------
\part{The front-end \& menus}
```

```
   \include{ch-menu}    % menu batch file
%-----------------
\part{The data access and display program}

    \include{ch-dpin}    % data program introduction
    \include{ch-dpmm}    % main module
    \include{ch-dploo}    % Loop one etc
    \include{ch-dpfk}    % F-keys
    \include{ch-main}    % main module - code
    \include{ch-subs}    % all other SUBS except Datex
    \include{ch-DXmod}    % Datex MODule

%-----------------
\part{Data storage and files}

   \include{ch-data}

%-----------------
\part{The printer program}

   \include{ch-print}    %% introduction
   \include{ch-pbat}    %% printall.bat
   \include{ch-plot}  %% plotan3a.pb  program
   \include{ch-pexam}
   \include{ch-pgnu}  %% GNUplot files
   \include{ch-pan2} % prtanes2.tex 2e and 2.09  versions
   \include{ch-pdrug} %% includes drug-11h.pb and both 1 and 2 drg files

%-------------------------------

\part{Appendix}
\appendix
     \include{colophon}
     \include{all-files.tex} %% = a list of all file-names used in the book
%-------------
%\fi
 %\include{part.tex} %% typesets the three parts
%%------------
\end{document}
```

Appendix C

Colophon (PDF version)

This PDF version was generated using PDFLaTeX defined using the following code.

```
\documentclass[a4paper,oneside]{book}

\usepackage{fancyhdr}
\usepackage{oldstyle} %% page nos
\usepackage{fancyvrb}    % for Verbatim
%%---------packages for Poland article-----------
\usepackage{latexsym} % for the \Box symbol for TUG98 figure \Box symbol
\usepackage{multicol}
\usepackage{graphicx}
\usepackage{mathptmx} %% times amd maths
\usepackage{miscrwdn} %% needed for the \anglebracket{} macro for
\usepackage{ifpdf}                    %% SpaceLabs and Datex
%-------------------
\usepackage{mathspic}  % for \PiCTeX\ symbol

\oddsidemargin=0.5cm  % to allow room for code extending to RHS

%%-------------------------
\ifpdf
   \usepackage[a4,pdftex]{crop}         %% Forces a4 size with PDF
   %\usepackage[cross,a4,pdftex]{crop}  %% show PAPER edge with cross
   % \usepackage[frame,a4,pdftex]{crop} %% show PAPER edge with line
   %% test on cover with  crop frame to show paper edge
   \usepackage[verbose]{microtype}      %% pdftex  nice margin alignment
%%------------hyperref-----------------
%% see ref = Guide to Latex  (Kopka) p 251--261
%% make hyperlinks etc------------
   \usepackage{hyperref}
   \hypersetup{%%% pdftex,  % not if using \ifpdf
        debug=true,
        pdftoolbar=true,  pdfmenubar=true,
        %backref=true,  pagebackref=true,  %% only with bibliography
```

```
        plainpages=false,
        breaklinks=true,
        linktocpage=true,
        % pdfpagemode=None,    %% bookmark window closed
        bookmarksopen=true,
        bookmarksopenlevel=0,  %% (book=0, chap=1, article=2)
        bookmarksnumbered=true,
        bookmarkstype=toc,
        pdfwindowui=true,
        hyperindex, hyperfigures=true, hyperfootnotes,
        colorlinks, urlcolor=blue,  linkcolor=blue,
        %% in the pdfinfo stuff use just one  comma /ONLY/ at end of line
        %% or use {} if want to use commas
        %% for pdfstartview see p 253 Guide to latex (Kopka)
        pdfstartview=FitH,   %%FitH, FitV ,
        pdfview=XYZ,
        pdfcenterwindow=true,
        pdfnewwindow=true,
        pdfpagelayout=SinglePage, %TwoColumnRight,
        pdfpagelabels=true,
        %----------------------
        pdftitle={Anaesthesia Record System 5a},
        pdfauthor={RWD Nickalls 2009},
        pdfsubject={Anaesthesia},
        pdfkeywords={Anaesthesia, anesthesia, computing, records},
        %% pdfcreator=latex,
      }
\else
\fi
%--------------
%%------ mathsPIC  macros -----------
\newcommand{\gap}{\protect\texttt{\char'40}\protect} %% <space> symbol
\newcommand{\bs}{$\backslash$}

%%----------dunhill fonts for rwdn logo-------------
\newfont{\dunfont}{cmdunh10}
\newfont{\largedunfont}{cmdunh10 scaled\magstep2}
\newfont{\Largedunfont}{cmdunh10 scaled\magstep3}
\newfont{\LARGEdunfont}{cmdunh10 scaled\magstep4}

%%------------document macros for page refs--------
\newcommand{\page}[1]{page~\pageref{#1}}
\newcommand{\p}[1]{p.~\pageref{#1}}    %% a short p.55 format
\newcommand{\chaptercode}[1]{Chapter~\ref{#1}}
\newcommand{\figurecode}[1]{Figure~\ref{#1}}
\newcommand{\tablecode}[1]{Table~\ref{#1}}
%-------------------------------------------------

%%============================================================
\begin{document}
```

```
\frontmatter
\pagenumbering{roman}
%------cover----------------
\thispagestyle{empty}
%% cover   printed separately
\vspace*{2cm}
%%%%%\input{cover5a.tex}  %% cover for "Background" version only
\input{coverout-cd.tex} %% for 'hard' cover book version (different font)
\cleardoublepage

%%-----------------
%% half-title page  (this is page i)
\vspace*{3cm}

\vspace{3cm}
\begin{center}
\textsl{\Large The single biggest problem we face is that of
 visualisation}

\vspace{2\baselineskip}
{\large \hspace{3cm} Richard P.\ Feynman (1918--1988)
   }\footnote{The Mathematical Gazette (1996); \underline{80}, 267.}
\end{center}

%%-----------------------------
\cleardoublepage
%% title page

\vspace*{3cm}

\begin{center}
\Huge{Anaesthesia Record System 5a}
\end{center}

\pdfbookmark[0]{titlepage}{titlepage}

\vspace{3cm}

\begin{center}
\large
{\Large Richard W.\ D.\ Nickalls},\\
 {\ } \\
 Department of Anaesthesia,\\
Nottingham University Hospitals,\\
 City Hospital Campus,\\
 Nottingham, UK.\\
 {\ }\\
 \textit{dick@nickalls.org}\\
 \textsf{www.nickalls.org}
```

```
 \end{center}

\vspace{4cm}
\begin{center}
%% use Dunfont 10 pt for logo
{\dunfont \framebox{r}\framebox{w}\framebox{d}\framebox{n}}\\
{\ }\\
August 2001
\end{center}
 %-------------------------

\newpage
\vspace*{10.7cm}
{
\begin{center}
\large
Anaesthesia Record System 5a\\
 Copyright\ \ {\copyright}\ \   RWD Nickalls\ \ 1994---2001
\end{center}
 }

\input{ch-preface}

%-------------------------
\tableofcontents
\addcontentsline{toc}{chapter}{Contents}

%--------------------
%%\section{Contributors}

%\section{Preface}

%%--------------
\pagestyle{fancy}
\fancyfoot{}  %% clears existing foot stuff
\fancyhead{}  %% clears existing head stuff
\fancyhead[LE]{{\oldstyle{\thepage}}\hspace{5mm}\small{RWD Nickalls}}
\fancyhead[RO]{\small{RWD Nickalls}\hspace{5mm}\oldstyle{\thepage}}
\fancyhead[RE, LO]{\small\leftmark}

\mainmatter

\part{Screenshots}
\input{screenshots.tex}
%---------------
\part{Background}
    \include{ch-hist}   % introduction & info
    \include{ch-tug98}  % Poland talk
    \include{ch-dxmon}    % Datex AS/3 monitor serial interface
     \include{ch-syst}   % system overview
```

```
%-----------------
\part{The front-end \& menus}
   \include{ch-menu}    % menu batch file
%-----------------
\part{The data program}
    \include{ch-dpin}    % data program introduction
    \include{ch-dpmm}    % main module
    \include{ch-dploo}    % Loop one etc
    \include{ch-dpfk}    % F-keys
    \include{ch-main}    % main module - code
    \include{ch-subs}    % all other SUBS except Datex
    \include{ch-DXmod}    % Datex MODule
%-----------------
\part{Data storage}
   \include{ch-data}
\part{The printer program}
   \include{ch-print}    %% introduction
   \include{ch-pbat}    %% printall.bat
   \include{ch-plot}  %% plotan3a.pb  program
   \include{ch-pexam}
   \include{ch-pgnu}  %% GNUplot files
   \include{ch-pan2} % prtanes2.tex 2e and 2.09  versions
   \include{ch-pdrug} %% includes drug-11h.pb and both 1 and 2 drg files
\part{Appendix}
\appendix
   \include{colophon-A}  %% paper version
   \include{colophon-B}  %% pdf version
   \include{fileslst} %% = a list of all file-names used in the book
%%------------
\end{document}
```