Software controlled proportional valve for NIBP measurements

Heesakkers, Jeroen

Award date:
1997

Dräger

Technische Universiteit tu/e Eindhoven

Software controlled proportional valve for NIBP measurements

Author: J. Heesakkers

Date: August 1997

Place: Best

Dräger

Technische Universiteit tu/e Eindhoven

Business Unit Monitoring,
Research and Development.

Faculty of Electrical Engineering,
Department of Measurement and Control Systems (MBS),
Section of Medical Electrical Engineering

Software controlled proportional valve for NIBP measurements

Author: Jeroen Heesakkers
graduate University of Technology Eindhoven

Written by order of: prof. dr. ir. P.P.J. van den Bosch
dr. ir. P.J.M. Cluitmans
University of Technology Eindhoven

A.W. Bodbijl
Dräger

Date: August 1997

Place: Best

Summary

The blood pressure in the human body can be determined by pressurizing a cuff that is wrapped around a persons arm and measuring pressure pulsations inside the cuff, while the cuff is deflated. The oscillometric method of Non Invasive Blood Pressure (NIBP) measurement uses this concept for the determination of systolic-, diastolic- and mean blood pressure.
The equipment that is engineered by Dräger is able to perform automated NIBP measurement by pressing one button, thereby facilitating the work of physicians and nursing staff. The equipment uses a pump to inflate a cuff to an initial cuff pressure and uses two deflate valves to deflate the cuff pressure. The cuff pressure is measured with a pressure transducer.
Because the valves that deflate the cuff have only two positions; open or shut, the cuff is deflated stepwise (steps of approximately 8 mmHg are taken). A deflate valve is opened shortly to deflate the cuff to a preferred pressure level. The valve is then closed and the pressure pulsations in the cuff at this pressure level are measured. The amplitude of the pulsations ranges (for most patients) from 0 to approximately 5 mmHg, depending on the cuff pressure. After a number of pulsations have been measured, the cuff is deflated again to the next pressure level. The amplitude of the measured pulses at each level is used to calculate blood pressure.
Lately, a new generation of valves has become available; proportional valves, which offer current (or voltage) controllable flow restriction at a cheap price.

The purpose of this research was to investigate the usability of a proportional valve for NIBP measurement. The possibility of linear deflation with a proportional valve was investigated. For this investigation, the valve was integrated in a test setup with a Dialog 2000 monitor device and a personal computer. For this setup a data acquisition system was developed. Using the data acquisition system, a digital feed forward PI controller was implemented that performs linear cuff deflation by using the proportional valve. The data acquisition and control system was developed according to the strategies for real-time system specification from Hatley & Pirbhai and the Dräger coding standard.

The proportional valve is a non linear component, which has a hysteresis of 15%, a dead zone of approximately 50% of the control voltage range and an undefined point at which the plunger first reacts to the applied voltage (end of the dead zone). Despite these properties, measurements show that with the current setup, the cuff pressure can be controlled within 1 mmHg accuracy. The maximum accuracy is limited by the sample rate of the digital controller and the resolution of the analog-to-digital and digital-to-analog conversions.

The advantages the new method of deflation offers are: estimated 25% shorter average measurement time and a more comfortable measurement for the patient.
Investigations remain to be done on how the measurement time can be shortened even more in relation to the patients heart rate. Also the integration of the data acquisition and control system in a monitor device remains subject of future research.

Preface

This report was written in the scope of the completion of my study at the University of Technology in Eindhoven, faculty of Electrical Engineering. The Dräger company offered me the possibility to do a research at the Research and Development department in Best, for which I am very grateful.

The assignment was to investigate a new method for NIBP (non invasive blood pressure) measurement. The new method required a continuous deflation of the cuff that is used for NIBP measurement. In order to be able to measure NIBP with this new method, high demands are made concerning artifact recognition and suppression.
At first the research concerned the development of a device that controls an analog pneumatic valve and the development of an algorithm that can deflate the cuff with constant pressure, regardless the cuff volume.
After this, a new software algorithm that can detect and measure pulses was planned to be developed in order to determine blood pressure.

During the research I also wrote the document 'Structured analysis of a static cuff pressure controlling application', which can be read in addition to Chapter 4 and 5. Furthermore I wrote the manual for the data acquisition and control system.

I want to thank the people at the Research and Development department of Dräger in Best for their kind help and time they took to answer my questions. Especially, I want to thank my supervisor, Aad Bodbijl for his advice, critical remarks and help during the assignment. I also want to thank John Wijers for his 'real-time' advice and for introducing me to the C-world.
I also want to thank the people at the university at the section of Medical Electrical Engineering for their critical remarks and questions at the time of the intermediate presentations. Especially I want to thank my supervisor dhr. Cluitmans for his support during the application at Dräger and for the involvement he showed to make this assignment successful. I also use this opportunity to send kind regards to professor van den Bosch who took the final responsibility for my assignment.

Table of contents

Chapter 1

An introduction to NIBP measurement

This chapter introduces the reader to the oscillometric measurement of blood pressure. Using the Dialog 2000 as an example, the automated non invasive blood pressure measurement is discussed. Because the measurement is based on the deflation of a cuff, wrapped around a patients limb, this is modeled in a mathematical review to show the behavior of the cuff pressure under this conditions. Further, the use of open/close valves to deflate a cuff in the current application is discussed.

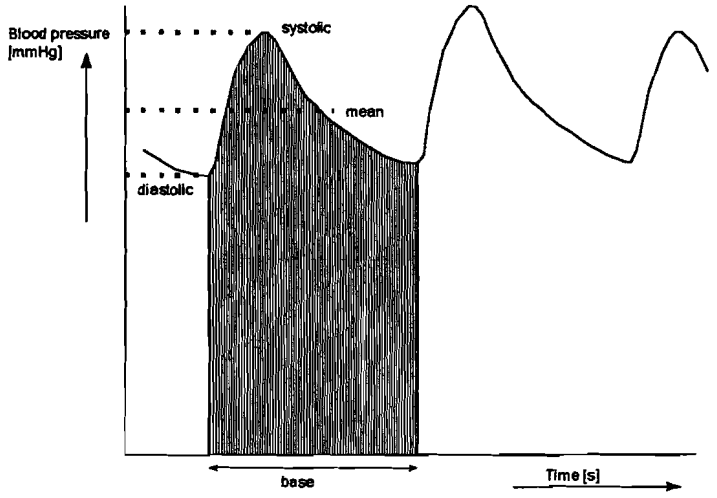
1.1 Medical parameters; NIBP

The medical parameter NIBP is the Non Invasive Blood Pressure measurement that provides a physician information about a patients condition. Non invasive means that pressure is measured without inserting a measurement device into the body.
Blood pressure exists because of the pump action of the heart. The heart fills itself with blood and contracts to pump the blood into the arteries to establish the circulation of blood in the body. The number of times the heart contracts in one minute is defined as the heart rate. When the heart contracts it forces the blood through the arteries, causing the blood pressure to rise to the systolic pressure level. After contraction, the blood pressure decreases to the diastolic level, when the heart refills itself with blood. The contraction and relaxation of the heart causes the blood pressure to have a pulse shaped fluctuation. An example of how a continuous blood pressure measurement signal could look like is shown in Graph 1.1. (It is not possible to obtain this signal with the NIBP method; usually it is obtained with invasive techniques.)

When measuring the blood pressure there are three values of interest. These are: systolic-, diastolic- and mean blood pressure, all three shown in Graph 1.1. Notice that mean pressure is not the diastolic pressure plus the arithmetic average of the difference between systolic- and diastolic pressure. Mean pressure depends upon the height and the shape of the arterial pressure wave. When the blood pressure is known at each moment in time, the mean pressure can be calculated by averaging the marked area by the width of the wave at its base. In practice, the mean pressure

level is often derived from the systolic- and diastolic levels by means of heuristic determined algorithm.

Notice that not every pulse has the same shape or amplitude, so systolic-, diastolic, and mean pressure can change with each heartbeat. When measuring blood pressure in a short span of time, it is assumed that the difference in blood pressure for subsequent pulses is not too large. The heart rate, also a variable number, determines the frequency of the pulses. For a healthy human at rest the heart rate is typical 60 beats per minute, but can increase with exertion to a maximum of approximately 200 beats per minute. These figures are only meant as an indication to the reader, because they can vary from person to person depending on physical condition and age.



Graph 1.1 Blood pressure.

When measuring NIBP the following properties of systolic- and diastolic pressure are important:

Systolic blood pressure:
When an external force, equal to the systolic pressure, compresses the vasculature that surrounds an artery, the artery is shut close, preventing the blood to flow.

Diastolic blood pressure:
When an external force, with a magnitude of the diastolic pressure, compresses the vasculature that surrounds an artery, this has no effect on the flow of blood in the artery.

Because of historical reasons, blood pressure usually is measured in millimeters mercury (mmHg), referring to the pressure of a column filled with mercury that was used to measure blood pressure in early days. However, the correct SI unit is the kilo Pascal (kPa).

Blood pressure can be measured in various ways, including the palpatory method, the flush method, oscillometric method, auscultatory method and ultrasound kinetoarteriography. The subject of this research is the oscillometric method that will be discussed in more detail in the next paragraph.

1.2 Automated oscillometric NIBP measurement

Two monitoring devices, the Parameterbox and the Dialog 2000, both engineered and built by Dräger are able to acquire a patients blood pressure with the non invasive oscillometric method. Concerning the NIBP measurement the equipment can be modeled as shown in Figure 1.1.

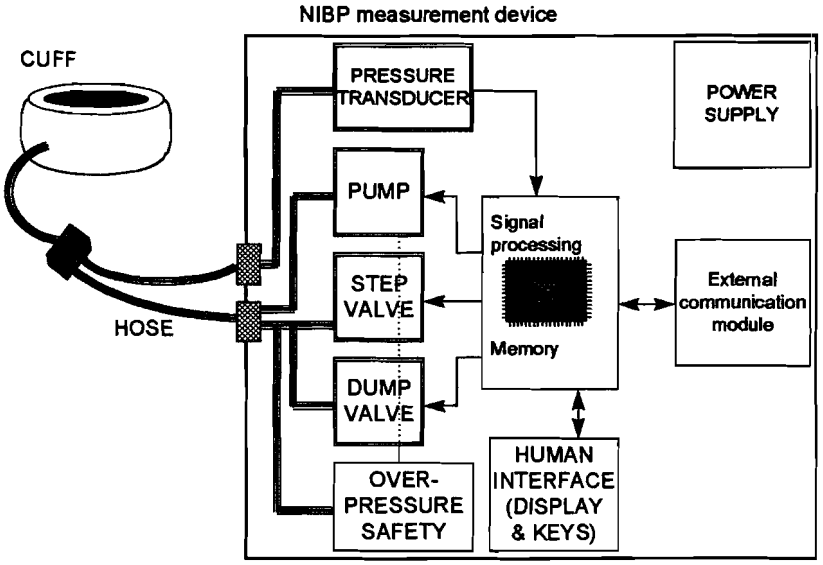
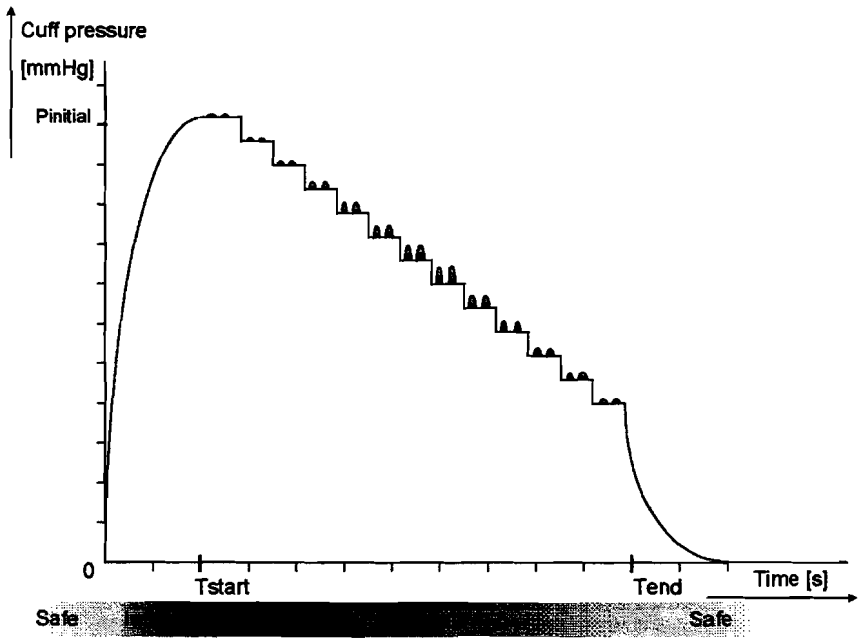


Figure 1.1 NIBP measurement equipment model.

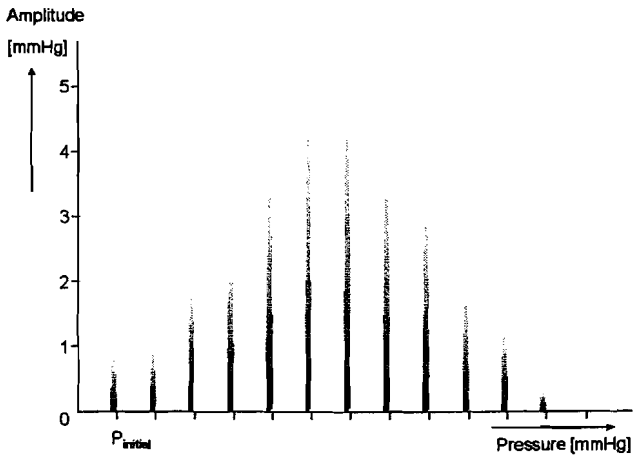
The oscillometric method of blood pressure measurement uses a cuff that is wrapped around a patients limb (often the upper arm). When the cuff is inflated, the surrounded limb and its vasculature is compressed, thus preventing the blood to flow freely. Initially the cuff is pressurized to a level above the systolic pressure; this means no blood will flow through the compressed limb. The cuff pressure is held at a constant level while pressure pulsations in the cuff are measured by the NIBP measurement device. The cuff pressure is decreased stepwise, with a step size of approximately 8 mmHg. The algorithm that deflates the cuff keeps the pressure at the set level, long enough to detect at least two consecutive pressure pulsations. The measured pulsations have to meet two artifact detection tests. If the amplitude of the pulsations differs no more than an acceptable small amount (test 1) and if the time interval between them closely matches previous time intervals (test 2), the average amplitude of the two samples and the cuff pressure is stored in a memory. If one of these two artifact detection tests fail, the cuff pressure is kept at the same level until two consecutive oscillations are detected that meet the specified requirements. In case of bad measurement conditions, when the cuff pressure is held on the same level too long, a time-out device aborts the measurement and deflates the cuff to a level below 10 mmHg.
Sampling of two consecutive matched oscillation amplitudes, averaging them, and storing the result along with the cuff pressure at each step, continues for (for example) five deflations after the step at which the oscillation average is a maximum (this depends on the algorithm that is used for blood pressure calculation).

This complete information set is used to determine systolic-, diastolic-, mean blood pressure and heart rate. The possible heart rate for a measurement ranges from 30 to 250 beats per minute. Graph 1.2 shows the cuff pressure during an ideal NIBP measurement.



Graph 1.2 Ideal NIBP measurement data.

When looking at the pressure pulsations collected during measurement, an increase and decrease of amplitude is seen (Graph 1.3). The amplitude of the pressure pulses varies from approximately 0 to 4 mmHg. The reason why the amplitude of the pulsations increases and decrease during a measurement has been investigated by various researchers [9], [20] and [24]. However, there is still no identical statement that describes the phenomenon.



Graph 1.3 Amplitude of measured pulses.

A possible explanation for the increase and decrease of the pulsation amplitude will be given. However, it should be kept in mind that this explanation can not fully cover nor replace the extensive and complex research of the earlier mentioned articles.

Consider Figure 1.2. A cuff is wrapped around a patients arm and is pressurized with air. The cuff volume can be considered constant and independent of the applied pressure. When the cuff pressure is above systolic blood pressure (Figure 1.2a) there is no blood flow in the compressed artery. Only small pressure pulsations are measured in the cuff due to the expansion of the artery when the heart contracts and tries to drive blood into the compressed artery.
When the cuff pressure is decreased below systolic level (Figure 1.2b), blood will be allowed to flow, at the moment the heart contracts and when the blood pressure is higher than the cuff pressure. The artery wall will expand to permit the blood to flow. The radial displacement of the artery wall (d) at the moment the blood flows through the artery is directly transferred to the cuff, because the muscle that surrounds the artery is flexible, but incompressible.

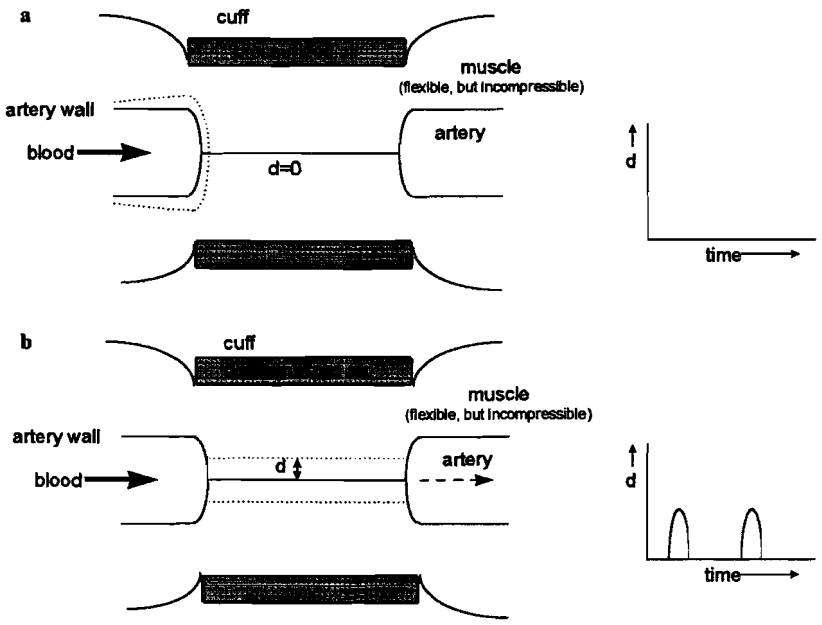


Figure 1.2a, b Model of an arm occluded by a cuff.

When the cuff pressure is decreased, the radial displacement of the artery and the time the blood is allowed to flow, both increase. The radial displacement reaches its maximum, when the cuff pressure is about the mean blood pressure level. This can be seen in Figure 1.2c. (The pressure at which maximum oscillations are measured is still investigated).

Figure 1.2d shows the situation when the cuff pressure is below the mean pressure level.

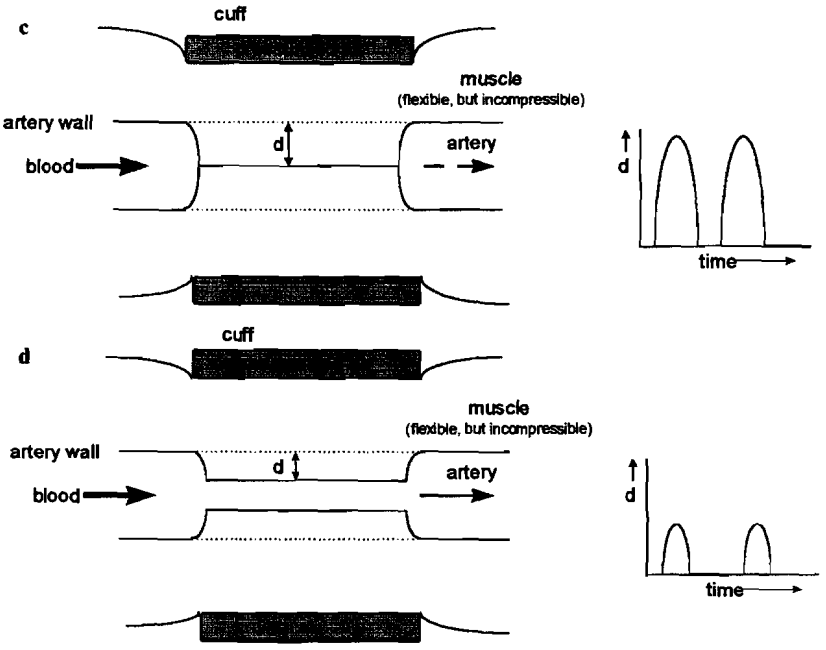


Figure 1.2c, d Model of an arm occluded by a cuff.

Because the mean blood pressure in the artery is higher than the cuff pressure, there will be a constant blood flow through the artery and therefore, the radial displacement will not be as large as in Figure 1.2c any more when the heart contracts and drives blood through the artery. Finally, when the cuff pressure is decreased below the diastolic blood pressure level, the artery is not compressed anymore and there will be no pulsations transferred to the cuff.

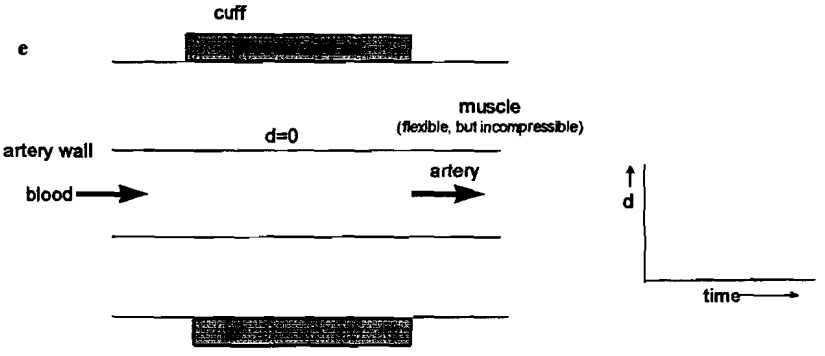


Figure 1.2e Model of an arm occluded by a cuff.

After having collected enough pulsation amplitudes, the cuff is deflated to a pressure below 10 mmHg. The faster the cuff can be deflated below this level, the faster the limb can recover from the measurement.

1.3 Cuff deflation during NIBP measurement

In order to be able to control the pressure inside a pressurized chamber by means of a valve, some aspects on the subject are investigated.

Assume a chamber of volume V_1 filled with air, at a certain pressure P_1 and temperature T_1 with an opening at one side to a second volume V_2 (Figure 1.2). The area of the orifice is A_{12} and the pressure outside the chamber is P_2. The enclosed air in the chamber has a mass of m_1 and the chamber has a certain compliance which is modeled by a flexible wall that has a pressure dependent displacement of dx [m].

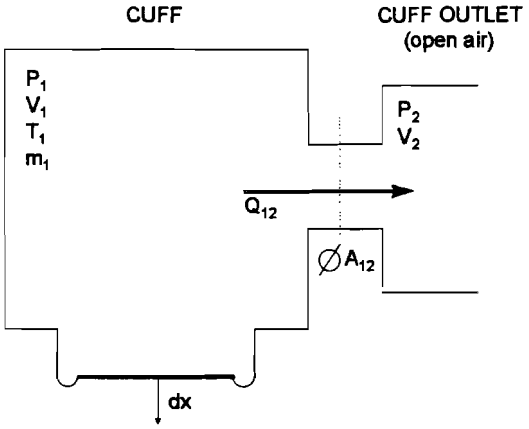


Figure 1.2 Configuration model of a cuff.

The equation of state relates pressure P [kPa], gas density ρ [kg/m^3] and temperature T [K] by the gas constant R [J/(kmol·K)]. For air this is:

$$P = \rho \cdot R_{air} \cdot T \qquad \text{(Equation 1.1)}$$

With R_{air} the specific gas constant for air, that is derived from the universal gas constant R, using the molar mass M [kg].

$$R_{air} = \frac{R}{M}$$

The gas density is defined as:

$$\rho = \frac{m}{V}$$

Where m [kg] is the mass of the enclosed air in volume V [m^3].

For the configuration in Figure 1.2 the pressure P_1 can be stated as:

$$P_1 = \frac{m_1 \cdot R_{air} \cdot T_1}{V_1} \qquad \text{(Equation 1.2)}$$

Now, assume the pressure P_1 larger than the pressure P_2, resulting in an air flow Q_{12} [m^3/s] out of the chamber. When a small displacement dx is assumed during deflation, the volume decrease is negligible. Also an isothermal expansion of the gas is assumed, which will leave the temperature of the gas unchanged.
The pressure change in time t [s] can then be described by:

$$\frac{dP_1}{dt} = \frac{R_{air} \cdot T_1}{V_1} \cdot \frac{dm_1}{dt} = \frac{P_1}{m_1} \cdot \frac{dm_1}{dt} \qquad \text{(Equation 1.3)}$$

Because air flows out of the chamber the mass of the enclosed air decreases. This is considered a negative flow. The change of mass can be calculated by the weight flow W_{12} [kg/s]:

$$\frac{dm_1}{dt} = -W_{12} = -\rho \cdot Q_{12}$$

Now the flow is defined as:

$$Q_{12} = C_d \cdot A_{12} \cdot \sqrt{\frac{2 \cdot (P_1 - P_2)}{\rho}} \qquad \text{(Equation 1.4)}$$

in this equation C_d [$(\sqrt{m})/s$] represents the discharge coefficient. The discharge coefficient depends on the properties of the gas and the physical construction of the orifice. Equation 1.4 is valid for both laminar and turbulent flow. Whether the flow is laminar or turbulent can be determined by calculating the Reynolds number Re. When the Reynolds number is calculated to be smaller than 2300 the flow is considered laminar. A turbulent flow is characterized by a Reynolds number greater than 3000. When the Reynolds number has a value between 2300 and 3000, the flow can be considered in a transition from laminar to turbulent. In equation 1.4 C_d is changed according to the condition of the flow.
By defining the flow in both laminar and turbulent condition, the change in pressure can now be written as:

$$\frac{dP_1}{dt} = -\frac{P_1}{m_1} \cdot C_d \cdot A_{12} \cdot \sqrt{2 \cdot \rho \cdot (P_1 - P_2)} \qquad \text{(Equation 1.5)}$$

When the start of deflation is assumed at the moment t_a the pressure after a lapse of time, for example at the moment t_b can be calculated by integration:

$$\int_{t_a}^{t_b} \frac{dP_1}{dt} dt = \int_{t_a}^{t_b} -\frac{P_1}{m_1} \cdot C_d \cdot A_{12} \cdot \sqrt{2 \cdot \rho \cdot (P_1 - P_2)} dt$$

This can be rewritten as:

$$\int_{t_a}^{t_b} \frac{1}{P_1} dP_1 = \int_{t_a}^{t_b} -\frac{1}{m_1} \cdot C_d \cdot A_{12} \cdot \sqrt{2 \cdot \rho \cdot (P_1 - P_2)} dt$$

when the integration on the right side is replaced by ΔDS, which can be considered a deflation speed parameter, the equation can be written as:

$$\ln P_1[t_b] - \ln P_1[t_a] = -\Delta DS$$

The pressure at moment t_b can now be calculated as:

$$\ln \frac{P_1[t_b]}{P_1[t_a]} = -\Delta DS$$

$$P_1[t_b] = P_1[t_a] \cdot e^{-\Delta DS} \qquad \text{(Equation 1.6)}$$

The calculation of integral ΔDS over a certain time period can be done using an iteration technique, where first a trial value of P_1 is needed. After calculating ΔDS the trial value is compared with the calculated value. If the difference is larger than the required accuracy, the trial value is adjusted and ΔDS is calculated again. This is repeated until an accurate value of P_1 is found.

Equation 1.6 shows that the pressure reaches a steady state point within a finite time where the pressure drop across the orifice is zero, because the integral ΔDS is time dependent. Also the equation shows an exponential decrease of pressure when a fixed restriction is used to deflate a cuff. In order to deflate a cuff by, for example, a linear line the orifice area has to be controlled.

Notice the similarities between the deflation of a pressurized cuff with a fixed restriction and the discharging of a capacitor with a resistor. The charged capacitor is discharged according to:

$$U(t) = U_{\text{initial}} \cdot e^{-\frac{t}{RC}}$$

However, the steady state condition is not reached within finite time while discharging a capacitor.

1.4 Current application; open/close valves

In the current NIBP measurement equipment, two valves are used to deflate the cuff. Both the valves are normally open. The valves close when a control voltage is applied to the coil.
Legal prescriptions demand that the measurement equipment has two valves to avoid accidents in case of a malfunction.
Because of safety reasons, the valves have relative large orifices, which allow fast deflation of a cuff in case a measurement is aborted or when the equipment is defective.
To be able to deflate different cuff sizes, the orifice of one valve is larger than the orifice off the other valve. This will be explained later.
Table 1.1 shows the time both valves need to deflate a cuff of 500mL.

initial pressure 250 [mmHg], volume 0.5 [l]	deflate time [s]	end pressure [mmHg]	valve name
valve 1	18	10	step valve
valve 2	6	0	dump valve

Table 1.1 Valve deflate times.

During NIBP measurement the cuff is deflated stepwise. The step valve is used to deflate small amounts of air (causing the pressure inside the cuff to decrease relative slow); the dump valve is used to control relative large air flows (causing relative fast pressure decrease in the cuff). Of course the pressure decrease inside the cuff, that is caused by opening a valve depends on various conditions, such as the time the valve is opened, the size of the valve restriction and the size (volume) and compliance of the cuff.

The theory of Chapter 1.3 describes the pressure decrease in a cuff when a fixed restriction is used.
According to Equation 1.6 the pressure inside a cuff can be calculated.

$$P_1[t_b] = P_1[t_a] \cdot e^{-\Delta DS} \qquad \text{(Equation 1.6)}$$

When similarities with the discharging of a capacitor are used, the pressure decrease in time can be described by:

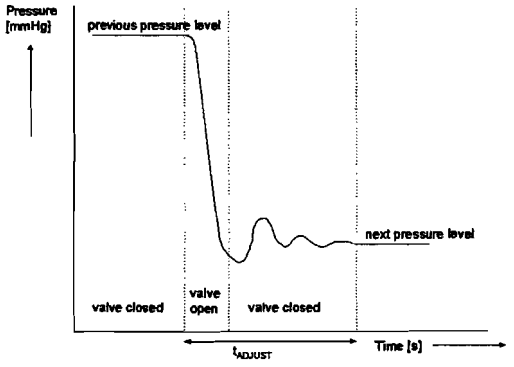
$$P(t) = P_{\text{initial}} \cdot e^{-\frac{t}{RC}} \qquad \text{(Equation 1.7)}$$

Where R models the orifice area of the valve and C models the cuff volume. $P_{initial}$ is the cuff pressure at the moment the valve is opened.
Because a small cuff volume is represented by a small C, a large R (small valve orifice) is needed to deflate such a cuff volume in small steps. For this reason the step valve has a smaller orifice than the dump valve.

When the minimum applicable time interval a valve can be opened (t_{open}) is set by physical conditions, such as the valve response time (approximately 20ms) or the sample rate of the valve controller, this results in a limited capability of setting a preferred pressure level.

Starting with an initial pressure, an algorithm selects either the dump- or the step valve and opens the valve. The pressure is measured and when the preferred pressure is reached, the valve is closed. When the pressure has decreased too much, the pump is turned on again, to inflate the cuff. When the pressure level is reached the pump is turned off. Due to the motor response time the level can again be overpressured, requiring a deflation again.
So, given a large and a small orifice and an air pump any preferred pressure level can be set. However, it is considered a drawback that setting the pressure at a preferred level takes more time when a high accuracy is required.

Another drawback of the current method is due to the deflation algorithm. The relative large orifices cause the pressure to drop sharply during a transient from one pressure level to the next pressure level. The pressure decrease is displayed in Graph 1.5



Graph 1.5 Deflation step.

When the valve is opened the pressure decreases sharply (exponential) to the new pressure level. When the new level is reached, the valve is closed again. The pressure will exhibit ringing due to the compliance and capacity of the hose and cuff. Because the amplitude of the ringing can be of the same magnitude as the patient pulses, the measurement cannot continue for the time t_{adjust} which is needed to establish the new pressure level.
Also, when the cuff is inflated, the pump causes the cuff pressure to pulsate with a relative large amplitude. Therefore, while and some time after the setting of a certain pressure level, no pressure pulsations from the monitored patient can be measured.

Chapter 2

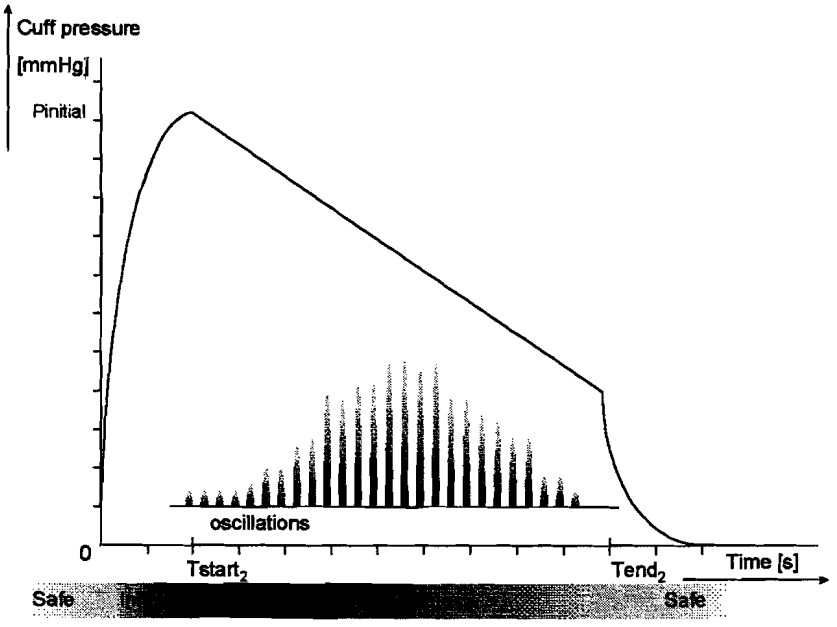
Starting point

This chapter contains the description of the assignment that Dräger Medical Electronics issued as subject for a graduate student. This assignment serves as a starting point for making the requirements for the system to be developed. The assignment is split up in four consecutive stages. The requirements for a new system are defined.

2.1 Assignment

When measuring NIBP, a cuff is wrapped around a patients arm. The cuff is inflated until an initial pressure is reached. Then the current measurement algorithm deflates the cuff pressure stepwise while measuring.
A new method for measuring requires a continuous deflation of the cuff. In order to be able to measure NIBP with this new method high demands are made, concerning artifact suppression and recognition.
At first the research shall concern the development of a device that controls an analog pneumatic valve and the development of an algorithm that can deflate the cuff with constant pressure, regardless the cuff volume.
After this has been concluded, a new software algorithm that can detect and measure pulses can be developed in order to determine blood pressure.

After this brief introduction to the assignment, more detailed information was collected. The new method of cuff deflation during NIBP measurement is best explained by Graph 2.1.

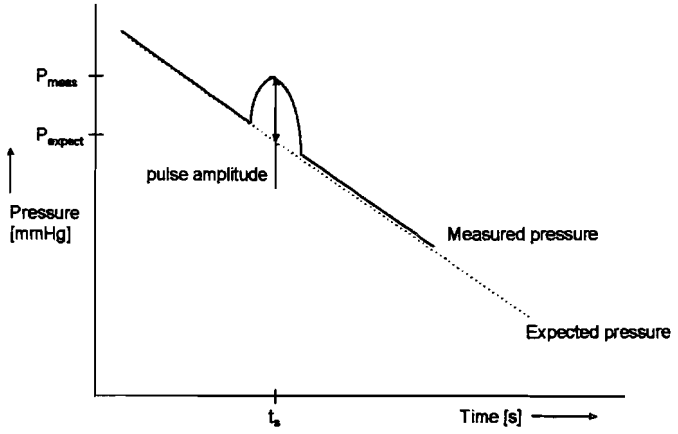


Graph 2.1 New method of deflation.

When a proportional valve is used, the cuff can be deflated continuous instead of stepwise. A proportional valve offers the possibility of controlling the pressure at any preferred level. The valve can be in any position between fully open and fully close, including these extremes. Using this advantage the inflated cuff can be deflated in a controlled way. A preferred pressure level can be set in a very short time, independent of the initial pressure. Intelligent use of a control algorithm is required because the valve response time or sample time can still disturb the performance.
Also, linear deflation causes the static cuff pressure to be available (and meaningful) at any moment. The cuff pressure does not suffer from pressure transients or settle times, when the control algorithm is using a smooth control signal. This is important when measurement time is short.
Using the proportional valve for linear deflation (Graph 2.1) offers an easy calculation of pulse pressure amplitude. The amplitude of the pressure pulsations can be calculated by subtracting the expected cuff pressure (linear line) from the measured cuff pressure, which is an easy and fast operation in a microprocessor environment.

$$A_{\text{pulse}} = P_{\text{meas}}(t_s) - P_{\text{expect}}(t_s)$$

This is illustrated in Graph 2.2.



Graph 2.2 Calculation of the measured pulse amplitude

The proportional valve has the disadvantage of being very non-linear, having a relative large hysteresis and dead zone, so a controller with feedback will be necessary to position the valve as required.
Compared to stepwise deflation, linear deflation offers two advantages.

1 Shorter measurement time.

Because the pressure signal is a continuous signal, the pulsations in the cuff can be measured continuous. This will result in a shorter measurement time.

Example:
Initial cuff pressure: 180 mmHg
End cuff pressure: 20 mmHg
Stepsize: 10 mmHg

During a measurement $\frac{(180-20)}{10} = 16$ steps are made. When each step requires approximately 500ms to establish the next pressure level, a measurement can be 8 seconds shorter, when deflating linear. With the current equipment, a measurement takes 30 seconds (average). This means an improvement of approximately 25%.

2 Comfort

The linear deflation method is more comfortable to the patient. A continuous pressure decrease in the cuff is a more pleasant feeling than a step wise deflation. Also the shorter measurement time adds to the comfort for the patient.

The exponential pressure decrease (Equation 1.7), that exists when a cuff is deflated with a fixed restriction is not preferred because of the following two reasons:

1 'Unknown' R and C

When the pressure decrease can be described by Equation 1.7, a problem exists in the determination of R and C. R is set by the restriction size of the deflate valve and C is set by the cuff and hose volume.
Determination of R should be performed for each valve in each measurement device, because the valves are produced with a relative large tolerance, that is to be expected for low cost valves. This is not realistic in a practical high volume production and maintenance environment.
Determination of C should be performed for each cuff and hose that is connected to the device. In practice this will mean measurement of C before each new blood pressure measurement, because the equipment cannot signal the change of a cuff or hose. This will slow down the measurement speed.

2 Long measurement time

A measurement with an exponential deflation curve will result in a long measurement time. This is illustrated by the following example:

Example:
The exponential deflation curve shows the largest pressure decrease at the start of a measurement. The derivative of Equation 1.7 is:

$$P'_{expect}(t) = -\frac{P_{initial}}{RC} \cdot e^{-\frac{t}{RC}} \qquad \text{Equation 2.1}$$

At the start of a measurement ($t = t_{start} = 0$) this derivative is at its maximum:

$$P'_{expect}(t_{start}) = -\frac{P_{initial}}{RC} = P'_{expect,max} \qquad \text{Equation 2.2}$$

To measure the blood pressure level with a maximum decrease in pressure of 6 mmHg/s (EN 1060-3 (European Norm)) and assuming an initial pressure level of 180 mmHg, the maximum RC constant can be calculated as:

$$RC_{max} = -\frac{P_{initial}}{P'_{expect,max}} = -\frac{180}{-6} = 30$$

Assume the cuff has to be deflated from 180 mmHg to 20 mmHg, this will take $t_{measurement}$ seconds.

$$t_{measurement} = -RC_{max} \cdot \ln\frac{P_{end}}{P_{initial}} = -30 \cdot \ln\frac{20}{180} = 66s$$

This measurement time is unacceptable long and will be no improvement over the current method, which takes 30 to 35 seconds (average) for adult patients.
Also this method only works for one cuff size. If this method is to be applied for different cuff sizes, a range of valve restrictions has to be available.

The example shows that it is useful to control the valve restriction. For the reason of easy pulse calculation, the valve was controlled to obtain the linear pressure decrease.

2.2 Partitioning of the assignment

It was agreed to partition the assignment into four stages, where each stage completes a certain part of the development.

Stage 1:

Research to investigate the possibility to control the air pressure when deflating a cuff, using a proportional valve and a microprocessor based measurement device. A proportional valve offers the possibility to control air flow continuous between a minimum and maximum value and thereby controlling the pressure inside the deflated volume.
This requires a data acquisition system that can collect measurement data and send information to control an analog valve. Hardware and software has to be developed for a data acquisition system.

Stage 2:

It is required to know how the valve should be controlled in order to deflate cuff volumes of different size. A control algorithm has to be developed for linear deflation of a cuff. This algorithm is to be integrated in the data acquisition system.

Stage 3:

Testing of the hardware and software application and characterization of the performance. Collection of measurement results for different cuff volumes and different deflate times.

Stage 4:

Development of an algorithm to detect pressure pulses during linear deflation. Also development of an algorithm to calculate blood pressure, from the collected measurement data. Again testing and performance measurement.

These four stages will be described subsequently.

2.3 Requirements for the system to develop

After considering the assignment and receiving more detailed information from people working at Dräger R&D it was decided that the new system should comply with the following specifications:

- The system should control the static cuff pressure in such a way that the amplitude of cuff pressure pulsations can easily be calculated. A linear deflation method is preferred.

- Using linear deflation it has to be investigated how NIBP measurement can gain on:
  - measurement speed,
  - measurement accuracy,
  - making the measurement more comfortable for the patient.

- It should be possible to have measurement data visible during cuff pressure control.

- Measurement data is to be stored for post processing.

- The user can describe the control of static cuff pressure and measurement specific details, such as initial cuff pressure and deflate time.

Chapter 3

Design constraints

In an ideal research environment, the available hardware is not limited and the researcher has a free choice of selecting equipment. In case of this research project, this would probably lead to the use of a servo valve, for it's good controllability and a computer system with an integrated data acquisition application. However, this is not the case in this project, where cost limits, compatibility with existing equipment, limited time and preferred components are introduced. Because the idea of linear cuff deflation was a new development and no information had ever been collected on this subject, this required a 'design from scratch'. Considering the limited time to develop a complete data acquisition system, digital controller and NIBP software, the use of existing hardware and software was considered necessary to deliver a prototype system within the available time. Dräger provided the existing techniques to be used in the new system and thereby established a great deal of the new system's performance characteristics.
All limiting conditions provided an environment where the freedom of design and implementation was reduced and demanded compromises on various points. The conditions that influenced the design and their effect, are discussed in the following paragraphs.

3.1 Available and recommended hardware

The system that will be developed should be easy to integrate with the existing monitoring equipment. This means a Dialog 2000 or Parameterbox serves as a basis for the new system. A Dialog 2000 evaluation model was chosen to be modified for testing.
The proportional valve was chosen by Dräger from a preferred supplier.

3.1.1 The Dialog 2000 and PC

The Dialog 2000 is a portable monitor device developed and built by Dräger Medical Electronics in Best. The monitor is used for measurement of medical parameters, such as NIBP, ECG, SpO_2, IBP and Temperature. The Dialog 2000 has a built-in LCD for easy readout of measurement data and setting information and six front panel keys for control. The software allows the user to

navigate through a menu structure to modify the Dialog's settings. In Appendix A technical information about this device is shown.

It was preferred to let the Dialog function as a prototype for linear deflation, where not too much changes to the existing hardware are allowed. This meant that solving the problem had to be done using external hardware and new software.
Because of the possibilities that software offers to solve a problem and because the Dialog 2000 is a microprocessor based device, it was considered a good solution to cover most of the new system's implementation in software.
By making this decision, another limiting condition was created. Because of the fixed project time of six months, the application that was to be developed could not be developed within the Dialog 2000. The complex software architecture and interrelated software modules for the medical parameters did not allow the creation of a new application within the project time.
Thus the idea emerged to connect a PC to the dialog by using the serial port and make an application on the PC, where the Dialog could be set into a test mode, making the device controllable by the PC. The proposed setup defines almost the complete hardware environment (Figure 3.1).

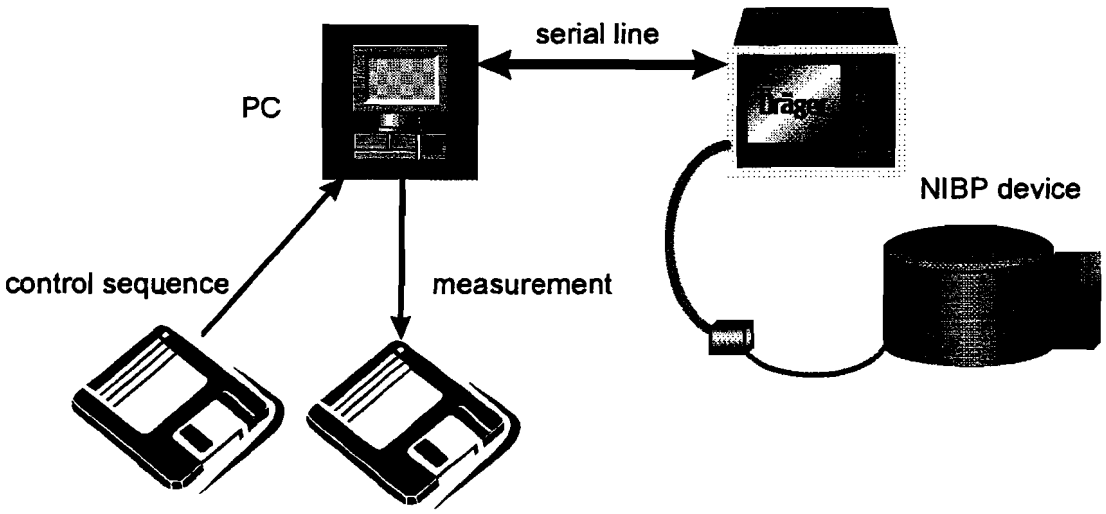


Figure 3.1 Hardware configuration.

The Dialog 2000 is a microprocessor based measurement system that is controlled by software applications running on the operating system VRTX. Because the scope of this project is limited to the NIBP parameter, the Dialog 2000 is discussed from this point of view.
The Dialog 2000 functions, concerning NIBP measurement are shown in Figure 1.1.

The cuff that is used for the oscillometric NIBP measurement is connected to the Dialog 2000 by a double hose. One hose is used for inflation, deflation and overpressure detection. The other hose is used for measurement. This measurement method allows the device to measure the pressure in the cuff accurate, without pressure drop across the hose. The overpressure safety measures pressure in the inflate/deflate hose and is a hardware implemented circuit. This circuit switches off the NIBP power of the valves and pump if the pressure in the cuff exceeds a level of 300 mmHg. Because the valves are normally open, this will result in a safe situation. The overpressure safety overrules the microprocessor settings if the cuff is overpressured.

A NIBP measurement can be described as follows. First the setting for adult or neonate is set by the user. According to this settings the software sets (alarm) limits for variables such as initial cuff pressure, systolic, diastolic and mean pressure.
When the user presses the NIBP start button, the CPU closes the step valve and dump valve and activates the pump to inflate the cuff. The CPU monitors the signal from the pressure transducer and switches the pump off when the initial pressure is reached. Then a measurement is started and the CPU selects either the dump valve (large flow) or the step valve (small flow) to change the cuff pressure. When the pressure drop has been larger than preferred (e.g. a valve was opened too long) the pump can be activated again to increase cuff pressure again. The motor is driven by a pulse width modulated signal that ranges from 0% duty cycle to 100% duty cycle in 255 steps.
The cuff pressure is measured by a transducer that is connected to an amplifier. This amplified pressure signal is converted to a digital signal by a 20 bit analog-to-digital converter (ADC). This signal is sampled by the NIBP software application at a rate of 75Hz. A communication system within the Dialog 2000 takes care of data transport between software tasks and hardware.
More detailed information about the Dialog 2000 hardware functions can be found in document [15].

3.1.2 The proportional valve

The proportional valve is a low-cost alternative to the more accurate servo valve. Because the cost of components is of high importance, the servo valve is not considered as alternative. Although the servo valve offers much better feed forward characteristics its price is in the range of thousands of guilders, where the proportional valve costs less than a hundred guilders.
The proportional valve is used to restrict the flow of air in a pneumatic circuit. An electropneumatic proportional valve has the advantage of flow control proportional to an applied electronic control signal. To understand the operation and properties of the proportional valve some device specific details are discussed.

A proportional valve was first developed by modifying a directional valve. These modifications included changes on the solenoid, the armature stroke and the spool bearings.
The large hysteresis (typical 15%) of the proportional valve is caused by the reverse magnetization of the magnetic circuit and the friction between mechanical components.
Figure 3.2 shows a cross-section of a proportional valve.

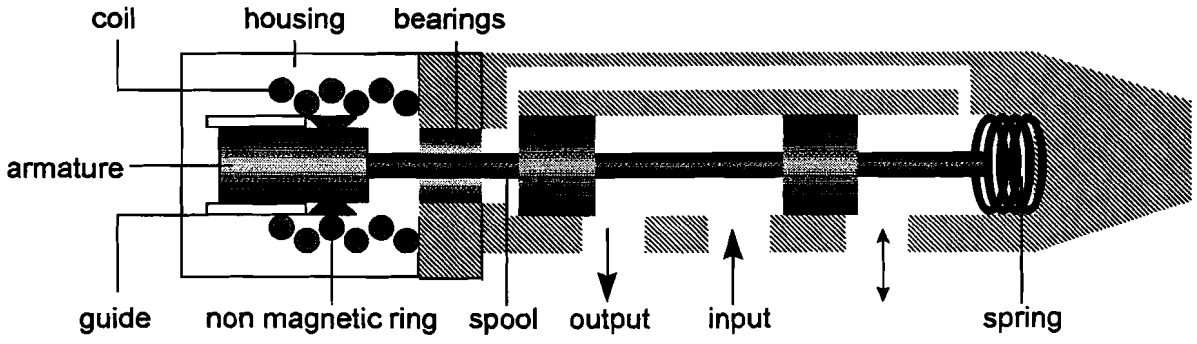
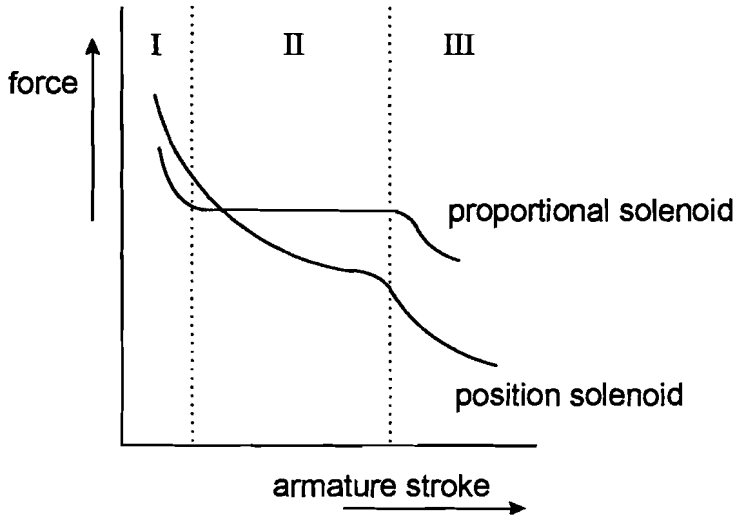


Figure 3.2 Proportional valve cross section.

The proportional solenoid is a modified version of a position solenoid. The position solenoid has an air gap through which magnetic flux lines pass. In a two-position valve the force exerted by the solenoid is proportional to the air gap. In a proportional solenoid the air gap is changed, so the force exerted by the armature is constant and does not change with spool position. This offers the possibility of controlling the force through the armature by a coil current. Graph 3.1 shows the difference between these types of solenoids.
The armature stroke of a proportional valve is smaller than the stroke of a directional valve. A typical value of this stroke is between 2 to 4 mm.
The spool bearings should minimize the friction forces that thwart the movement of the armature and spool. In most valves a synthetic sleeve bearing is used.

The housing, armature and guide are made of ferromagnetic material and form a closed magnetic circuit. The guide is interrupted by a non magnetic ring. When a current flows through the coil, the magnetic flux flows through the housing, the guide and armature. At the driver ring the largest part of the flux crosses the radial air gap to the armature, because it's magnetic resistance is far smaller than the resistance of the axial air gap.
When designed properly, the armature position does not influence the passing of flux and a distance independent force on the armature will exist (Graph 3.1, area II). The constant force-air gap relationship allows the designer to regulate the force of the armature with the current through the coil.



Graph 3.1 Armature stroke vs. force.

Because a spring, with a linear force displacement dependency ($F = C \cdot u$) is used, the displacement is directly proportional to the force.

The used proportional valve is a new developed valve, which was supplied for test purposes. There were two types of proportional valves available. Appendix B shows the properties and electrical data of these valves.

3.2 Software development

An application in C language has to be implemented to control the static cuff pressure.
On first hand the software has to be developed for a PC, keeping in mind that the application can easily be ported to a Parameterbox or dialog 2000.

3.2.1 Structured analysis and structured design

For developing a system various techniques can be used. For development of this controller the method of Hatley and Pirbhai for structured analysis and design was adopted. This method is used at the R&D department and seemed to be a proper choice for development.

The method includes the following aspects:
- Definition of requirements
- Analysis
- Design

The requirements were set by Dräger and are included in the assignment (Chapter 2.1).
There is much time spent on analysis of the system. The analysis describes what a user can expect from the system. How these specifications are met is decided in the design and implementation phase.

3.2.2 Coding standard

For implementation of the design a coding standard is available. The coding standard specifies the coding and naming conventions to use when writing source, using the ANSI-C programming language.
The advantage of using a coding standard is that the program source code presents itself in a uniform look. This results in great advantages when someone has to modify or look up someone else's code. It also leads to less programming errors which might arise from hard to understand C constructions.
Some programmers experience the coding standard as a disadvantage, because it restricts their way of programming to the rules of the standard.

The coding standard contains templates for source and include files and specifies naming conventions of variable types.
Developing software using the coding standard requires a very good documentation and takes more time than an unrestricted way of programming. However, the extra time that is spent during development highly facilitates maintenance and portability.

3.3 Goal and planning

Before starting with the assignment, a planning was made. The planning is a matrix where in the first column all tasks are listed. The time was assigned to the tasks by a percentage of the available time in one week.
The planning aimed the completion of a control application for static cuff pressure control in an NIBP application. During the progress of the work, the planning was adjusted to a single control application. This adjustment was necessary, because the development of the data acquisition system made clear that a NIBP application would take more time than was available. Also the new developed system showed some new aspects and possibilities for NIBP measurement, which could not have been predicted at first hand.
At completion of the control system, more time was spend on measurements and writing recommendations, so the new aspects can be used in future research.

Chapter 4

Analysis

The analysis describes what functions the system that will be developed should perform and what a user can expect the system to do. Of course the system should meet the requirements that were defined earlier.
The analysis described in this chapter is a summary of the document "Structured analysis of a static cuff pressure controlling application" [14]. Due to its length and level of detail this document is not suitable to be included in this report.
The analysis document consists of an introduction and engineering requirements. The engineering requirements describe the external interface requirements, capability requirements, data element requirements, sizing and timing requirements and the design constraints.

4.1 External interface requirements

The external interface requirements model the process that is under development in its environment. The interfacing with the environment and the interaction with the environment is defined. This is described in a context diagram (Figure 4.1).

Every rectangle is called a terminator and models an external process that the new system interacts with. For each terminator, the following data is defined:

1) Short description of the terminator's use and behavior.
Usually the name of the terminator is self explaining, but this allows the designer to describe the terminator in more detail.

2) Input events generated externally.
The events the terminator receives from external sources.

3) Input data requested by this system.
This describes the data- and/or control signals the terminator requests form the process.

4) Data outputs.
The data generated by this terminator.

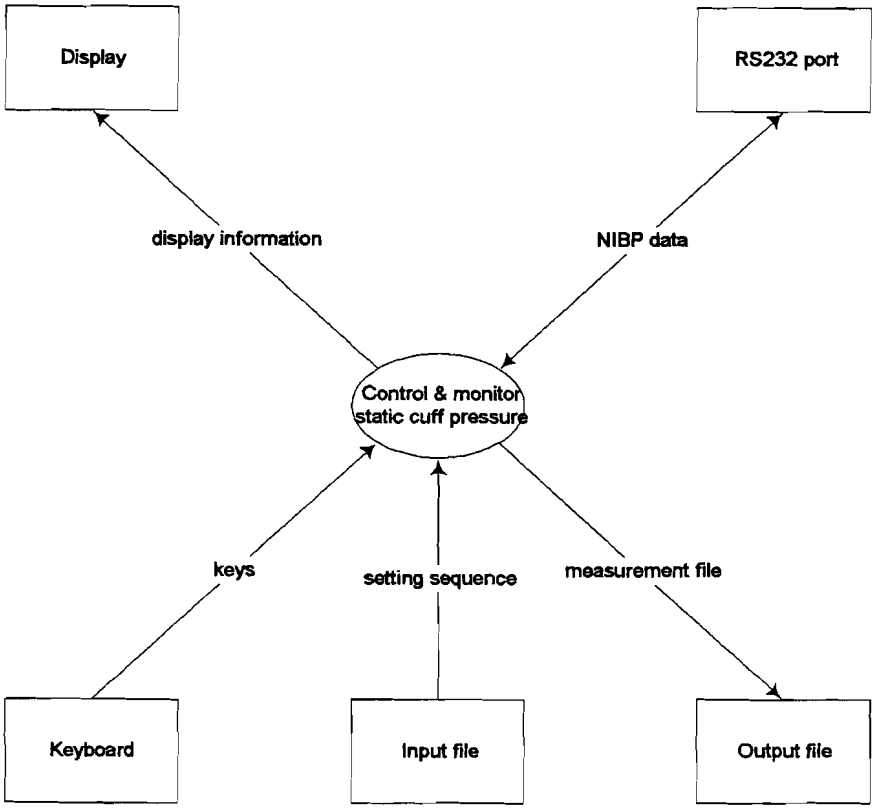


Figure 4.1 Context diagram.

4.2 Capability requirements

The capability requirements describe the new system in full detail. The description is leveled, where each level gives the designer an overview of the system with a certain degree of detail. The top level gives a complete overview, but with very little detail. Each lower level gives less overview, but more detail.
The capability requirements describe the processes that are needed to meet the system requirements. A process should not be confused with a task. Because a process describes 'what' a certain module should do (analysis) and a task is an implementation specific module ('how'), processes and tasks are two different things.

Each process is decomposed in sub-processes until a primitive process, the pspec is defined. The pspec describes in full detail what a process is expected to do and what other primitive processes it interacts with.
Data flow and control flow are strictly separated and are displayed in the data flow diagram and the control flow diagram.
An example, taken from [14] is shown in Figure 4.2 and Figure 4.3.

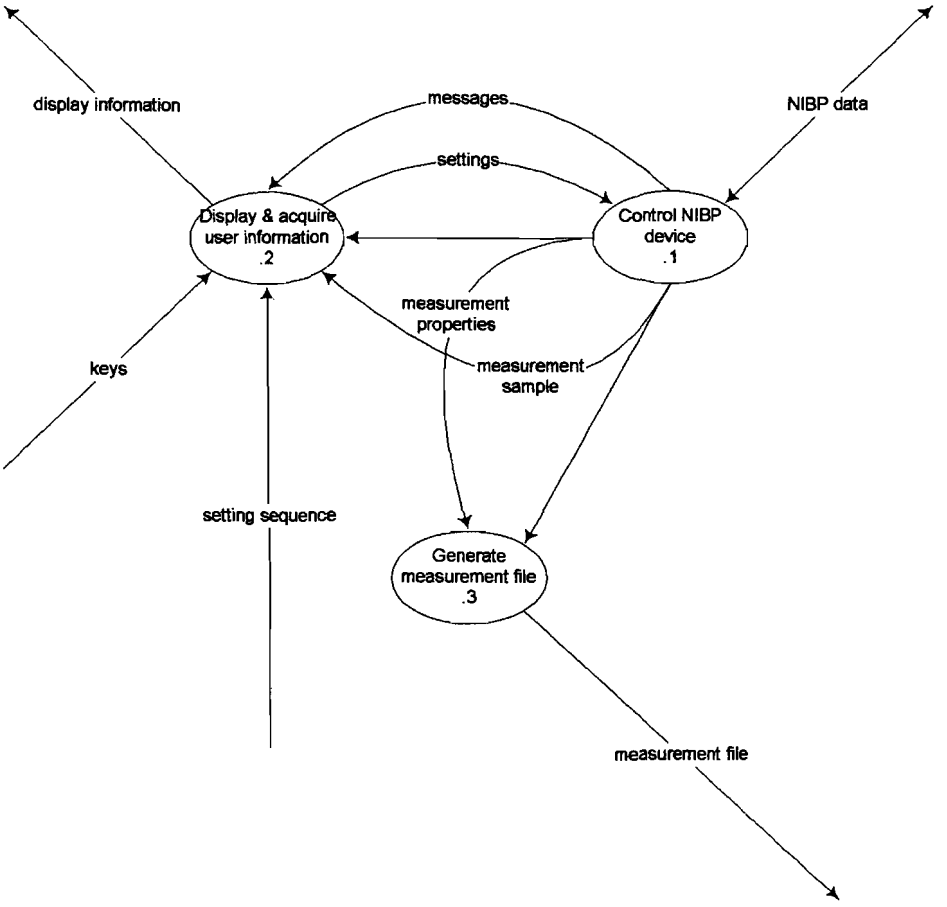


Figure 4.2 Example of the data flow diagram; the top level diagram DFD0.

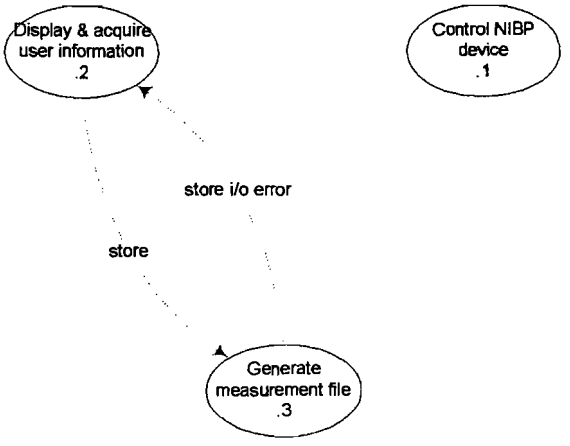


Figure 4.3 Example of the control flow diagram; the top level diagram CFD0.

Each operation on control is described in a control specification, the CSPEC. The CSPEC consists of a process activation Table (PAT) and a state transition diagram (STD). The PAT describes the activation and priority of processes with regard to other processes. The STD describes the states a process can be in and the events that cause transitions between states.

4.3 Data element requirements

The data element requirements describe all data- and control flows that are used in the capability requirements. Each element can be composed of other data elements. Primitive data elements are the lowest level elements. The elements are described by:
- Name of the data element
- Brief description

for non-primitive data elements:
- Data element composition

for primitive data elements:
- Limit/range of values required

4.4 Sizing and timing requirements

The sizing and timing requirements describe the real time performance of the system. The chapter describes the distribution of data into the system at a fixed sample rate. Also the requirements that exist for controlling the proportional valve and static cuff pressure within error limits have effect on timing and sample rates.

4.5 Design constraints

The design constraints are described in this report in Chapter 3; Design constraints.

Chapter 5

Digital PID control

The proportional valve is controlled by a digital PID controller. Because the analysis of this controller is not described in the analysis document, this chapter discusses the digital PID controller and digital feed forward control system. Also some aspects of digital control are reviewed.

5.1 Control system

The control system model is shown in Figure 5.1.

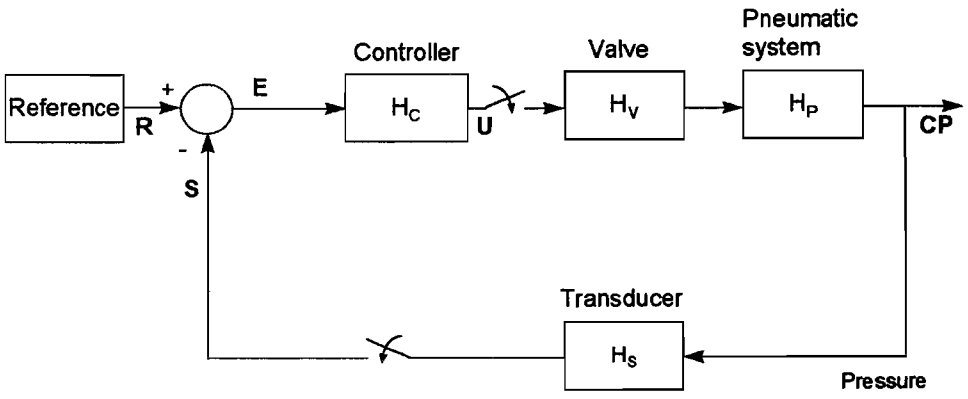


Figure 5.1 Control system model.

In this model the controller is modeled by its transfer function H_C. This function will be described in the next paragraph.
The valve transfer function is given by H_V. The other pneumatic components, such as hose, cuff and connections are modeled by the H_P transfer function. H_S models the pressure transducer transfer function and is assumed to be equal to 1. This is allowed, because the measurement of pressure is performed by a hardware module that meets very strict requirements with regard to pressure accuracy.
These functions are of course discrete transfer functions. Because continuos control theory is the basis for the control loop, the transfer functions are first described in the Laplace domain (s-domain) and later on transferred to the discrete time domain (z-domain).

For this system the closed loop transfer function can be described by:

$$\frac{CP}{R} = \frac{H_C H_V H_P}{1 + H_C H_V H_P H_S}$$

where CP is the controlled cuff pressure signal, and R is the reference signal.
For a parallel continuous PID algorithm H_C can be written as:

$$H_C(s) = \frac{U}{E} = K(1 + \frac{1}{s\tau_i} + s\tau_d)$$ Equation 5.1

5.2 Discrete implementation

To implement a continuous control algorithm such as a PID controller on a digital system, it is necessary to approximate the derivatives and the integral that appear in the control algorithm. A few different ways to do this are discussed.

5.2.1 Proportional action

The proportional term is

$$P = K(bR - S)$$

Where b is an constant that is used to multiply the reference signal in order to eliminate amplification or attenuation errors, caused by measurement.
This term is implemented simply by replacing the continuous variables with their sampled versions.

$$P(t_n) = K[bR(t_n) - S(t_n)]$$ Equation 5.2

where $\{t_n\}$ denotes the sampling moments at which a new value is calculated.

5.2.2 Integral action

The integral term is

$$I(t) = \frac{K}{\tau_i} \int_0^t E(s)ds$$

Using this form, it follows that

$$\frac{dI}{dt} = \frac{K}{\tau_i} \cdot E$$

Approximating the derivative by a difference,

$$\frac{\left(I(t_{n+1}) - I(t_n)\right)}{T} = \frac{K}{\tau_i} \cdot E(t_n)$$

Where T is the differential time between two sample moments. In a sampled system, T can be calculated by: $T = 1 / f_{sample}$, where f_{sample} is the sampling frequency.

This leads to the following recursive equation for the integral term:

$$I(t_{n+1}) = I(t_n) + \frac{KT}{\tau_i} \cdot E(t_n) \qquad \text{Equation 5.3}$$

5.2.3 Derivative action

The derivative term is given by

$$D = K \cdot \tau_{d\cdot} \frac{dE}{dt}$$

The derivative action may result in difficulties if there is high frequency measurement noise. A sinusoidal measurement noise

$$n = a \cdot \sin(\omega \cdot t)$$

will give the following contribution to the control signal:

$$u_n = K \cdot \tau_d \cdot \frac{dn}{dt} = a \cdot K \cdot \tau_d \cdot \omega \cdot \cos(\omega \cdot t)$$

The amplitude of the control signal can therefore be large, depending on the frequency (ω) of the noise. The high frequency gain of the derivative term is limited to avoid this difficulty. This limitation can be implemented by:

$$\frac{\tau_d}{N} \cdot \frac{dD}{dt} + D = K \cdot \tau_d \cdot \frac{dE}{dt}$$

there are several ways of approximating the derivative

5.2.3.1 *Forward Differences*

Approximating the derivative by a forward difference gives:

$$\frac{\tau_d}{N} \cdot \frac{\left(D(t_{n+1}) - D(t_n)\right)}{T} + D(t_n) = K \cdot \tau_d \cdot \frac{\left(E(t_{n+1}) - E(t_n)\right)}{T}$$

This can be rewritten as

$$D(t_{n+1}) = \left(1 - \frac{T \cdot N}{\tau_d}\right) \cdot D(t_n) + K \cdot N \cdot \left(E(t_{n+1}) - E(t_n)\right) \qquad \text{Equation 5.4}$$

5.2.3.2 *Backward Differences*

The derivative can also be approximated by a backward difference:

$$\frac{\tau_d}{N} \cdot \frac{\left(D(t_n) - D(t_{n-1})\right)}{T} + D(t_n) = K \cdot \tau_d \cdot \frac{\left(E(t_n) - E(t_{n-1})\right)}{T}$$

This can be rewritten as

$$D(t_n) = \frac{\tau_d}{\tau_d + N \cdot T} D(t_{n-1}) + \frac{K \cdot \tau_d \cdot N}{\tau_d + N \cdot T} \cdot \left(E(t_n) - E(t_{n-1})\right) \qquad \text{Equation 5.5}$$

5.2.3.3 *Tustin's Approximation*

Another approximation proposed by Tustin is commonly used, given by:

$$D(t_n) = \frac{\left(2 \cdot \tau_d - T \cdot N\right)}{\left(2 \cdot \tau_d + T \cdot N\right)} \cdot D(t_{n-1}) + \frac{2 \cdot K \cdot N \cdot \tau_d}{\left(2 \cdot \tau_d + T \cdot N\right)} \cdot \left(E(t_n) - E(t_{n-1})\right) \qquad \text{Equation 5.6}$$

Notice that all approximations have the same form:

$$D(t_n) = a_i \cdot D(t_{n-1}) + b_i\left(E(t_n) - E(t_{n-1})\right)$$

but with different values of parameters a_i and b_i. The approximation of Equation 5.4 requires that $\tau_d > N \cdot T/2$. The approximation becomes unstable for very small τ_d. The other approximations (5.5 and 5.6) are stable for all values of τ_d. Notice, that Tustin's approximation (Equation 5.6) and the forward differences approximation (Equation 5.4) give negative values of a_i if $\tau_d < N \cdot T/2$. This is undesirable because the approximation will then exhibit ringing. So, only the approximation in Equation 5.5 will give good results for all values of τ_d.

5.3 Feed forward algorithm

A feed forward signal is added to the control system as is shown in Figure 5.2.

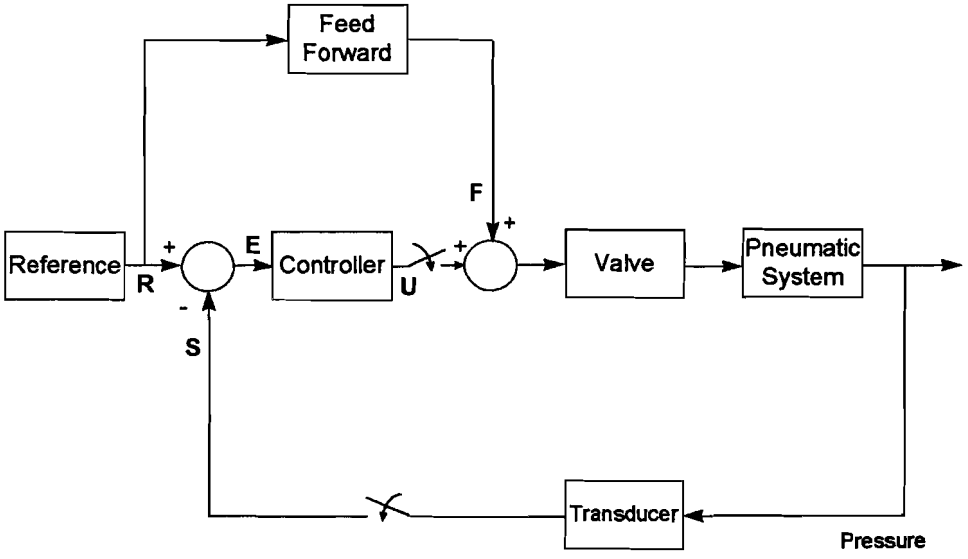
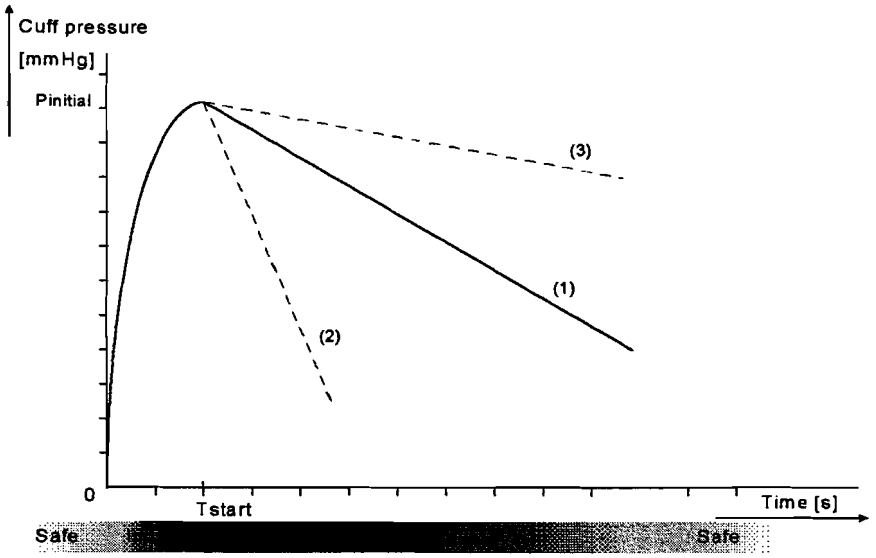


Figure 5.2 Feed forward control system.

Feed forward is added in order to reduce the error signal (E) especially at the initial condition when the PID controller has not yet gained information about the controlled process. This is important, because the valve has a dead zone and hysteresis. If no feed forward would be used the controller will be unable to control the pressure with a small error from the start of a measurement.
This will be explained using Graph 5.1.



Graph 5.1 Errors without feed forward.

The preferred pressure drop is given by line (1). Without feed forward, a quick settling controller (assuming a relative high gain) will generate overshoot. Overshoot, resulting in a large control signal will result in a large pressure drop (2). Because the cuff cannot be pressurized again, while measuring, this is an unrecoverable error. Of course the cuff can be inflated again by turning on the air pump, but the pump introduces a large pressure ripple signal while inflating that drowns the signal of interest.
On the other hand, a slow controller will be unable to find the valve setpoint within a quick time, resulting in a pressure drop that is too small (3). This will result in an unacceptable long measurement time.

It is considered useful to provide the controller with feed forward information about the valve in order to calculate a setpoint at measurement start and to reduce errors.

The reference pressure signal is given by:

$$P_{REF}(t) = P_{INITIAL} - \frac{\left(P_{INITIAL} - P_{END}\right)}{T_{DEFLATE}} \cdot t \qquad \text{Equation 5.7}$$

When considering the reference signal, the desired pressure drop is:

$$\frac{dP_{REF}}{dt} = P'_{REF} = -\frac{\left(P_{INITIAL} - P_{END}\right)}{T_{DEFLATE}} \qquad \text{Equation 5.8}$$

for a linear deflation during a timespan $T_{DEFLATE}$.

Approximating the pressure decrease when a valve with fixed restriction is used, gives: (Equation 5.9)

$$P_{VALVE}(t) = P_{INITIAL} \cdot e^{-\frac{t}{RC}}$$ Equation 5.9

and

$$\frac{dP_{VALVE}}{dt} = P'_{VALVE}(t) = -\frac{P_{INITIAL}}{RC} \cdot e^{-\frac{t}{RC}}$$ Equation 5.10

The pressure decrease at the start of a measurement (t=0) will be:

$$P'_{VALVE}(0) = -\frac{P_{INITIAL}}{RC}$$ Equation 5.11

An ideal initial controller setpoint would be $P'_{VALVE}(0)=P'_{REF}$.
To provide the information that allows the controller to find this setpoint it is required to know how $P'_{VALVE}(0)$ relates to the applied control signal magnitude.
It can be seen that $P'_{VALVE}(0)$ is depending on the initial pressure, a restriction constant R and a volume constant C.
Of course, the initial pressure is known when the controller starts, so the relationship

$$U \sim RC$$

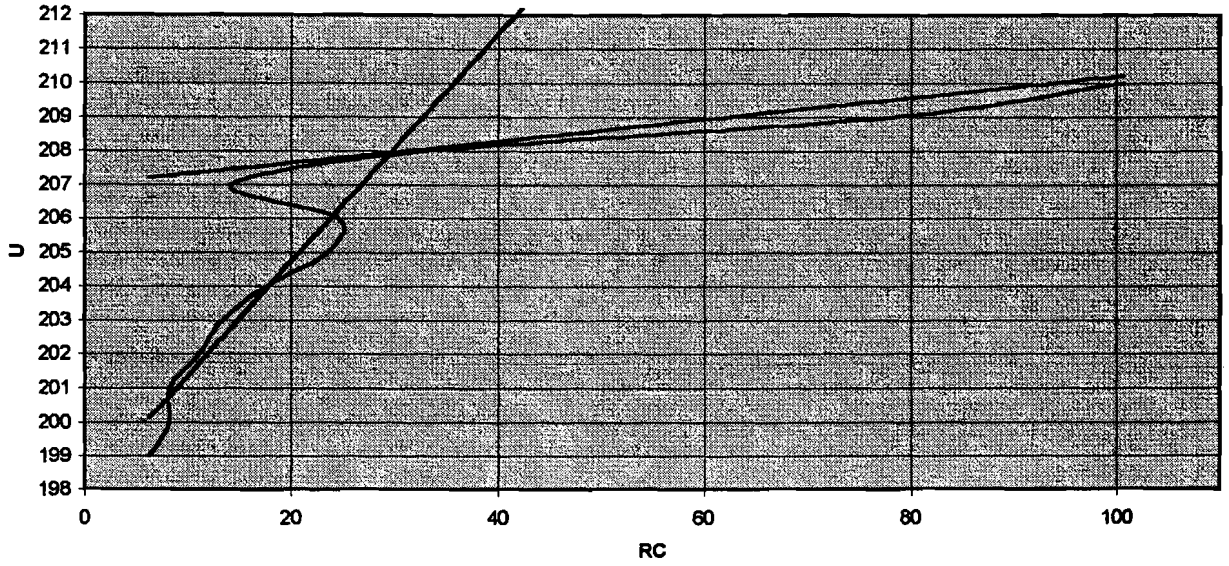
has to be determined, where U represents the control signal magnitude.
In order to acquire this information several measurements were carried out.
The pressure drop was recorded for a range of control signal magnitudes and after that the derivative was plotted in relation to the applied control signal.
Because the valve response varies within a certain range, the derivative is characterized by an average value.
The graphs have been added in Appendix D. Table 5.1 gives the results of the measurements.

Control signal magnitude U	RC	Graph name
210	100.70	u210.xls
209	78.28	u209.xls
208	33.10	u208.xls
207	14.22	u207.xls
206	24.59	u206.xls
205	23.50	u205.xls
204	17.65	u204.xls
203	13.37	u203.xls
202	11.11	u202.xls
201	8.200	u201.xls
200	8.235	u200.xls
199	6.261	u199.xls

Table 5.1 Measurement of RC.

The results were collected, using a cuff size of 500mL, which was inflated to an initial pressure of 200mmHg.

In order to calculate the feed forward signal, the relationship $RC \sim U$ is necessary. The relationship is plotted in Graph 5.2. In Graph 5.2, the $RC \sim U$ relationship is approximated by two linear lines, because this was considered a practical fit, which could be used in a computer program, without too much calculation time.



Graph 5.2 U~RC relationship and practical approximation.

The feed forward algorithm was then calculated to be:

for RC ≥ 30 : $U=0.032*RC+207$

for RC < 30 : $U=0.3362*RC+198$

using the intersection of the linear approximations to guarantee a continuous control signal.

RC is used as input signal for the feed forward section, calculated by using Equation 5.11. In the software implementation the cuff size is measured while inflating the cuff. The RC valve is adjusted according to the detected cuff size.

5.4 Aspects of digital control

Because the cuff pressure is controlled with a digital system, aspects of digital control that can deteriorate performance are considered. The aspects that have to be taken into account are digital oscillation, output wind-up, rounding and pressure errors due to sampling.

5.4.1 Digital oscillation.

This is an oscillation, originating from the limited resolution of the ADC and/or DAC. In the closed loop the least significant bit will toggle between 0 and 1. Improving the resolution of the ADC or DAC will diminish the amplitude of the oscillation.
In the control loop, the Dialog 2000 performs both AD and DA conversions. The analog pressure signal is converted with a 12 bit resolution over a range of 0 to 350 mmHg.
DA conversion is carried out by the TPU part of the microprocessor. The TPU generates a pulse width modulated signal with an amplitude of 5V and a dutycycle between 0 and 100%. The pulse width modulation is controlled in 255 steps.

5.4.2 Output wind-up

This effect can exist because of long term integration of a positive or negative error signal which can lead to an overflow in the arithmetic part of the microprocessor. This can result in a erroneous control signal. Output windup can be prevented by detecting the overflow condition and resetting the integrator.
To solve this problem in a control application the values of the involved variables must be checked and limited to a maximum or minimum value. Also the integrating part of the control algorithm can be reset if a variable reaches the limits of its range.

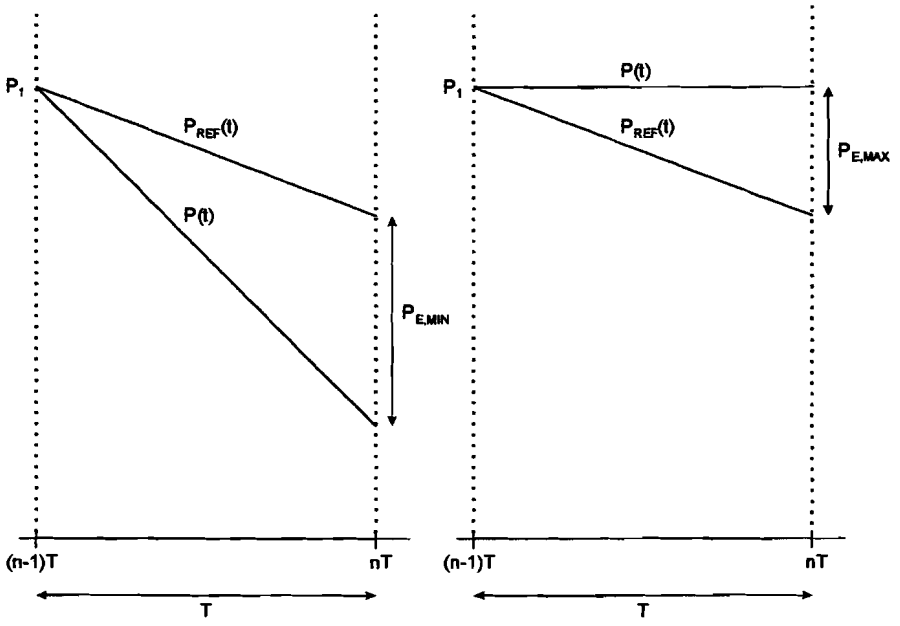
5.4.3 Rounding

When signal values are very small, the truncation or rounding errors that are made by the arithmetic part of the microprocessor can result in erroneous control. This effect will occur when a very high sample rate, compared to the sample signal, is used. The problem can be solved by using more accurate calculations than strictly required for the output signal calculations.

The aspects to take into account that have been discussed now, can be applied to any digital control system. For the application that controls the cuff pressure, some other effects that relate to digital control have also to be reviewed.

5.4.4 Pressure errors

The worst case errors in pressure control are made when the valve is either shut or full opened between two sample moments.
This is graphically explained in Graph 5.3.



Graph 5.3 Worst case errors.

The errors are given by $P_{E,MIN}$ for the worst case error in pressure drop and $P_{E,MAX}$ for the worst case error when the valve is closed. The pressure drop in timespan T is assumed to be linear. In fact this pressure drop has an exponential shape, but when sample time T is short, compared to the time constant of the exponential term a linear approximation is allowed. The sample time in the case is assumed to be in the order of milliseconds, where the time constant for a fully opened valve is in the order of seconds.
The reference pressure is given by P_{REF}.
P_1 is the actual cuff pressure at the sample moment t=(n-1)T.

The reference pressure can be given by:

$$P_{REF}(t) = P_{INITIAL} - \alpha \cdot t$$

where

$$\alpha = \frac{(P_{INITIAL} - P_{END})}{T_{MEASUREMENT}}$$

In case the control algorithm calculates a minimum output signal, that will result in a fully opened valve, the pressure at the next sample moment t = nT is given by:

$$P(t) = P_1 - \frac{P_1}{T_{deflate}} \cdot t$$

where $T_{deflate}$ is the time in which the pressure will be decreased to 0 mmHg, assuming a linear pressure drop.
The error can be calculated according to:

$$P_{E,MIN}(nT) = P_{REF}(nT) - P(nT)$$

This can be written as:

$$P_{E,MIN}(nT) = -\alpha \cdot T + \frac{P_1}{T_{deflate}} \cdot T$$

When the worst case error $P_{E,MIN}$ is defined, a minimum sample frequency $f_{s,min}$ can be calculated.

$$T = \frac{P_{E,MIN}(nT)}{\left(\frac{P_1}{T_{deflate}} - \alpha\right)}$$

$$f_{s,\min} = \frac{1}{T}$$

In case the control algorithm calculates a maximum output signal, that results in a shut valve, the pressure drop during the sample time T is zero.

$$P(nT) = P_1$$

The error $P_{E,MAX}$ is then calculated by:

$$P_{E,MAX} = P(nT) - P_{REF}(nT)$$

$$P_{E,MAX} = \alpha \cdot T$$

When a maximum worst case error is imposed, the minimum sample frequency for this error can be calculated to be:

$$f_{s,\min} = \frac{1}{T} = \frac{\alpha}{P_{E,MAX}}$$

Examples for error calculations concerning the control application are given in Appendix E.

Chapter 6

Design and Implementation

After an analysis of the system was completed, it was decided which part of the system was to be implemented in hardware and which part should be software implemented. Of course the design issue has been dealt with already, as described in chapter 3, so design and implementation are included in the same chapter.
The hardware- and software environment that is used supports a real-time multitasking system and proved to be a proper environment for developing a digital controller. The controller is embedded in an application that offers the user access to the controller, the controller information and the measurement data.

6.1 Hardware design and implementation

The hardware consists of a personal computer, a serial communication line and a modified Dialog 2000. Chapter 3 describes the considerations for choosing this setup, that might be considered limiting for the digital controller performance. However it showed to be a practical and useful environment for analyzing valve and controller performance.

6.1.1 PC system

A personal computer system was chosen to host the control application and the user interfacing implementation. The computer has a serial port (RS232) for communication with the Dialog 2000.
The computer system features the following specifications:
- 80486DX processor at a clock speed of 50MHz.
- Memory 16Mb RAM and at least 10Mb of free disk space.
- A VGA color screen, AT keyboard and Microsoft pointing device.
- UART 16550 serial communication port, supporting at least a 19k2 baudrate.

The personal computer that executes the application was also used to develop the application. This resulted in quick testing and development possibilities.

The serial communication line is a standard RS232 communication line with a resistor of 10kΩ added between pins 1 and 9. Appendix C shows the serial line configuration.

6.1.2 The Dialog 2000

The Dialog 2000 was modified on several points, listed below:

modification number:	*modification description:*
1	The pump output is used to drive the proportional valve.
2	The step valve output is used to drive the pump.
3	The step valve is permanent closed.
4	Communication port speed is set to 19k2.

For the modifications 1,2 and 3 an external adapter was built. The schematics of the adapter are shown in Appendix G.
Modification 4 required downloading of a new software version in the dialog 2000. The application software was changed in file pm2000/applic/hwsystem/scidriver.c, where in line 60 the value of 9600 was changed to 19200.
This software was downloaded using the HP software probe. Details on the software modification procedure are described in [32].

The proportional valve that was connected to the hardware adapter is the FAS 01-226P-010H0 valve. Appendix B shows detailed technical information on this valve.

6.2 Software

The hardware implementation covers a part of the functions that were specified in the analysis. The remaining implementation problems are solved in software.
The implementation of the static cuff pressure controlling process has to be a real-time application, considering the feedback controller sub-process with it's fixed sample rate. However not all sub-processes have real time demands (e.g. the display process, does not need to update the information on fixed time intervals). This leads to an implementation of subprocesses in different tasks, each task with its own real time demand.
To allow this concept to work on a computer system, a real time, multitasking operating system is required.

6.2.1 Introduction to real-time multitasking systems.

A task consists of a collection of machine code statements that can be executed by a microprocessor. In a multitasking environment, various tasks exist, that are all executed at the same time.

In a single processor environment, real multitasking is not possible, because the processor can only execute one statement at a time. The smallest executable collection of statements is called a thread. In order to let the processor execute more tasks, threads of different tasks are executed subsequently. For correct operation of this switching between threads of different tasks, an operating system is required.

In a single processor environment the multitasking operating system distributes the available processor time to the tasks to be executed. Each task is allowed to utilize the microprocessor, depending on the operating system's algorithm to assign processor time to tasks. After the execution of a thread, the operating system (also running on the same microprocessor) takes some time to decide which thread should be executed and to prepare the system for executing this task. (make memory available, load the stack and register information of this task).
This can be explained graphical in Figure 6.1 where three tasks are handled by one processor.

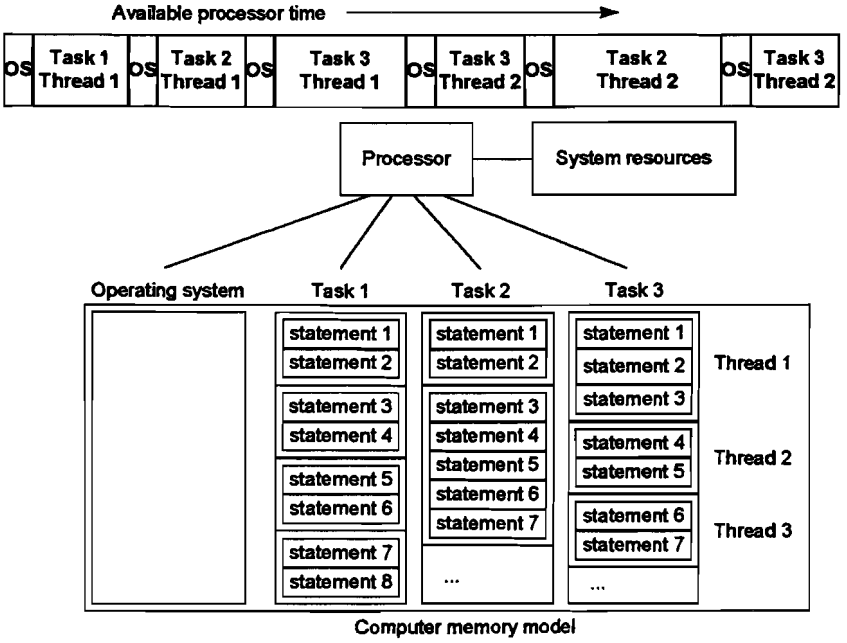


Figure 6.1 Multitasking; one processor executes threads of different tasks.

Multitasking systems can be divided in two groups; time sharing systems and real time systems.

6.2.1.1 Time sharing multitasking

When several tasks are requiring more processing power than the actual performance of the processor, the available processor time has to be distributed 'fairly' to all tasks. This means that the operating system assigns processor time (time slices) to each task, using a scheduling algorithm. This often results in a decrease of processor efficiency, because of the substantial scheduling overhead. When a timesharing multitasking system runs two tasks that are scheduled equally, each task can for example be slackened to 40% of the speed it would run with, when executed in a single task environment.

In these systems tasks are often running independent and inter task communication is hardly supported for real time applications. Examples for time sharing multitasking systems are UNIX and Windows.

6.2.1.2 Real time multitasking

In a real time system there has to be more available processor time than the tasks demand. When this is not the case, real time performance is not guaranteed. In a real time system there is no 'fair' distribution of processor time. Each task is assigned a priority level and tasks can claim processor time according to their priority. A task with a high priority can take processor time from a low priority process at any time, regardless of the fairness of such a task switch. Because there is more processor time available than needed, each task will have sufficient time to run.
For real time systems it is important to react to events within a specified time. Detection of external events is often implemented with interrupt handlers and a real-time system with short interrupt latency.
Regarding the task reaction time, real time multitasking systems can be divided in cooperative- and preemptive multitasking systems. For preemptive systems the task reaction time is about the interrupt latency time, whereas for cooperative systems the task reaction time is the time between two kernel calls. When using cooperative multitasking, the tasks have to 'cooperate' by making calls to the kernel, allowing the kernel to switch active tasks.

6.2.2 The real-time kernel

Because a time sharing system is no useful runtime environment for a discrete controller application (the principle of 'fair' processor time distribution is not desired in a real time environment), the RTKernel program is chosen to give MSDOS real-time multitasking specifications.
The tasks use cooperative multitasking in order to be able to give the programmer more control of the task execution. To explain this, the operation of the RTKernel scheduler is discussed.

6.2.2.1 The scheduler

The scheduler decides which task can utilize the processor at a certain moment. Only the task state and the task priority are considered when making this decision.
The RTKernel scheduler is an event driven system, in contrast with most other interrupt driven schedulers. An event is an inter-task-communication that can be generated by either a task or an interrupt handler. Events can lead to a change of task state.
RTKernel also uses a timer interrupt handler, which is merely an interrupt handler that can change the state of a task that waits for a certain moment in time.

The scheduler works according to four rules:

1) From all tasks which are in 'ready' state, the active task is the task with the highest priority.
2) If there are more tasks in 'ready' state, with the same priority, the task which has not been activated for the longest time, is activated.
3) When several tasks are waiting for an event, the task with the highest priority is activated at the occurrence of the event.
4) Except for time slicing task changes, a task change is only made if rule 1 would otherwise be violated. The number of task switches is minimized.

Using preemptive multitasking, these rules are never violated. However, using cooperative multitasking, rule 1 can be violated between the occurrence of an event and the first kernel call of the active task.

6.2.2.2 Task switching

The RTKernel distinguishes three methods of task switching; blocking, activating and time slicing.

Blocking: The blocking task change occurs when the active task blocks it's own execution. This happens when a task is for example waiting to receive a message or releases the CPU by calling a delay function. In this case, the RTKernel selects the next task according to the task change rules. If there is no task available, the idle task is activated.

Activating: The activating task change occurs when a task with a higher priority than the active task changes to the 'ready' state. This is the case when a task with a high priority waits for a message and a task with low priority is active. As soon as the message is received by the high priority task, the active task is blocked and the previously blocked task is activated.

Time slicing: Time slicing task changes are made by the scheduler which blocks the active task after a certain time span.

Beside the three methods of task switching, there is also the difference in cooperative- or preemptive task switching. Preemptive task switching is initiated by an interrupt handler. Cooperative task switching is initiated by the tasks. Blocking task switches are cooperative task switches, for they can only be initiated by tasks. A blocking preemptive task switch is therefore considered an error condition by the RTKernel.
Activating task switches can be made either cooperative or preemptive.

6.2.2.3 Tasks

A task consists of a C function without parameters and a stack. Also a task priority, ranging from 1 to 64 is assigned. A higher priority means a higher demand for real time and when deciding which task is activated only the relative difference between priorities is considered.

A task is referenced to, by using a unique task handle, assigned at the time of task creation.
The stack of a task contains all local variables that are declared within the task function. This allows creation of tasks (or functions) within a task (or functions). This offers the possibility to open the same function more than once, in different tasks, without data loss.
Also all tasks can reference to global data.

At initialization of the RTKernel module, two tasks are initiated; the idle task and the main task. The idle task has priority level 0, ensuring that this process will never have the same or a higher priority than a new defined process. The idle task is activated when there are no other tasks that can be activated.
The priority of the main task is assigned at initialization of the kernel and the stack size is equal to the stack size that is assigned by the operating system.

A task has always one of the following states:

Current: The task that is currently running (active) is in the state current. Only one task can be in current state

Ready: All tasks that can be activated are in this state. Usually all tasks in this state have a priority level that is the same or lower than the active task.

Suspended: Tasks in this state cannot be activated. This state can only be entered by the function RTKSuspend. RTKResume sets a task state back to ready

Blocked: A task in this state cannot be activated because it is waiting for an event. This task can only be set in ready state by another task or an interrupt handler.

Delaying: This state is entered by a task that is waiting for a certain lapse of time. After the time has lapsed, the task is put in ready or current state by the RTKernel timer interrupt handler.

Timed: This state is a blocked state, with a time limit. This task state gets ready or current after occurrence of an event or a lapse of time.

6.2.2.4 Inter task communication

The possibilities for tasks to communicate are by means of semaphores, mailboxes and message passing

Semaphores: Semaphores are used for synchronization of tasks. A semaphore offers the possibility to exchange information about the activating or suspending of tasks.

Mailboxes: Mailboxes are an extension of the semaphore principle and are used for exchange of variable types between processes. The size of a mailbox is

configurable; a mailbox slot can be configured and also the number of slots per mailbox.

Message passing: Message passing is used for direct data exchange between processes without buffering. This requires the communicating tasks to be ready and synchronized.

6.2.3 External software modules

The software application that was developed, uses functions of (commercial) ready for use software modules. These modules are supplied by RTKernel, Borland and Dräger. The external modules facilitated the writing of a new application For these modules, their function and interfacing with the self developed software application is described.

6.2.3.1 RTKernel modules

The RTKernel modules offer services that are useful in a real-time multitasking environment. The modules are supplied with examples of the application of functions.
The header files of the modules can be included in source code. The modules used for implementation will be discussed briefly.

RTTime module: The timer module is a timer device driver for RTKernel. It supplies a variable frequency interrupt timer and a high resolution clock. The module can be configured to use various hardware timer services such as the PC timer chip or the real-time clock. Default the PC timer chip is used. This module can be added to a program by including the rttimer.h file.

RTKeyboard module The keyboard module supplies functions to read keys or events from the keyboard or to put character codes in the keyboard buffer. The module keeps the CPU free for other tasks to use when waiting for keyboard input (no polling). This module is included by the rtkeybrd.h file.

RTCom module The communication module allows fully interrupt driven serial communication from 50 to 115200 baud, assuming hardware to support this speed. I/O addresses and IRQ's are configurable and a send and receive buffer of 64Kb (maximum) is available. Also the FIFO buffers of the 16550 UART are supported. The header file is rtcom.h.

The rtkernel.h module file is also included, which supplies all basic RTKernel functions. For correct linking the rtktl.lib (large memory model) is added to the project tree.

6.2.3.2 *Borland C++ modules*

The modules from the Borland C++ environment are not discussed one by one. This paragraph contends with reciting the included modules. More information can be found in the C manuals [18], [26] and [34].
The Borland graphics interface is used to obtain graphic results on the VGA screen (graphics.h) and several other modules, such as stdio.h, stdlib.h, conio.h, string.h
were used.

6.2.3.3 *The service protocol*

For communication with the Dialog 2000 a service protocol is used. The protocol conforms to the standard issued by the Dräger service department and is described in [31]. The protocol assumes a serial communication device connected to the service terminal of the Dialog 2000.
Most important commands of the protocol are:

<identifier>	print the value of <identifier>. Values are displayed as decimal numbers.
<identifier><value>	set the value of <identifier> to <value>
REG <identifier>	report all subsequent changes of the value of <identifier>
UNREG <identifier>	stop reporting changes of the value of <identifier>
<CR>	repeats the last command
<control>E	toggles echo on/off.

Using the Dialog service protocol several values from actuators and sensors can be read, either direct from hardware or after software processing.

6.2.4 Software configuration

The software hierarchy, shown in the system environment, is displayed in Figure 6.2.

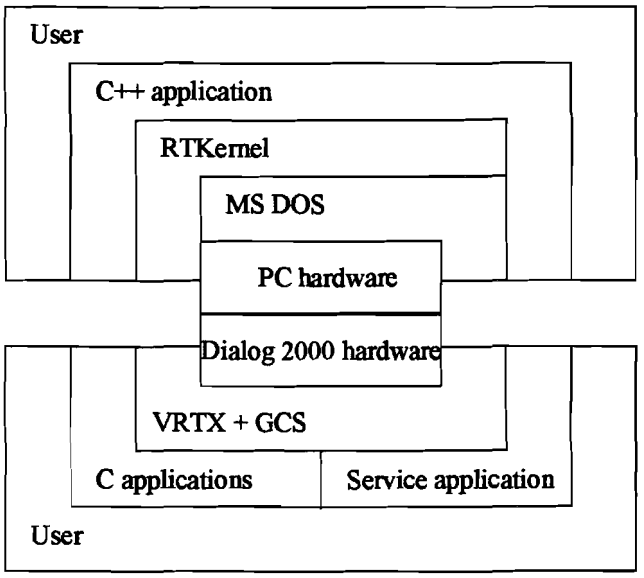


Figure 6.2 System hierarchy.

6.2.5 Static cuff pressure control application

This chapter describes the implementation specific details of the complete software application. This is done in several paragraphs, concerning the implementation of tasks, the inter task communication and assignment of priority level. Also the implementation of human interfacing, file handling and interfacing with the Dialog 2000 is described.

6.2.5.1 Design

The processes that were defined during the analysis phase were converted to a software design according to Figure 6.3.

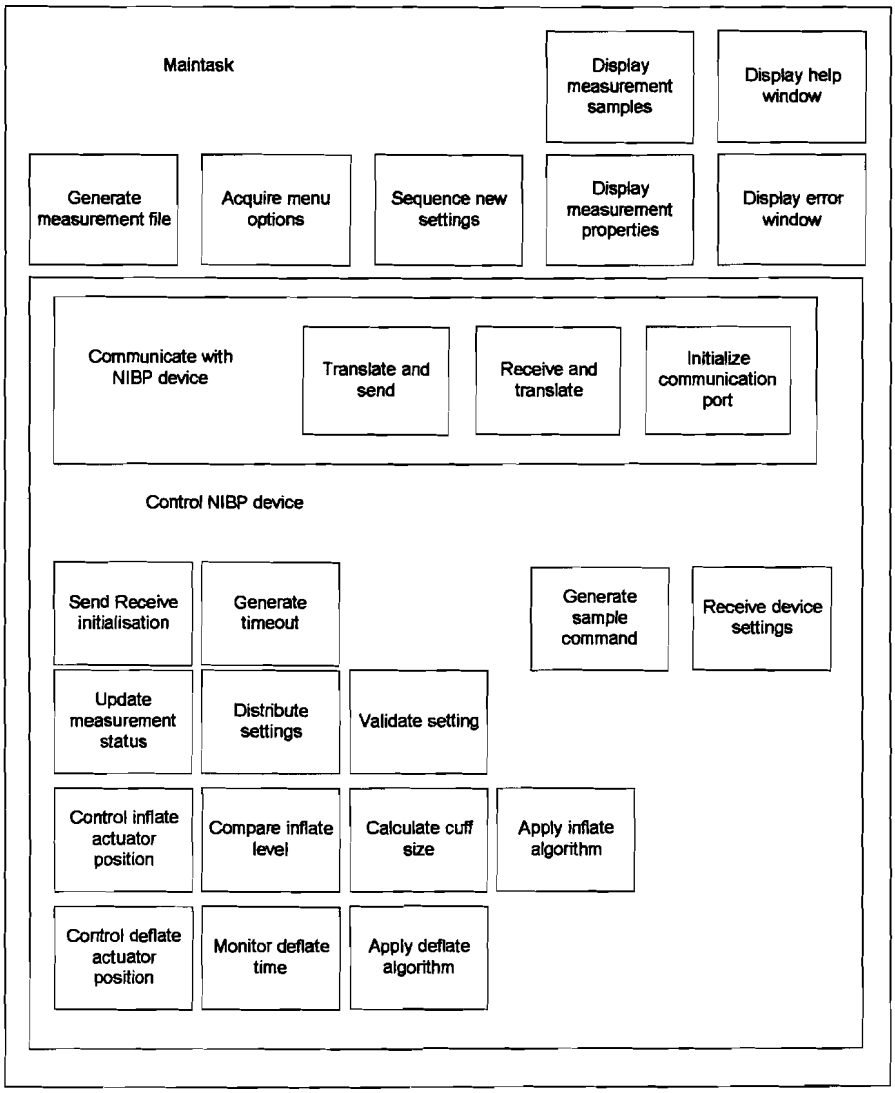


Figure 6.3 Cuff pressure control application.

Each rectangle in Figure 6.3 is implemented as a task. It was decided to create this many tasks because this would give a transparent translation from analysis to implementation and because this would give an opportunity to learn more about a multitasking environment.

6.2.5.2 *Task implementation*

Each task is implemented in the C program language. Each task is assigned the following properties:

Taskhandle:
The taskhandle is used throughout the program to refer to a process. A taskhandle is a unique name that has to be defined global in order to give other processes access to it.

Task code:
The task code is a C function without parameters, that describes the program code that should be executed by the task. An unrestricted number of tasks with the same task code can be created. This is allowed, because each task executes the same code, but with different local variables.

Priority level:
The priority level is the base priority of a task and should be an integer between 1 and 64. A high priority number means precedence over low priority tasks. A newly created task with a higher priority than the current running task, causes the new task to be activated immediately.

Stack size:
The stack size is the amount of memory in bytes, that is allocated and assigned to a task by the kernel at the moment of task creation. It should be verified that the compiler generates code which allows stack checking, at least during development phase. The RTKernel manual advises a minimum stack size of 4096 bytes during development.
The real amount of stack size that is assigned to a task is the stack size that is requested by the task creation function, increased with 256 bytes for interrupts and 27 bytes for internal kernel use.

Name:
The name of the task should not exceed 15 characters and is used for easy identification of the task. The kernel copies a pointer to this string, so the name can not be changed after task creation.

Together with the allocation of a stack, the kernel also allocates a TCB (Task Control Block) and a data buffer for the co-processor. The TCB is used by the kernel internally. The data buffer that is allocated for the co-processor has a size of 98 bytes, or 272 bytes (Borland C) when using the co-processor emulator.

Created tasks can be terminated, suspended, resumed or given a new priority level. Also a task can be questioned for its taskhandle, task state, task priority, task stack or detailed information.

6.2.5.3 *Inter task communication*

The tasks communicate by means of a mailbox system. For tasks that have data or control inputs, a receive mailbox type is declared. All mailboxes are assigned as global data types. A graphical view of a mailbox is given in Figure 6.4.

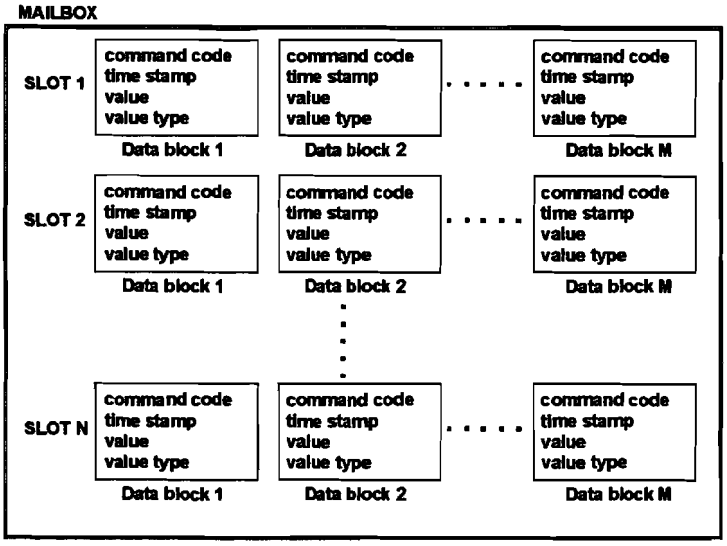


Figure 6.4 Mailbox structure.

A mailbox consists of N slots. Each slot can contain M data blocks. N and M are defined at the moment of creation of a mailbox and can not be changed at run-time. (The mailbox is declared as global static data).
Each data block contains the following information:

command code
The command code explains to the receiving task how the information in the data block should be interpreted or what kind of operation is expected on the information.

time stamp
The time stamp is used for data synchronization through the system and is useful in monitoring system performance.

value
This field can be used to store various types of information. The current application supports storage of one of the following data types: single character, array of characters, boolean, integer (signed or unsigned) or floating point number.

value type
This field indicates the data type that was stored in the 'value' field. Often the command code implies the use of a certain data type, but in case more data types are used with the same command code, this field gives a decisive answer.

6.2.5.4 Human interfacing

The human interfacing part can be divided in a display, keyboard and input file.

6.2.5.4.1 The display

The display is a standard VGA color screen with a resolution of 640 horizontal pixels and 480 vertical pixels. The available display size is divided into four windows. Each window has a number of configurable window parameters which are defined in the 'display.h' header file. The window parameters are:

X position:	The X position is a percentage of the available display width and marks the left side of the window.
Y position:	The Y position is a percentage of the available display height and marks the top side of the window.
Window width:	The window width is a percentage of the available display width.
Window height:	The window height is a percentage of the available display height.
Window background color:	This color is used for the template.
Window text color:	All text is displayed in this color.

Fixed window parameters are the character font and font size.
A change in display resolution does not affect the displayed information on the screen, because the display resolution is determined at run-time and all display parameters are calculated relative to the display resolution.
A graphical overview of window parameters is given in Figure 6.5.

The windows that are displayed:

Measurement samples window
In this window the collected measurement data can be displayed in a x-y graph. The window can display a configurable number of data traces. Each trace is scaled automatically and independent of other traces. Each trace has its own properties, like color, name, scale factor, upper limit, lower limit and visibility. Data display is started at the left side and moves to the right side with the increase of time. When the number of data points exceeds the available window width, the oldest data points are scrolled out of the window. The number of data points that are scrolled out of the window at one time can be defined in the 'display.h' header file.

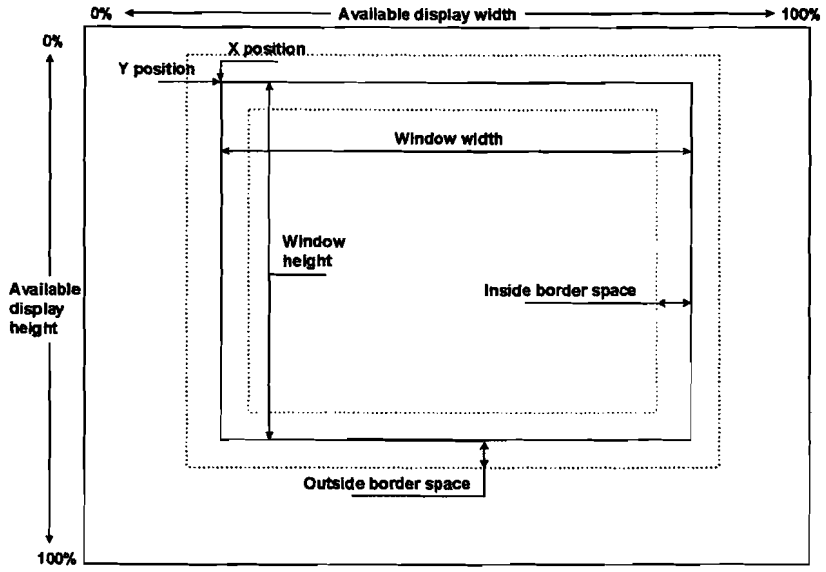


Figure 6.5 Window parameter definitions.

Measurement properties window
Measurement properties, such as measurement name, time, date, controller settings and initial cuff pressure are displayed in this window to give the user a quick impression of the measurement.

Error window
The error window displays error messages that concern the unexpected or erroneous execution of a measurement. This can include errors in the serial communication or errors in the measurement equipment.

Help window
The help window displays help messages for the user, concerning program operation (status messages) or available menu options.

6.2.5.4.2 The keyboard

The keyboard is used to give the user control of measurement execution. The keyboard allows the user to:
start a measurement,
save measurement data
and to terminate the program.
The available keys and options are described in the help window.

6.2.5.4.3 The input file

The input file allows the user to describe a measurement. The input file can be generated with any editor that can store a file in ASCII format. The syntax and allowed commands are described in appendix F.

6.2.5.5 File handling

Because the files that are used during a measurement are stored on a harddisk, it is important to describe how these files are accessed in the real-time environment. Harddisk operations generate interrupts that are handled immediately and therefore can disturb real-time specifications. In the application only one task can access a file, to prevent file sharing problems.
A measurement can only be started after the input file has been read. During a measurement no disk access is made. Data is collected in a memory file and is stored to the harddisk when the user presses the 'store' key.

6.2.5.6 Dialog 2000 interfacing

A serial line forms the communication with the Dialog 2000. The receiving and sending of serial data is handled by a RTKernel module. This module reads a the 'send' mailbox and delivers received data in a 'receive' mailbox. The size of these mailboxes is fixed at runtime and they have a different data structure than the mailboxes described in Chapter 5.5.3.
The communication is handled by two tasks. One task translates internal command codes to the Dialog 2000 service protocol and puts the translated data in the 'send' mailbox. An other task reads the 'receive' mailbox and translates the Dialog 2000 service protocol commands to internal commands. Both tasks use a translation table for translation of the Dialog 2000 service protocol and internal commands. The use of an internal and external language allows easy adaptation to other communication protocols and extension of commands.

Chapter 7

Measurement results

The measurements provided information on how well the controller is able to control cuff pressure and to get information about the usability of the valve. The measurement results were collected for different cuff sizes and different cuff deflate times. The results of the measurements are discussed in the last paragraph.

7.1 Test setup

The test setup, shown in Figure 7.1 was used to collect measurement information.

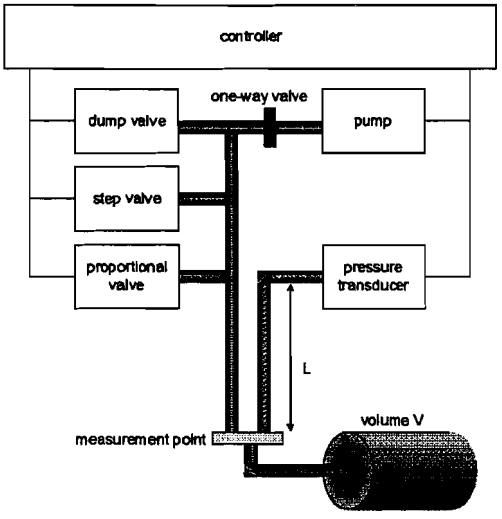


Figure 7.1 Measurement test setup.

A fixed volume V, which represents a cuff is connected by a hose to the measurement system. The measurement system consists of a controller, a pressure transducer and four actuators. The pump actuator is used to inflate the volume V to an initial pressure. Because the pump can only be switched on or off, inflation to a certain level is rather inaccurate for small volumes.

However, the smallest volume V can be inflated within 5% accuracy. Larger cuffs can be inflated more accurate. The pump feeds the air system through a check valve. This prevents reverse air flow when the pump is switched off.

The valves, used in this test setup are 'normally open' type valves. The dump valve is used to deflate the volume in a fast way and is used at the end of a measurement or in case a measurement is aborted. Also this valve is used to empty the cuff, when initializing a measurement.
The step valve is permanently closed. This valve is not used during a measurement, but is included in the configuration, because it is part of the Dialog 2000 equipment. Because this valve is normally open, a control signal to close the valve is applied.
The proportional valve is of main interest in this configuration, while this is the valve that is used to deflate the cuff during a measurement.
The pressure in the cuff is measured by a double hose system, that allows accurate measurement of the pressure inside the cuff model. The length of the hose (L) and the pressure drop across this length, has no influence on the accuracy of the measurement.
The controller is a modified Dialog 2000 (evaluation model) with a software version, that allows serial communication at 19200 baud, connected to a PC.

With this measurement test setup measurements are carried out. Figure 7.2 shows which volume sizes (V) and deflate times were used.

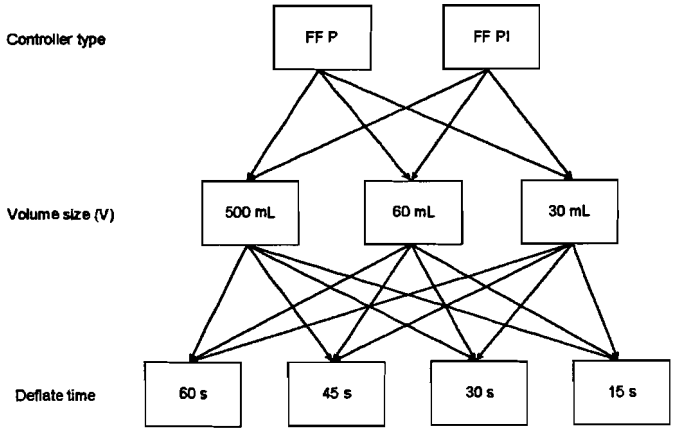


Figure 7.2 Measurements.

A measurement consists of the following steps, which are carried out in sequence:

1. Selection of the controller type.

The controller type to deflate the cuff was chosen to be either feed forward P (proportional) or feed forward PI (proportional and integrating).

2. Connection of the preferred cuff size.

For the measurements, syringes were used to simulate different cuff sizes.
The cuff volumes that were simulated are 500mL, 60mL and 30mL. The cuff is connected to the measurement system by a double hose of length L. A hose of 0.75 meter is used.

3. Selection of the measurement time.

For the measurement, deflate times of 60, 45, 30 and 15 seconds were chosen to test the controller speed and error.

4. Inflation of the cuff.

The initial pressure level is set to a level of 200mmHg, approximately 20mmHg above the initial pressure level commonly used while measuring NIBP. A relative high pressure has the advantage that controller errors are more easy to detect.

5. Deflation.

The fixed volume is deflated, using the selected algorithm and selected measurement time. The reference algorithm calculates reference pressure using equation 7.1

$$P_{ref} = P_{initial} - \frac{\left(P_{initial} - P_{end}\right)}{T_{deflate}} \cdot t \qquad \text{Equation 7.1}$$

where $P_{initial}$ is the initial pressure and P_{end} is the pressure to which the cuff should be deflated within the deflate time $T_{deflate}$.
For the measurements the following values are applicable:
$P_{initial}$ = 200mmHg (accuracy +5%),
P_{end} = 15 mmHg,
$T_{deflate}$ = 15, 30, 45, 60 seconds.
Deflation start is marked as t=0, where t is the actual time, since the start of deflation.

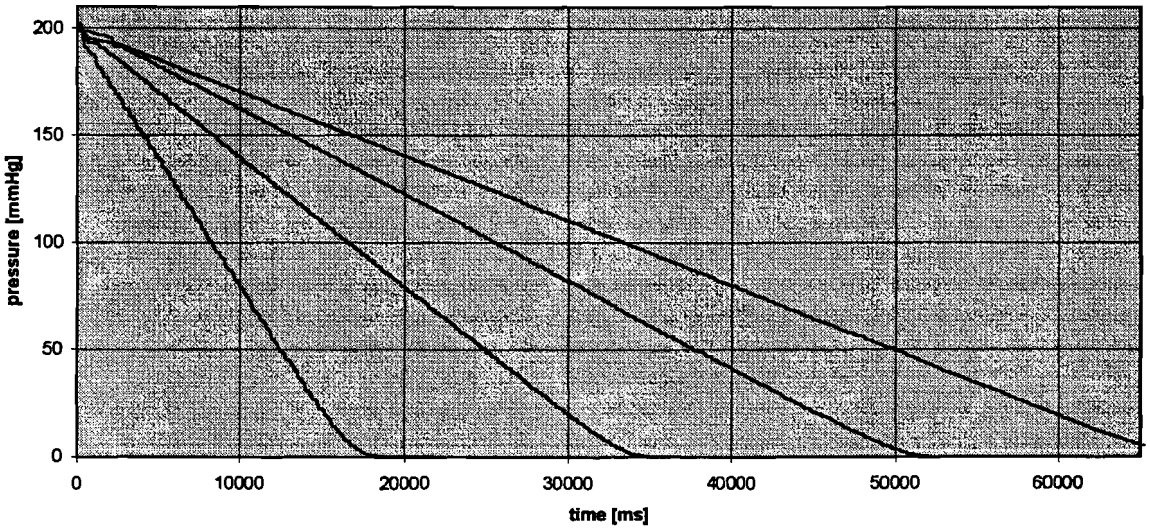
6. Storing of the measurement results

The measurement results are stored in a file.
For each sample, the sample time, the actuator positions, static cuff pressure and controller error is stored for post processing in for example Microsoft Excel.

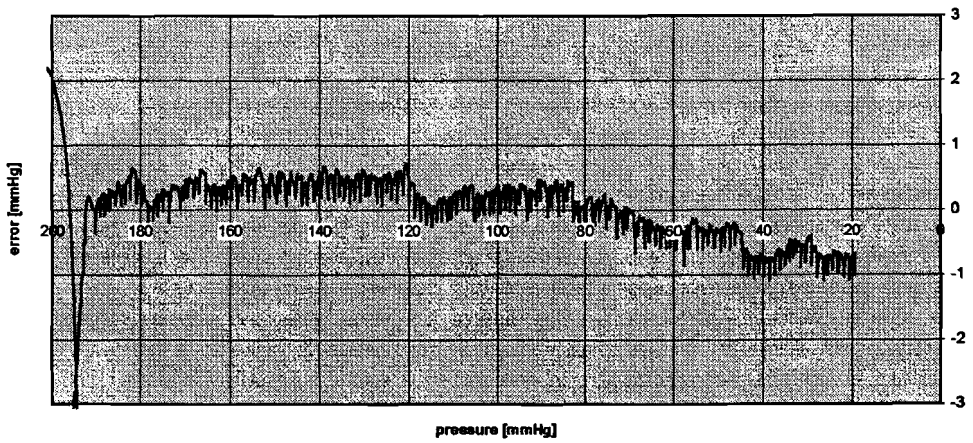
7.2 Measurement data

The complete data set, collected during the measurements, is included in Appendix H. In this paragraph the most interesting results will be shown.

Using proportional control, the pressure decrease during deflation is shown in Graph 7.1. Because a controller with only a proportional factor requires an error signal to generate output, it can never keep track of the reference signal without error. Because a feed forward signal is added to the control output, a relative small error remains. This is clearly shown in Graph 7.2 where the P controller, is used to keep track of the reference ramp signal. The controller gain was calculated for maximum response time and stable control operation. The displayed data is collected deflating a cuff of 500 mL during 15, 30, 45 and 60 seconds. The error signal is computed by comparing the actual cuff pressure with an ideal linear deflation curve.



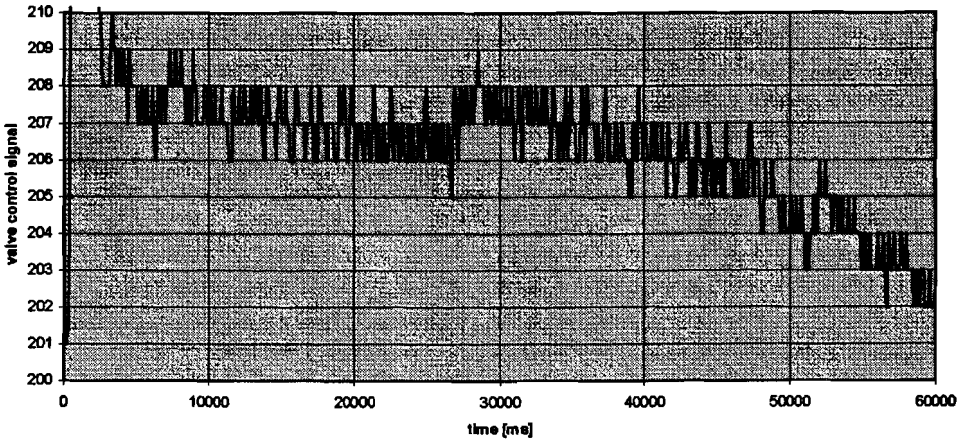
Graph 7.1 Pressure signal, deflating a 500 mL cuff in 15, 30, 45 and 60 seconds.



Graph 7.2 Controller error for 60 seconds deflation of a 500 mL cuff, using proportional control.

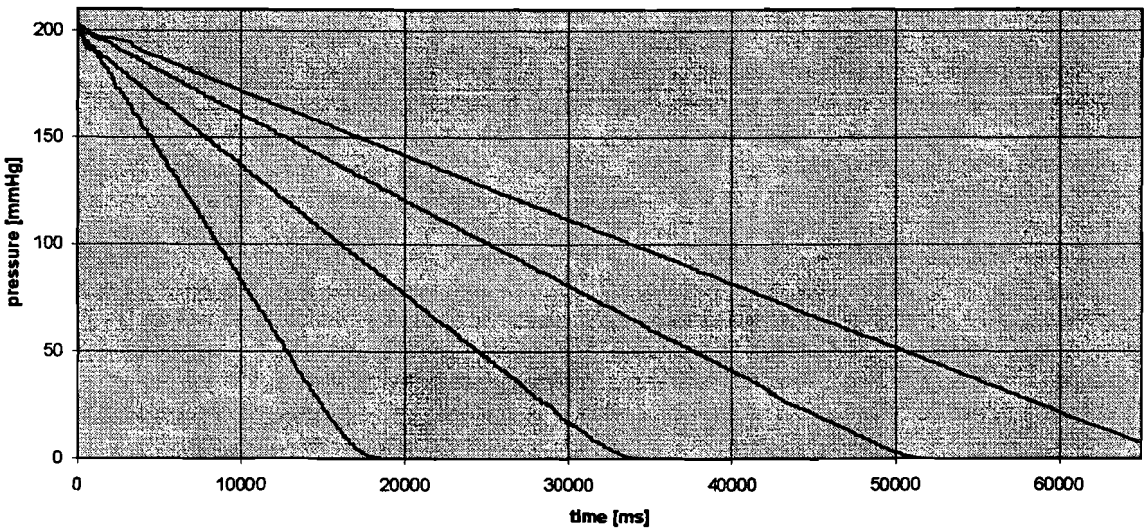
The valve control signal, that is applied to the proportional valve in case of 60 seconds deflation is shown in Graph 7.3. This signal is generated by the microprocessors TPU and is a pulse width modulated signal which can be related to the output voltage by Equation 7.2.

$$U_{valve} = \text{valve control signal} \cdot \frac{5}{255} \text{ [V]}$$ Equation 7.2



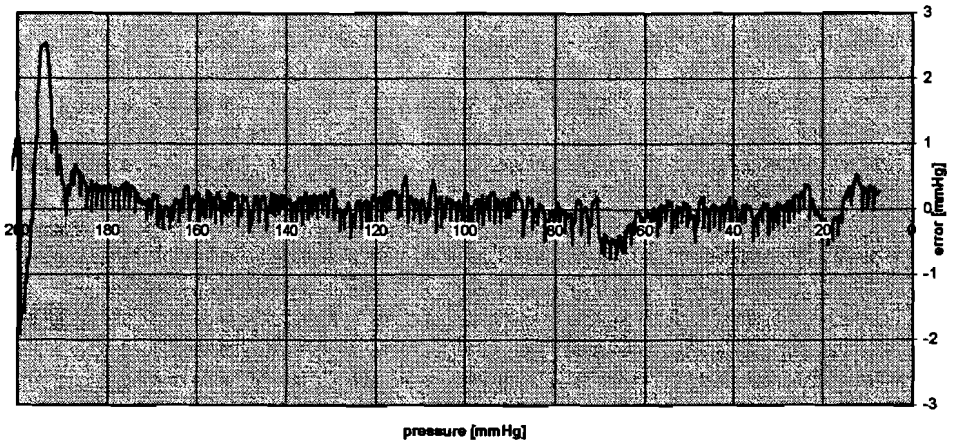
Graph 7.3 Valve control signal for 60 seconds deflation of a 500 mL cuff, using feed forward proportional control.

Using a feed forward PI controller to deflate a cuff of 500ml the following measurement results were obtained. Graph 7.4 shows the pressure drop for deflate times 15, 30, 45 and 60 seconds.



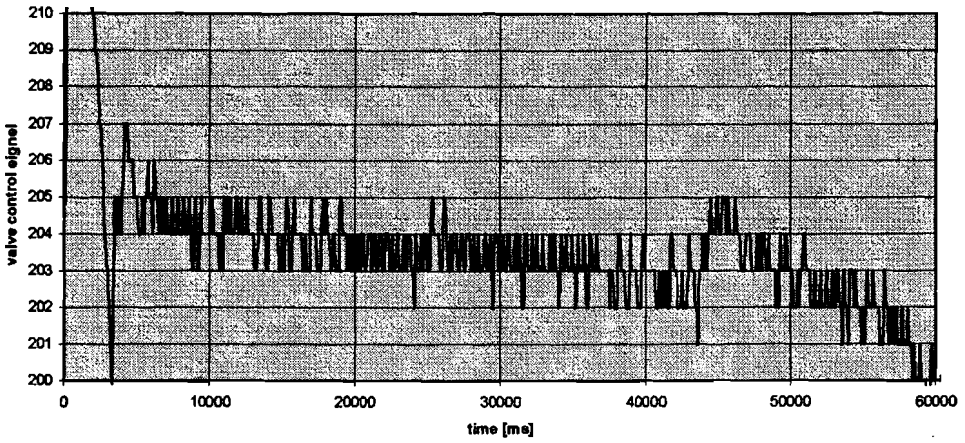
Graph 7.4 Cuff deflation for 15, 30, 45 and 60 seconds.

The controller error, compared with an ideal linear cuff deflation is displayed in Graph 7.5 for a deflate time of 60 seconds.



Graph 7.5 Controller error for 60 seconds deflation of a 500 mL cuff, using PI control.

To get more information on how the valve is controlled and how the control range of the valve is used, the recorder valve control signal is shown in Graph 7.6. This control signal applies to the deflation of a 500 mL cuff in 60 seconds.



Graph 7.6 Valve control signal for 60 seconds deflation of a 500 mL cuff, using PI control.

7.3 Data analysis

The measurement results are used to analyze the controller performance and valve controllability. The analysis is based on the complete set of measurements, as it is included in Appendix H.

7.3.1 Linear deflation graphs

The Graphs 6.1 and 6.4 show that linear deflation can be obtained with either a P or PI controller, for different deflate times. The cuff is inflated to a level of 200 mmHg and then deflated to an end pressure of 15 mmHg, during the deflate time.

7.3.2 Error signal graphs

The error signals (Graphs 7.2 and 7.5) show two different error types.

Controller settling error.
When the cuff is inflated and the controller is started, the settling of the controller shows in the error signal by a relative large error (> 2 mmHg). This error is unavoidable when a compromise between controller speed and accuracy is made. The feed forward signal, that is applied to minimize this error, can not eliminate this settling error because of a certain unpredictability in the valve characteristics. See Appendix 3 for valve characteristics.

'Static' control error.
When the controller is tracking the reference signal, a relative small error still exists between actual pressure and reference pressure.
The feed forward signal gives a rough estimation of the valve control signal, where the controller is used to eliminate the remaining error.
For a P controller the total elimination of error signal would require a perfect feed forward signal. As this is an impossible requirement, the P controller will add to the valve control signal.
Considering the properties of a P controller, it is clear that a zero error can never be obtained, but considering the error signal it can be stated that the feed forward algorithm gives good estimation data.
For a PI controller the error reduces to a signal around zero. Still the PI controller shows error signal.

7.3.3 Valve control graphs

Considering the valve control signal in both situations (Graphs 7.3 and 7.6) it can be seen that there is a problem in positioning the valve. Two problems are distinguished.

Resolution problem:
The controller is positioning the valve with a resolution of approximately 20 mV/valve control signal step. Looking at both Graph 6.3 and 6.6 it can be seen that the controller is oscillating around a certain setpoint. This is a normal phenomenon in digital controllers, but the amplitude of the oscillation affects the controlled pressure signal.

Limited use of valve control range:
The control range of the valve (Appendix 3) is used in a very small area. In Graph 7.3 the valve is controlled with a voltage between 4.12V and 3.96V. In Graph 7.6 the control voltage lies between 4.06V and 3.92V. This shows a inefficient use of output resolution. This can be traced to the valve, which allows a control signal between 0 and 5 V, but is in fact only controllable over a range of approximately 2 V.

Chapter 8

Evaluation of the control system

After analyzing the measurement results, the control application performance is evaluated. The evaluation will concern the accuracy and performance characteristics of the controller without blood pressure pulses and also the performance of the controller, integrated in an NIBP application.

8.1 Accuracy and performance.

The measurement results show that when a measurement is started, the controller uses some time to settle the control signal and to reduce error. This is caused by the proportional valve characteristic and controller demands concerning speed and accuracy. Because the proportional valve has an unpredictable dead band, the feed forward signal is an approximation of the signal that is required to control the valve. The P or PI controller is used to eliminate error signal.
The controller accuracy is also limited by the output resolution of the valve control signal. The measurements show that only 20% of the available output range is used during a measurement. During a measurement, when the controller is settled, this even reduces to 10% use of the control signal output range. In general the valve control signal is between 190 and 220 during a measurement.

8.2 Accuracy and performance in a NIBP application.

The performance of the controller in an NIBP application has been tested. For these tests the pressure port of a NIBP monitor tester was connected instead of a cuff. Because the measurements were performed in the last days of the assignment, no measurement results could be included in this report. Also the tests only gave a superficial impression of the performance of the system, because some limiting effects prevented faultless operation.
Limiting effects that affected the measurements were:

1 Sample rate.

The sample rate of the controller was set to a frequency of 10Hz. This frequency was 'chosen' because the serial port speed and communication protocol both limited higher frequencies. Choosing a higher frequency would cause the communication task to be active most of the time, preventing other tasks to be scheduled and thereby degrading the real-time performance.

2 No filtering.

Another limiting effect was that the pulses, generated by the NIBP monitor tester, could not be filtered out of the measurement signal and therefore slightly affected the controller operation. The pulses have a bandwidth of approximately 0.3 to 4 Hz, this corresponds to a heart rate of 250 to 20 beats per minute.

3 Pulse amplitude compared to controller error.

The measured pressure pulses have an amplitude that can be between 0 an 4 mmHg. Considering the error signal of the controller (<1 mmHg) this could give problems in the determination of blood pressure. This was not tested, because the software algorithm for pulse detection, artifact detection and blood pressure calculation was not available.
This meant the system could not be compared to the NIBP monitor tester, which was used to generate the pressure pulsations.

Chapter 9

Conclusions

In this chapter the first assignment and the results of the research are considered.

The proposed new method of linear cuff deflation was investigated with a data acquisition system that consists of a PC and a Dialog 2000 monitoring device that was modified to control a proportional valve. On this system a digital feed forward P/PI controller was implemented.
The control system was used to investigate the linear deflation of various cuff sizes (500 mL, 60 mL and 30 mL) during different deflate times (60 s, 45s, 30s and 15s), with two controller types (FF P and FF PI).
Because the available time allowed no further investigations, the pressure pulse detection algorithm and blood pressure calculation is subject for future research.

The proportional valve is a voltage controlled valve. The proportional valve that was used in the measurements shows to have a hysteresis of approximately 15%. The manufacturer specifies a control voltage range from 0 to 5 volts, although measurements show that only 50 % of this range is used to control the orifice area from 0 to its maximum. The valve has an unpredictable dead band zone (approximately 50%); when the valve is fully closed and a voltage is applied, to allow an air flow, the voltage that opens the valve is different for each measurement.
These specifications show that the valve is unfit for application in an NIBP device without feedback control application.

Despite the drawbacks of the valve, the research shows that the proportional valve, embedded in a control loop can be used to deflate a cuff for NIBP measurement with a linear pressure decrease. The linearity can be defined within error limits, depending on the sample rate, the used cuff size and deflate time. In the test setup an accuracy of approximately 1 mmHg could be attained at a sample frequency of 10 Hz with various cuff sizes and deflate times. It is mentioned that the error at the start of a measurement is in the range of ±4 mmHg, due to the unpredictable start position of the valve plunger.

The software, for this control system is developed according to the method of Hatley & Pirbhai and the Dräger coding standard. This proved to be a proper environment for the development of a soft real-time application in C code. The real-time aspects of the control application are necessary to control the cuff pressure with a proportional valve.

Compared to the current method of NIBP measurement, a decrease in measurement time is expected of approximately 25%, when an average measurement time of 35 seconds is considered.

The new method of linear deflation is more comfortable for the patient, compared with the current method of stepwise deflation.

The proportional valve control system has a limited accuracy of 1 mmHg. The pressure error that exists between reference signal and measurement signal can not be reduced to a smaller magnitude, due to hardware limitations. Most important limitation is the communication speed of the serial line and the communication protocol that is used to exchange data. Also the resolution of the digital-to-analog and analog-to-digital conversions limits the performance. Because the error signal can be in the same order of magnitude as the smallest measurement signals, this can give problems when calculating blood pressure.

Chapter 10

Recommendations

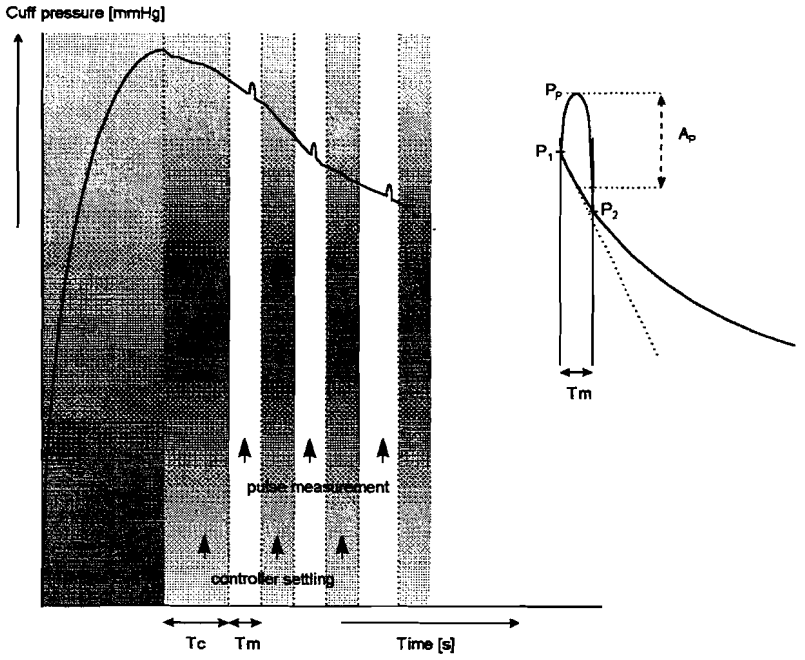
This chapter describes how the developed control application can be integrated in a NIBP measurement device. Recommendations are made, because the current prototype shows aspects that can be improved in further research. The recommendations concern the improvement of controller operation and the reduction of errors in blood pressure calculation.

10.1 Filtering

Because the measured blood pressure pulses affect controller operation it is recommended to use a low pass filter to eliminate pulse shaped pressure changes. To facilitate this filter operation it is also necessary to increase the controller sample rate.

10.2 Introducing local linearization

Because the pressure drop can never be controlled with zero error while using a proportional valve, the measurement signal will always contain the controller error. Because the error depends on uncontrollable conditions and therefore is unknown, it can not be subtracted from the measurement signal.
The measurements showed a typical error within approximately 1 mmHg after the settling of the controller. This error is large compared to the pressure pulses that are measured, which range from 0.5 to 4 mmHg.
The error is mainly caused by the non linearity of the proportional valve and (less) by the limited resolution of the valve driver.
To use the favorable properties of the proportional valve and let the non linearities spoil the measurement as less as possible, the following method of valve control is proposed.
The recommended measurement method of NIBP is described, using Graph 10.1.



Graph 10.1 Recommended method of NIBP measurement using a proportional valve.

In order to reduce the error signal in the measured signal it is recommended to leave the idea of controlling the static cuff pressure at any moment with a linear pressure decrease.
The recommended method is based on the same idea as linear deflation, but with a variable range of pressure decrease.
During a measurement two states can exist. Either the device is waiting for a pressure pulse, or the device is controlling cuff pressure.
Because the pressure pulses are predictable according to the heart rate, the moments at which the cuff pressure can be controlled are fixed. Assuming the maximum heart rate, HR_{MAX}, the controller time T_C can be defined as:

$$T_C = \frac{60}{HR_{MAX}} - T_M$$

During the cuff pressure control time T_C a controller can adjust the valve control signal. The reference signal for the controller is calculated by comparing the user preferred measurement time with an estimation of the total measurement time (with the current valve position). The estimation can be made by assuming an exponential pressure drop. If the estimation exceeds the allowed measurement time, the valve control signal is adjusted.

During the pulse measurement time T_M the cuff pressure is sampled with a sample rate T_S. A pulse detection algorithm detects a pressure pulse and linearizes the pressure drop between the start (P_1) and end (P_2) of the pulse (Graph 10.1).
The average of P_1 and P_2 can be subtracted from the pulse pressure P_P, to calculate the pulse amplitude A_P. Linearization of the pressure drop is allowed, because the pressure drop has an exponential shape and pulses are measured on the almost linear first part of the curve. Also the

time constant of the exponential curve is large (order of magnitude: minutes) in comparison to the pulse measurement time (order of magnitude: milliseconds).

10.3 Recommendation to reduce resolution errors

The limited resolution of the valve output driver is set by the pulse width modulated output of the TPU. This output allows a pulse width modulated signal between 0% and 100% dutycycle in a resolution of 255 steps. This means a resolution of approximately 20mV/step. Due to the limited control range of the valve this resolution is too rough. The error caused by this effect can be reduced by choosing a valve with a larger control range or redesign of the output driver. Redesign of the output driver can include the use of the MISO output on the CPU and an external DA converter.

10.4 Measurement time reduction

The length of an NIBP measurement is the time that is necessary to detect at least P valid pressure pulses from the patient. Assume that an average of $\bar{q}$ pulses is needed to determine a valid pressure pulse.
The total measurement time T_{MEAS} in seconds can then be given by:

$$T_{MEAS} = \frac{\bar{q} \cdot P \cdot 60}{HR}$$ Equation 10.1

when the heart rate HR is expressed in beats per minute.

Equation 10.1 shows that the measurement time can be decreased with an increase of heart rate. Because the controlled proportional valve can be used to perform almost any deflate time this offers the possibility of measuring NIBP in a minimum span of time.

Bibliography

1. Andersen, B.W., *The analysis and design of pneumatic systems*, John Wiley & Sons Inc., 1967.

2. Anderson, W.R., *Controlling electrohydraulic systems*, Marcel Dekker, Inc., New York, 1988, Fluid power and control series, nr. 7.

3. Arends, J.H.W.; Reimert, F.; Schrage J.J. *Basiskennis regeltechniek voor HTO elektrotechniek*, 1st impression, B.V. Uitgeverij Nijgh & Van Ditmar, Rijswijk, 1991.

4. Aström, K.J., Hägglund T., *Automatic tuning of PID controllers*, Instrument Society of America, 1988.

5. Biggelaar, W. van de, *R&D - C language coding standard*, version 1.2, Dräger Medical Electronics, Best, 1996.

6. Bronzino, J.D. *The biomedical engineering handbook*, CRC press, 1995.

7. Bruner, J.M.R., *Handbook of blood pressure monitoring*, 2nd printing, PSG publishing company, Inc., 1978, Postgraduate clinical techniques series; v. 1.

8. Franklin, G.F., Powell, J.D., Emami-Naeini A., *Feedback control of dynamic systems third edition*, Adison-Wesley Publishing Company, Inc., 1994.

9. Forster, F.K., Turney. D., *Oscillometric determination of diastolic, mean and systolic bloo pressure - a numerical model.*, Journal of Biomedical Engineering, vol. 108, November 1986.

10. Geddes, L.A. *The direct and indirect measurement of blood pressure*, Year book medical publishers, Chicago, 1970.

11. Guelen, I., *Ontwikkeling van een digitaal bestuurd instrument voor de drukregeling in een bovenarmmanchet*, Technische Universiteit Eindhoven, Eindhoven, 1996.

12. Haaf, E.A.W. ten, *Real-time data-acquisitie met niet-lineaire regeling / beschrijving van een Continue Variabele Transmissie*, Technische Universiteit Eindhoven, Eindhoven, 1997.

13. Hatley, D.J., Pirbhai, I. A., *Strategies for real-time system specification*, Dorset House Publishing Co., New York, 1987.

14. Heesakkers, J.P.J., *Structured analysis of a static cuff pressure controlling application*, version 0.1, Dräger Medical Electronics, Best, 1997.

15. Kaak, J.H., *HW description digital flash brd Dialog 2000*, Dräger Medical Electronics, Best, 1996.

16. Kleman, A., *Interfacing microprocessors in hydraulic systems*, Marcel Dekker Inc., New York, 1989, Fluid power and control series, nr 9.

17. Krist, T., *Fundamentele pneumatiek*, Technische Uitgeverij De Vey Mestdagh BV, Middelburg, 1979.

18. Lafore, R., *Programmeren met Microsoft C*, Academic Service, Schoonhoven, 1992.

19. Lausch, H. *Digitale Reglung hydraulischer Antriebe mittels pulsbreitenmoduliert angesteuerter Proportionalventile*, VDI-Verlag GmbH, Düsseldorf, 1990, Vorschritt berichte VDI reihe 8,Meß-, Steuerungs- und Reglungstechnik nr 213.

20. Mauck, G.W. et al., *The meaning of the point of maximum oscillations in cuff pressure in the indirect measurement of blood pressure. Part II.*, Transactions of the ASME vol. 102, February 1980.

21. Merritt, H.E., *Hydraulic control systems*, John Wiley & Sons Inc, 1967

22. Min, J.L., Schrage J.J., *Ontwerpen vam analoge en digitale regelsystemen*, 3rd impression, B.V. Uitgeverij Nijgh & Van Ditmar, Den Haag, 1982.

23. Nauta, S., *RS232C Driver*, version 2.0, Dräger Medical Electronics, Best, 1990.

24. Posey, J.A. et al., *The meaning of the point of maximum oscillations in cuff pressure in the indirect measurement of blood pressure. Part I.*, Cardiovascular research centre bulletin, 1969.

25. Ramsey, M., *'Blood pressure monitoring: automated oscillometric devices'*,1991.

26. Schildt, H., *Using Turbo C 2nd edition*, McGraw-Hill, Berkeley, 1989, Programming series.

27. Ward, P.T., Mellor S.J., *Structured development for real-time systems*, 5th impression, Prentice-Hall Inc., New Jersey, 1985, Volume 1: Introduction & Tools.

28. Ward, P.T., Mellor S.J., *Structured development for real-time systems*, Yourdon Inc., New York, 1985, Volume 2: Essential modeling techniques.

29. Ward, P.T., Mellor S.J., *Structured development for real-time systems*, 2nd impression, Prentice-Hall Inc., New Jersey, 1986, Volume 3:Implementation modeling techniques.

30. Wesseling, K.H., *Eén is ongeveer gelijk aan twee, en wat er aan te doen, intreerede*, Technische Universiteit Eindhoven, 1992

31. Wouters, P., *Service protocol user guide*, version 0.1, Dräger Medical Electronics, Best, 1994.

32. Wouters, P., *Software environment of Dialog 2000 project*, Dräger Medical Electronics, Best, 1996.

33. *RTKernel Real-Time Multitasking Kernel 4.0 für C/C++ Benutzerhandbuch*, On Time Informatik GmbH.

34. *Turbo C users guide version 2.0*, Borland International, Inc., 1988.

Dräger

Business Unit Monitoring,
Research and Development.

Technische Universiteit tu/e Eindhoven

Faculty of Electrical Engineering,
Department of Measurement and Control Systems (MBS),
Section of Medical Electrical Engineering

Appendix

to accompany the report:

Software controlled proportional valve for NIBP measurements

Author: Jeroen Heesakkers
graduate University of Technology Eindhoven

Written by order of: prof. dr. ir. P.P.J. van den Bosch
dr. ir. P.J.M. Cluitmans
University of Technology Eindhoven

A.W. Bodbijl
Dräger

Date: August 1997

Place: Best

Appendix A Dialog 2000 technical information

Technical Data

ECG	
Input	Type CF, for ECG cable with 3 or 5 electrodes
Leakage	1, 2, 3 or I, II, III, aVR, aVL, aVF, V
Amplification	4; 2; 1; 0.5; 0.25 mV/cm
Automatic detection and display of pacemaker impulses	
Defibrillation protected	
Heart rate	
Input	ECG, type 1, type2, SpO: automatic switchover
Displays	audible and visual; level adjustable with rotary knob
Measuring range	20 - 300 beats/min; mean value over 15 beats
Detection of asystole	no ECG for 6 s; message with display of duration
Respiration	
Input	via ECG electrodes; measurement of impedance
Display	breathing frequency value; mean value over 5 tidal strokes
Measuring range	0 - 150 tidal strokes/min
Invasive blood pressure	
Input	for two sensors; insulated; type CF
Zero calibration	semi-automatic
Measuring range	-10 to +320 mmHg
Units of measurement	mmHg or kPa
SpO_2	
Input	floating; type CF
Measuring range	0 - 100%
Plethysmogram	automatic control of amplitude
C-Lock™; NIBP lock	
NIBP	
Measuring method	oscillometric
Measuring range	10 - 290 mmHg Adult; 5 - 150 mmHg child
Units of measurement	mmHg or kPa
Max. cuff pressure	300 mmHg Adult; 150 mmHg child
Max. measuring time	90s adult; 60s child
Measuring intervals	Manual; automatic every 2, 5, 10, 15, 30, 60 minutes
Temperature	
Input	for two sensors, insulated, type CF
Measuring range	0.0 to 45.0 °C
Trend	
Number of trend parameters	max. 14 (depending on model)
Display of trend waveforms	any 2 at the same time
Time window	1.5; 3; 6 hours; automatic adjustment
Applications	
Infant mode	neonates, newborn babies and children up to 5 years
Adult mode	patients over 5 years old
Voltage	
NiCd pack	Operating time 3 hours; fast charge 2 hours
Mains operation	input voltage 85 to 264 V AC (47 to 440 Hz) with separate power pack
DC Voltage	10.5 to 30.0 DC with optional DC/DC converter
Dimensions/Weight	
Monitor	LCD 115 x 86 mm; resolution 320 x 240 pixel Pixel size 0.36 mm²
Machine dimensions	215 x 125 x 215 mm (W x H x D)
Weight	4.0 - 4.2 kg (depending on model)

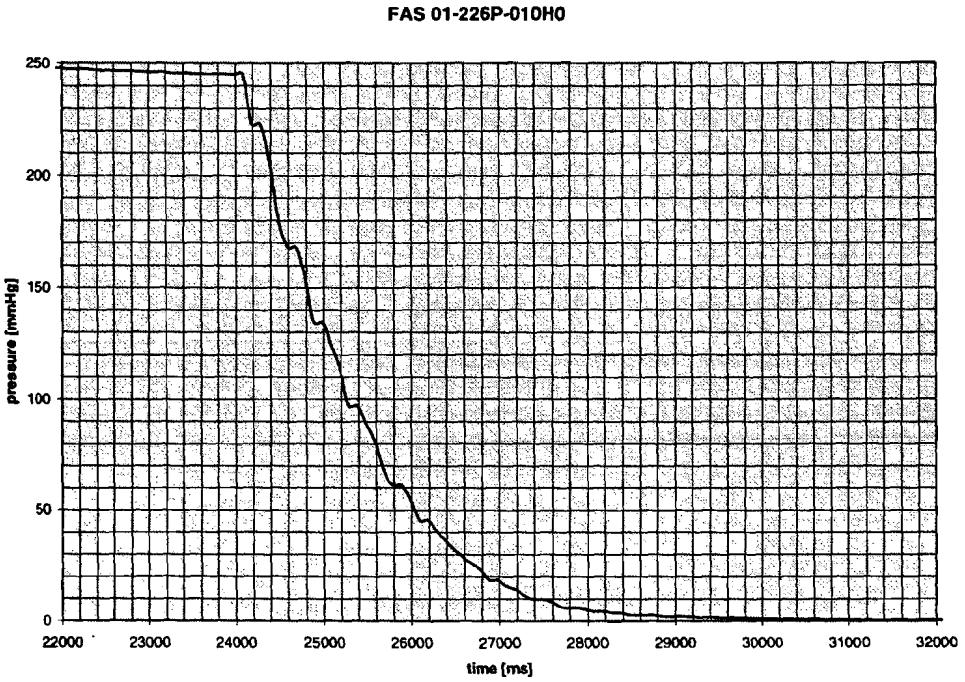
Order List

Dialog 2000/1 ECG, NIBP, 2 x Temp	
White model	5706400
Orange model	5706401
Dialog 2000/2 ECG, NIBP, SpO2, 2 x Temp	
White model	5706200
Orange model	5706201
Dialog 2000/3 ECG, NIBP, SpO2, 2 x invasive pressure, 2 x Temp	
White model	5706100
Orange model	5706101
Accessories set 1	5770038
consisting of:	
ECG standard cable; 3-core, 1.5 m	5720033
ECG patient cable; 3-core	5720030
NIBP hose short (1.5 m)	5720420
Adult cuff (25 - 35 cm)	5721040
SpO_2 standard cable (1.5 m)	5720085
Finger sensor, Durasensor DS-100A	5720072
Accessory set 2	5770039
consisting of:	
ECG standard cable; 3-core, 1.5 m	5720033
ECG patient cable; 3-core	5720030
NIBP hose short (1.5 m)	5720420
Adult cuff (25 - 35 cm)	5721040
Power supply unit	5720121
DC/DC converter	5720122
Dialog 2000 carrying case	5721910
Bracket for mounting monitor securely in vehicles	8412069

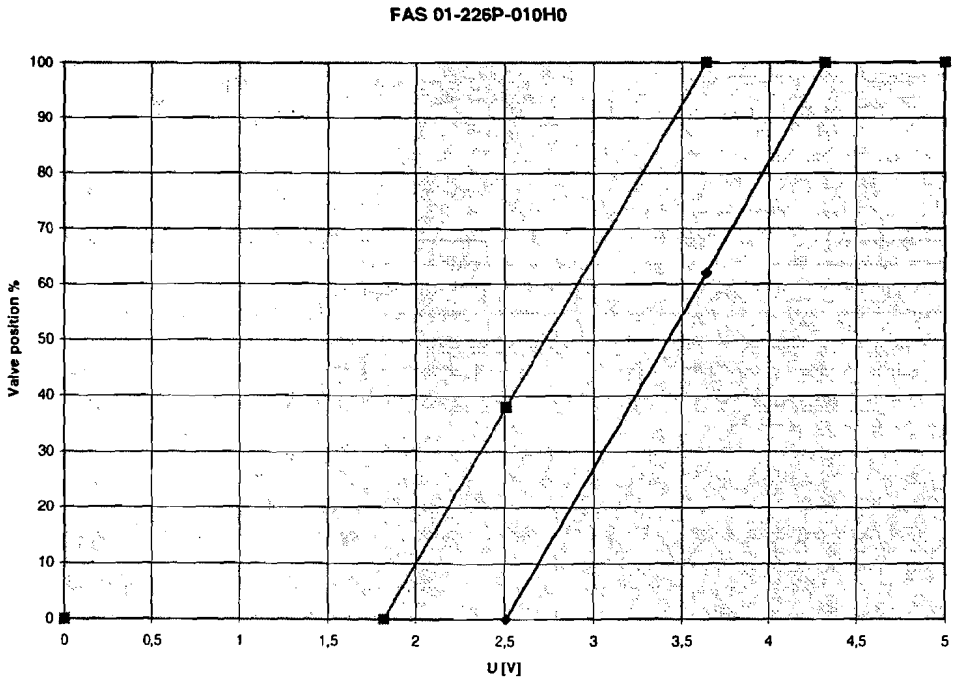
Drägerwerk
Aktiengesellschaft
Moislinger Allee 53 - 55
D-23542 Lübeck
Tel. (0451) 8 82-46 11
Fax (0451) 8 82-38 66

Appendix B Electrical valve characteristics

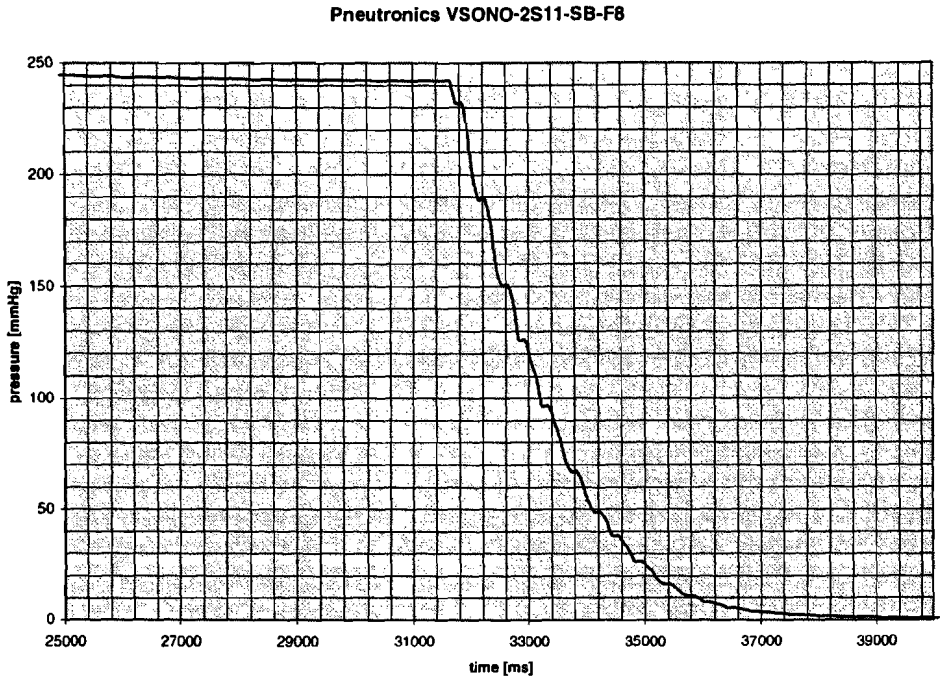
The data of the FAS 01-226P-010H0 proportional valve for 5V and the data of the Pneutronics VSONO 2S11-SB-F8 proportional valve for 5V, is also stored in the file 'ValveData.XLS'. The first graph shows the deflation of a 500 mL cuff, inflated to 250 mmHg. In this case the valve was fully opened.



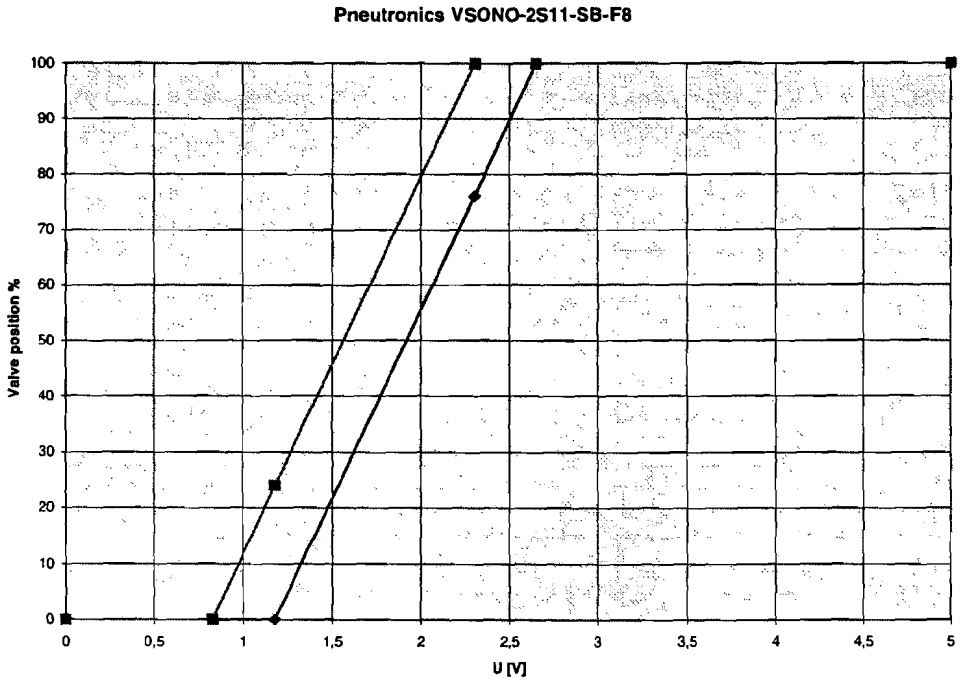
The measured hysteresis characteristic is shown below:



For the Pneutronics valve the deflation of a 500 mL cuff is according to:



The hysteresis is measured and shown below.



The data sheets of both valves follow on the next 2 pages.

microprop

micro-Magnetventil
2 Wege NC
proportional

fluid automation systems

Allgemeines

Die miniaturisierten Proportional-Magnetventile aus der Reihe microprop, erlauben die proportionale Regelung des Durchflusses von Flüssigkeiten oder Gasen anhand der Strom-, bzw Spannungsänderung.

microprop-Magnetventile bieten eine hohe Leistung in Hinsicht auf Durchfluss, Ansprechzeit, Hysterese und Wiederholgenauigkeit.

microprop-Magnetventile sind für die Ansteuerung aus SPS's oder Microprozessor gesteuerten Systemen ausgelegt.

microprop-Magnetventile gibt es in verschiedenen Ausführungsvarianten in Bezug auf elektrische Anschlüsse und Fluidteil.

Bitte rückfragen.

Flansch

fig.1

M5

fig.2

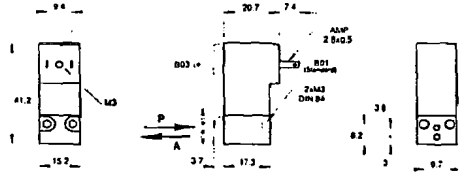
Technische Daten

Nennweite:	0,8 bis 2,0 mm
Druckbereich:	0 bis 12 bar
Durchfluss (Kv):	0,2 bis 0,9
Ansprechzeit: (0 bis 90% des maximalen Durchflusses)	10 bis 15 ms
Schaltspiele:	3000 cpm bei Anwendung on/off 0-100%
Temperaturbereich:	Umgebung: -10°C +50°C Medium: -10°C +30°C
Medienbeständigkeit:	Gase und Flüssigkeiten kompatibel mit den gewählten Werkstoffen
Mit Medium in Berührung kommende Materialien:	vernickelter Messing, Edelstahl 303 (DIN 1.4305) & 430 (DIN 1.4016), Polyamid 6/6, Dichtungsmaterial
Anschluss:	Flansch oder M5
Einbaulage:	beliebig
Befestigung:	2 Schrauben M3
Sonderausführung:	Auf Anfrage
Gewicht:	50 g (je nach Ausführung)

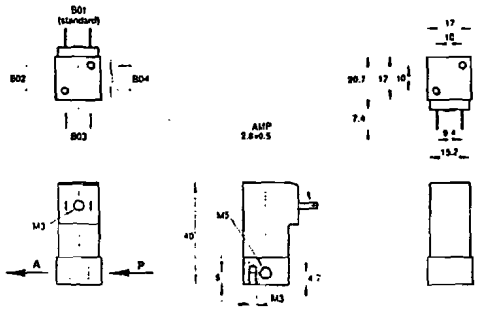
Elektrische Daten

Spannungen:	6; 12; 24 V DC
Leistungsaufnahme:	siehe Rückseite
Elektrischer Anschluss:	2 Steckfahnen oder 2 Litzen
Isolationsklasse:	F155
Schutzart:	IP 61 gemäss DIN 40050 IP 65 auf Wunsch
Einschaltdauer:	100% ED
max. Spulentemperatur bei 100% ED:	1W 45°C bei 20°C Umgebung 2W 60°C bei 20°C Umgebung
Sonderspannung:	auf Anfrage

Flansch fig.1



M5 fig.2



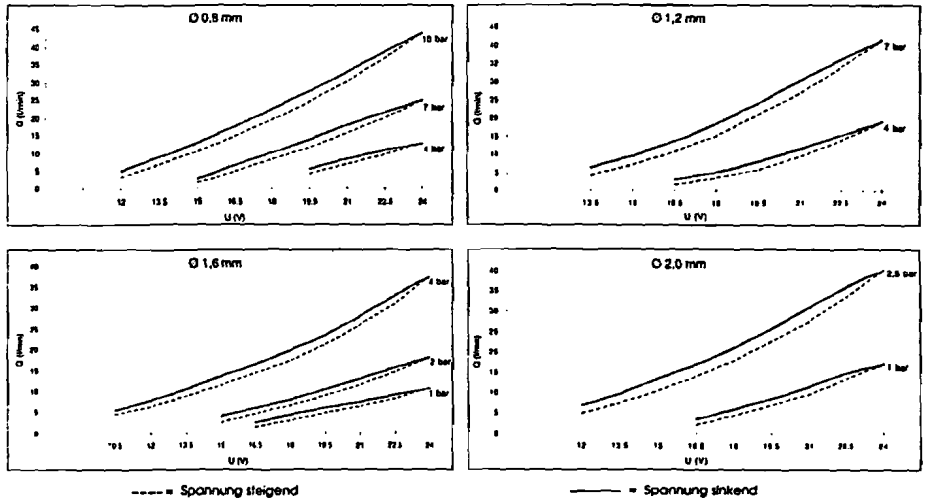
microprop

micro-Magnetventil
2 Wege NC
proportional

fluid automation systems

ANSCHLUSS	FIG. Nr.	NENNWEITE	KV-WERT	DRUCK-BEREICH		LEISTUNGS-AUFNAHME	BESTELLNUMMER
		mm	Kv MAX	bar MIN	bar MAX	watt DC =	VENTIL / WERKSTOFFE: VENTILGEHÄUSE / DICHTUNG / ZUSATZ
Flansch	1	0,8	0,2	0	12	2	01-216P-01- 2 . . 0 1 5 6
Flansch	1	1,2	0,3	0	7	2	01-216P-02- 2 . . 0 1 5 6
Flansch	1	1,6	0,5	0	4	2	01-216P-03- 2 . . 0 1 5 6
Flansch	1	2,0	0,9	0	2,5	2	01-216P-04- 2 . . 0 1 5 6
M5	2	0,8	0,2	0	12	2	01-216-001- 2 . . 0 1 5 6
M5	2	1,2	0,3	0	7	2	01-216-002- 2 . . 0 1 5 6
M5	2	1,6	0,5	0	4	2	01-216-003- 2 . . 0 1 5 6
M5	2	2,0	0,9	0	2,5	2	01-216-004- 2 . . 0 1 5 6

Spannung / Durchfluss Diagramm für 24 V DC :



----- = Spannung steigend ——— = Spannung sinkend

Auf Anfrage

1 - Elektrischer Stecker : Siehe spezielle Datenblätter.
2 - Elektrischer Anschluss durch Litzen : Auf Anfrage, Länge ca. 300 mm.

Bemerkungen

Für aggressive Medien gibt es verschiedene Ventilkörper und Dichtungen.
Für Sondermagnetventile bitte rückfragen.

Bestellbeispiel

01-216-003-20 24V DC
microprop, 2 /2 Wege NC, proportional, Anschluss M5, NW 1,6mm, Ventilgehäuse Messing vernickelt, Dichtungsmaterial Buna, Spannung 24V DC, Spulenstellung B01.

Werkstoffe

Ventilgehäuse		Dichtung	
2	Messing	0	Buna
		1	Viton
		5	EPDM

Nominal Coil Resistance (Ohms)	Upper Limit		Lower Limit		Maximum Hysteresis (% of Full Shut Off Current)	Gain Limits (applicable for 20% to 80% of Full Flowrate)	
	Offset Current (mAmps)	Landing Current (mAmps)	Offset Current (mAmps)	Landing Current (mAmps)		Maximum (SLPM/Amp)	Minimum (SLPM/Amp)
11	50	231	27	134	15	72.39	15.45
23	35	160	19	93	15	104.11	22.22
47	25	115	13	67	15	144.79	30.9
68	21	95	11	55	15	175.81	37.52
136	15	69	8	40	15	242	51.64
274	11	50	6	29	15	334.39	71.35
547	8	36	4	21	15	468.83	100.04
1094	5	24	3	14	15	686.01	146.39

Table 4-2: Model 2 VSONO Performance Requirements

Nominal Coil Resistance (Ohms)	Upper Limit		Lower Limit		Maximum Hysteresis (% of Full Shut Off Current)	Gain Limits (applicable for 20% to 80% of Full Flowrate)	
	Offset Current (mAmps)	Landing Current (mAmps)	Offset Current (mAmps)	Landing Current (mAmps)		Maximum (SLPM/Amp)	Minimum (SLPM/Amp)
11	74	304	33	177	15	90.23	19.11
23	51	212	23	123	16	129.75	27.48
47	37	152	17	89	15	180.45	38.22
68	30	125	14	73	15	219.12	46.42
136	22	91	10	53	15	301.61	63.89
274	16	66	7	38	15	416.76	88.28
547	11	47	5	27	15	564.32	123.77
1094	8	33	4	19	15	855	181.11

Table 4-3: Model 3 VSONO Performance Requirements

7.3 Cleanliness

The VSONO wetted components shall be cleaned in accordance with Pneutronics Standard Operating Procedure, MSP-000003-002.

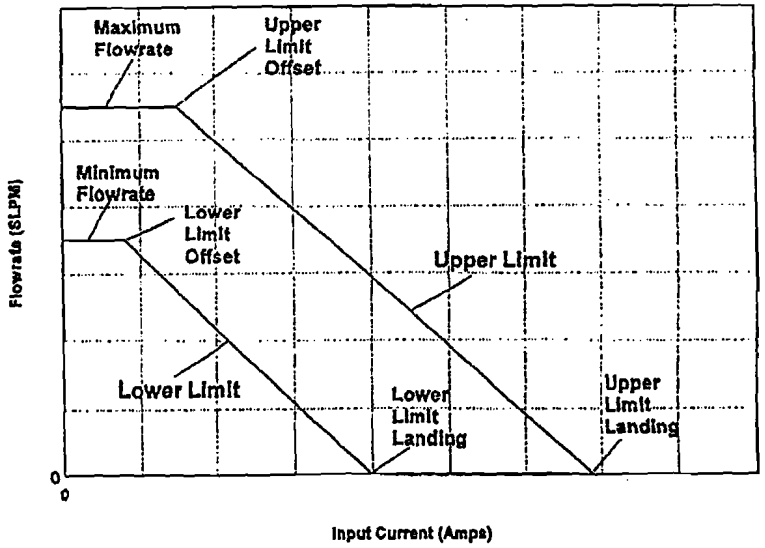
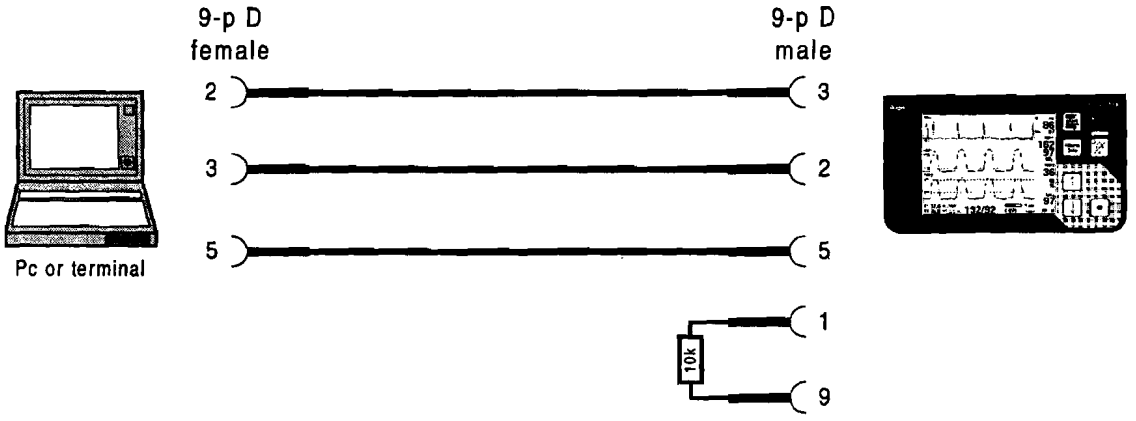


Figure 1: VSONO Calibration Limit Terminology

Appendix C Serial connection PC and Dialog 2000

It is possible to examine and change the internal settings of the Dialog 2000 and to obtain debugging information via the service port of a standard Dialog 2000. No special software version is needed. The service connector at the back of the Dialog 2000 contains a serial port (changed to 19200 baud, 8 bit, no parity), so it can communicate directly with a terminal (PC or a workstation). To enable the service protocol, a resistor of 10 K ohm must be present in to the connector at the Dialog 2000 side. See the picture below for the schematics of the cable between the Dialog 2000 and the terminal or PC.



By default, the output of the service protocol conforms to the standard as defined by the DWHL service department. To get more readable output for use with a terminal, <control> E must be pressed, which toggles between <LF> and <CR> <LF> end of line characters and echoes typed characters.

The service port of the Dialog 2000 recognizes the following commands:

? or H	Print a summary of available commands and identifiers.
<identifier>	Print the value of *<identifier>*. The value is displayed as decimal number. Identifiers can be listed with the "?" command.
<identifier> *<value>*	Set the value of *<identifier>* to <value>. Value can be decimal (e.g. 23) or hexadecimal (e.g. 0x12)
REG *<identifier>*	Report all subsequent changes of the value of *<identifier>*.
UNREG *<identifier>*	Stop reporting changes of *<identifier>*.
VERSION	Print software version etc.
DB_COMP *<n>*	Enable debugging output for software component *<n>*.
DB_TYPE *<n>*	Enable debugging output for type *<n>*.
DB_COMP	Print number of software component for debugging output.
DB_TYPE	Print number of type for debugging output.
GCS_BUFFERS	Print information about GCS buffer memory usage.
DIAGNOSTICS	Print the diagnostics logging data from EEPROM.

Other commands are:

<CR>	Repeats the last command.
<control>E	Toggle echo on/off. This is the first command to be given when using a terminal.
<control>R	Places the last command in the edit buffer.
<backspace>	Deletes the last character typed.

The identifiers that are supported by the Dialog 2000 are listed in the file `pm2000/service/srvcmnd.h`.

Appendix D Feed forward measurements; U~RC relationship

The measurements that were done to obtain the feed forward valve characteristics are stored in the directory feedforward/FASvalve. For each valve control signal, three measurements were done.

The measurements have the following naming convention:

Measurement file (ASCII):

U<valve control signal>_<measurement number>.DAT

example: ***U199_1.DAT***

The measurements are collected in a Microsoft Excel file:

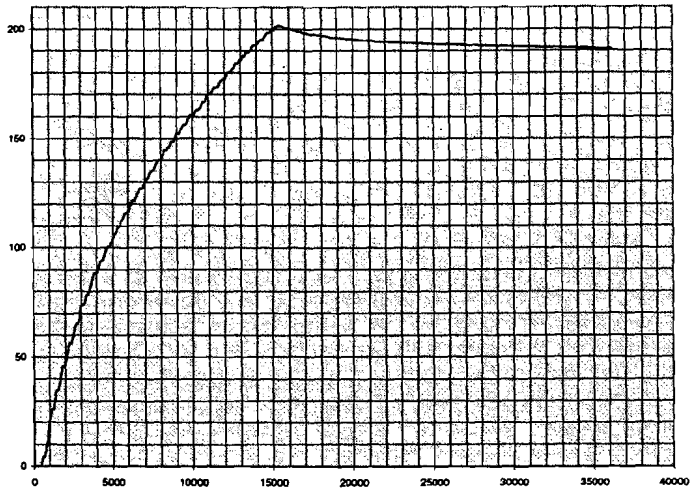
U<valve control signal>.XLS

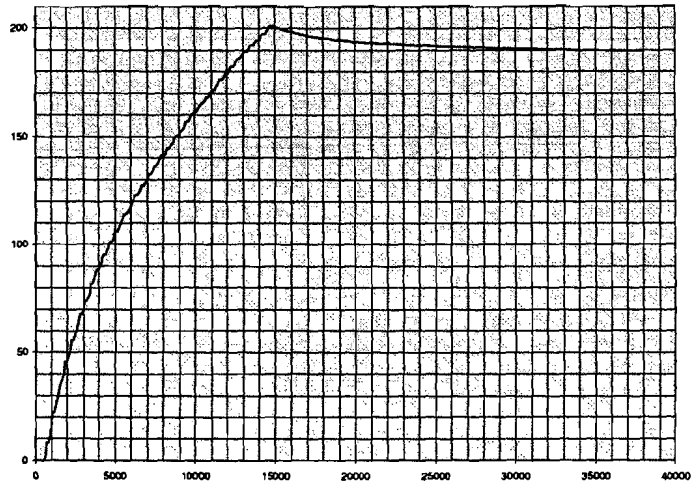
example: ***U200.XLS***

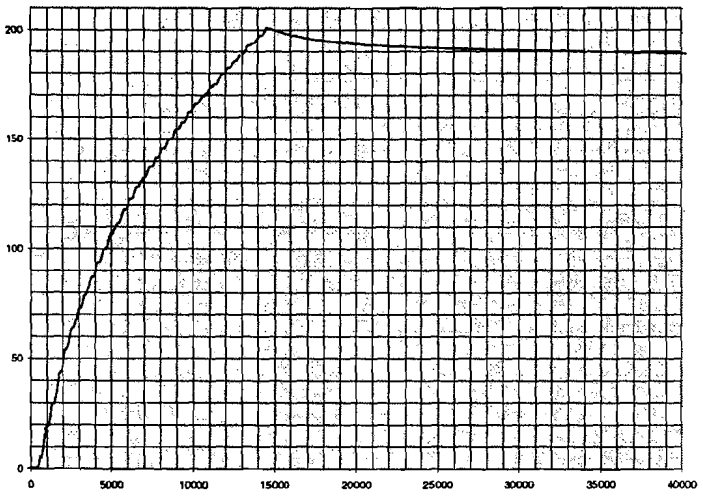
The RC~U relationship is calculated in the Microsoft Excel file:

RC~U.XLS

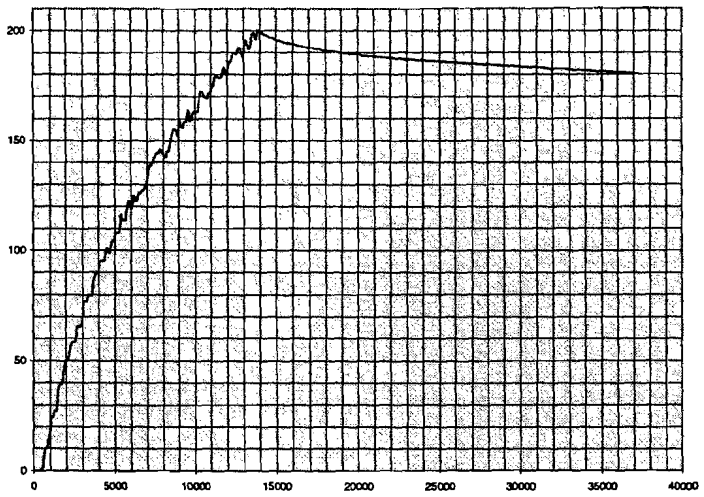
The next pages show the measurements from the Microsoft Excel files.

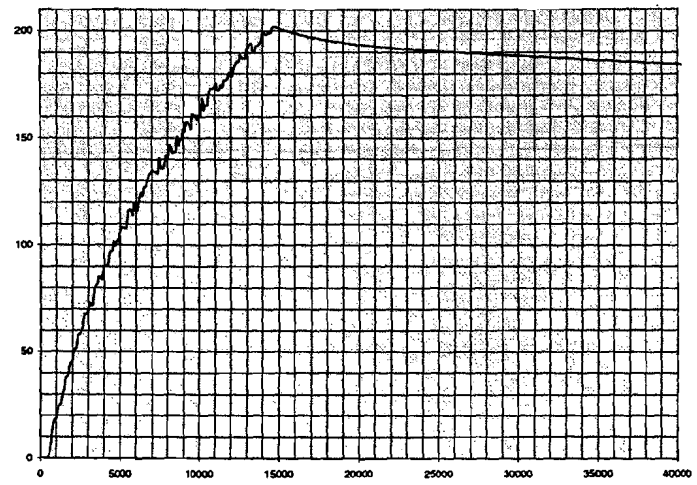


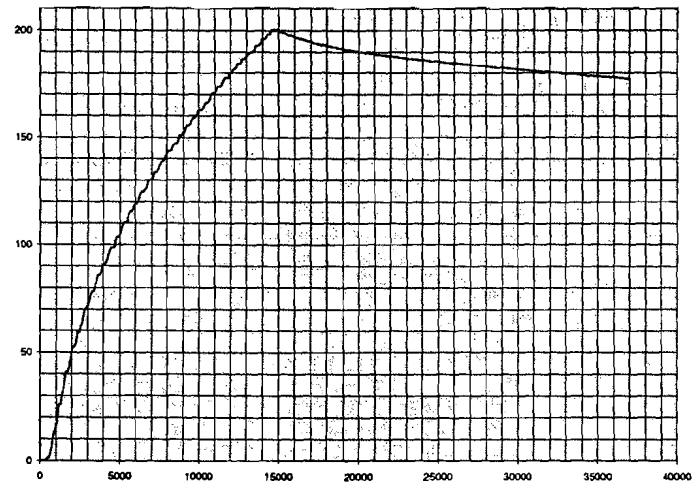




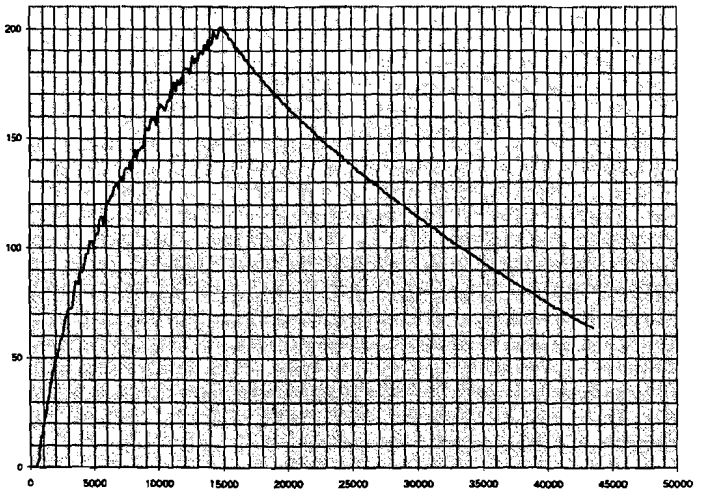
Three measurements for a valve control signal of 210. (U210.XLS)

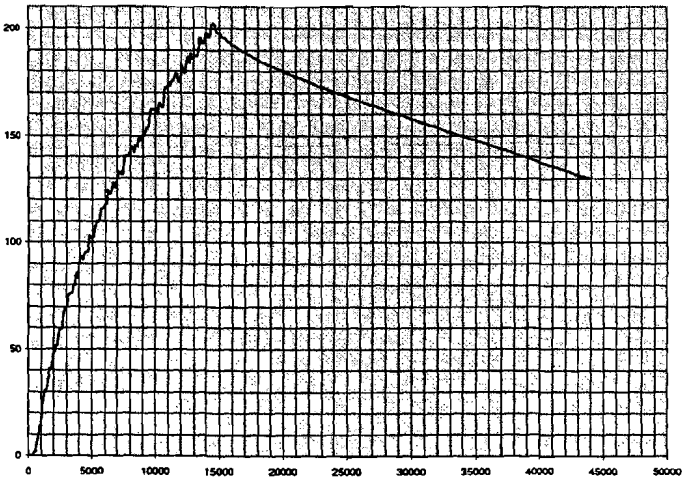


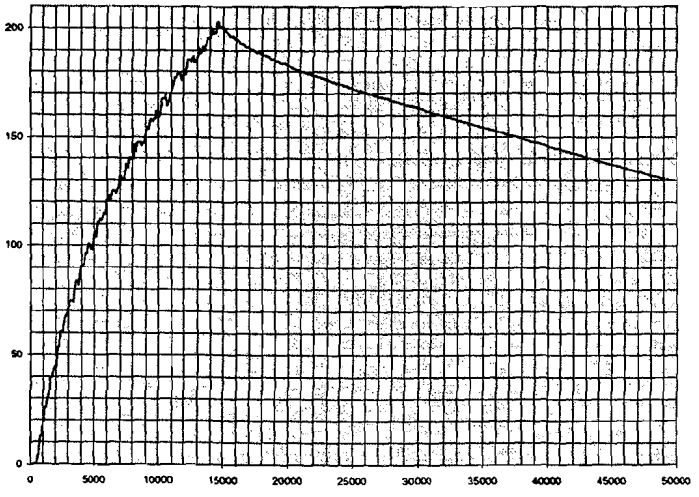




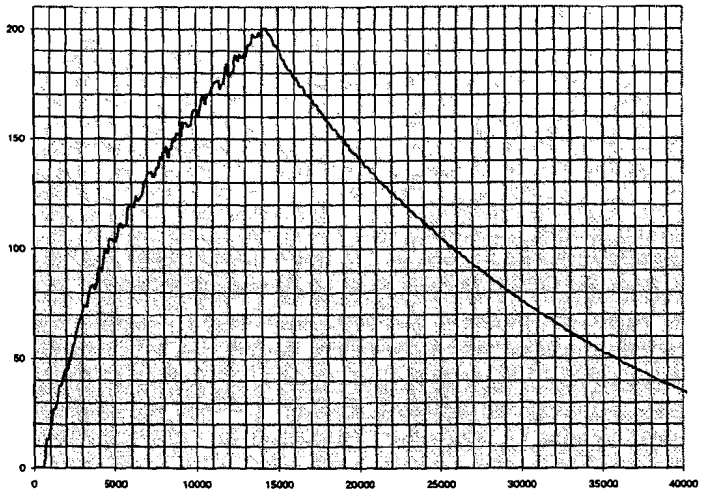
Three measurements for a valve control signal of 209. (U209.XLS)

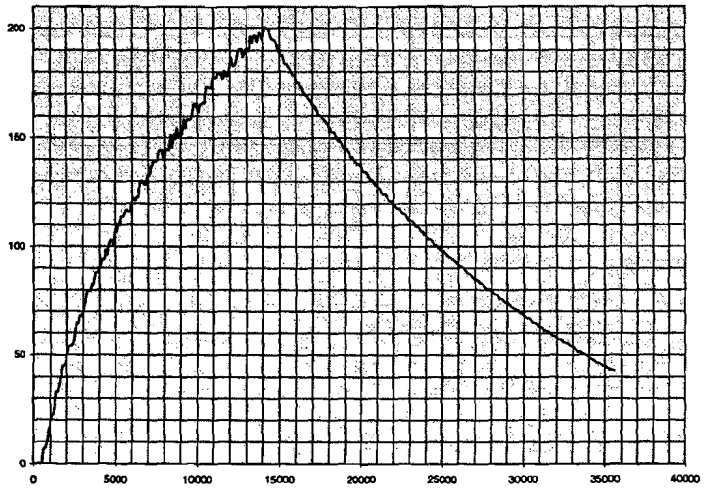


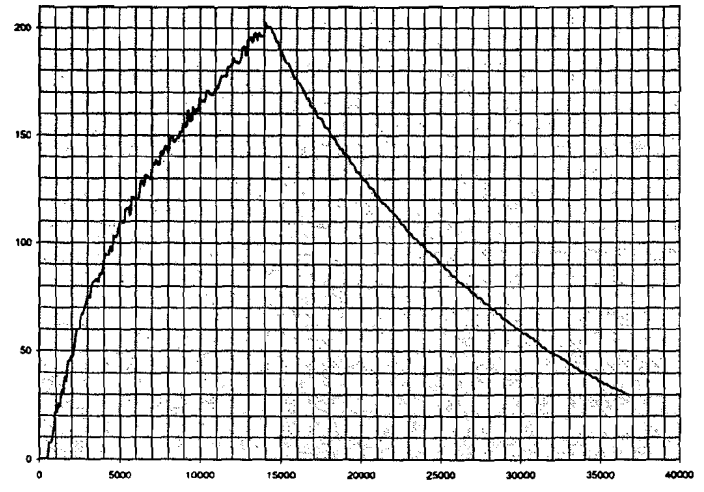




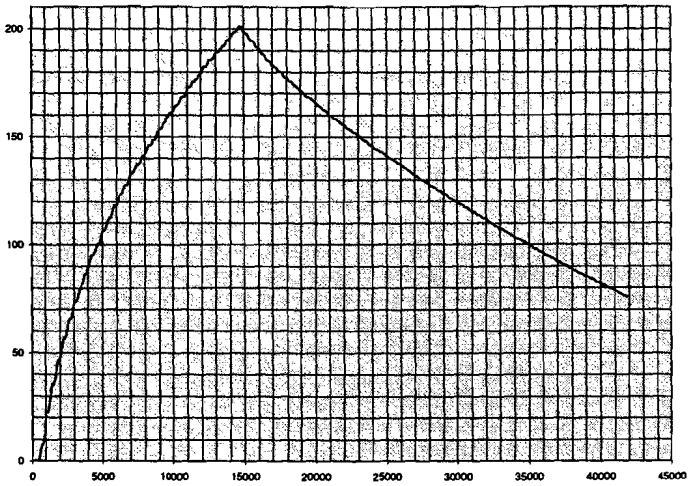
Three measurements for a valve control signal of 208. (U208.XLS)

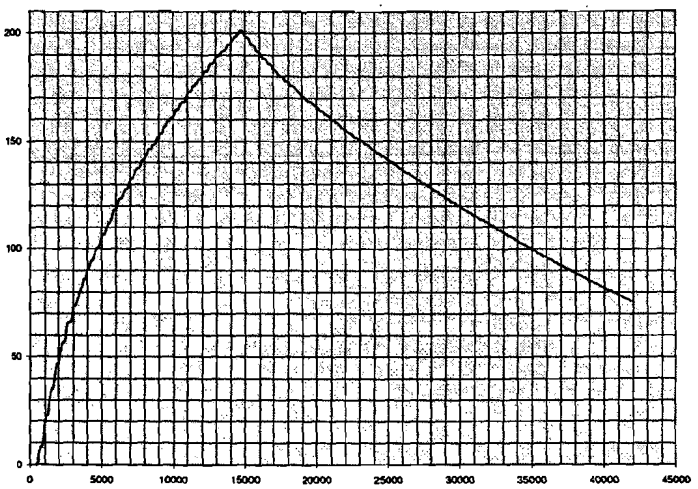


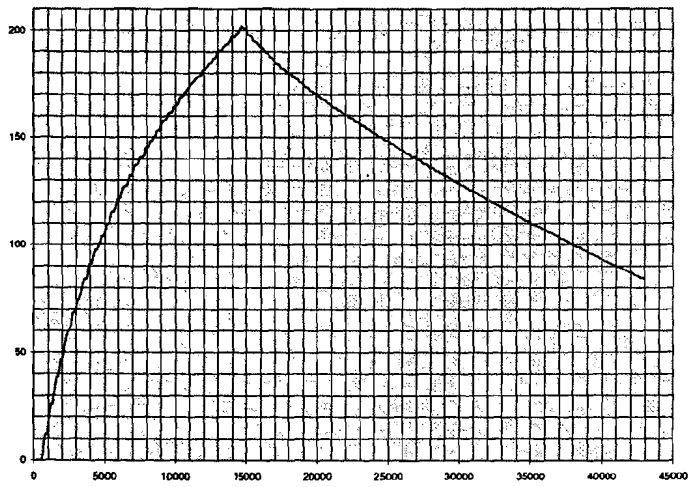




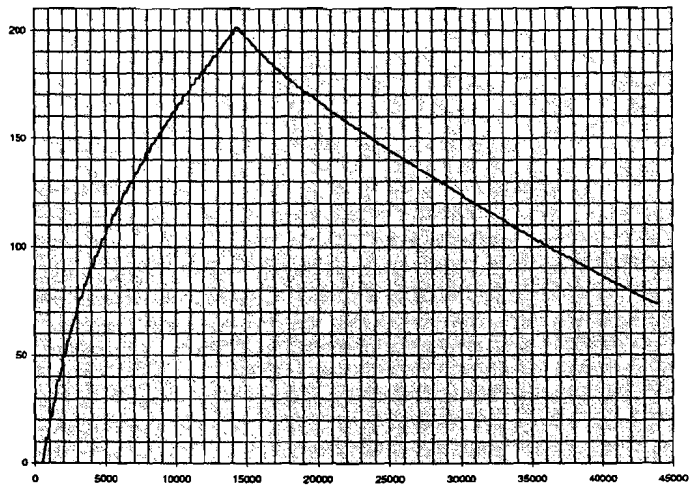
Three measurements for a valve control signal of 207. (U207.XLS)

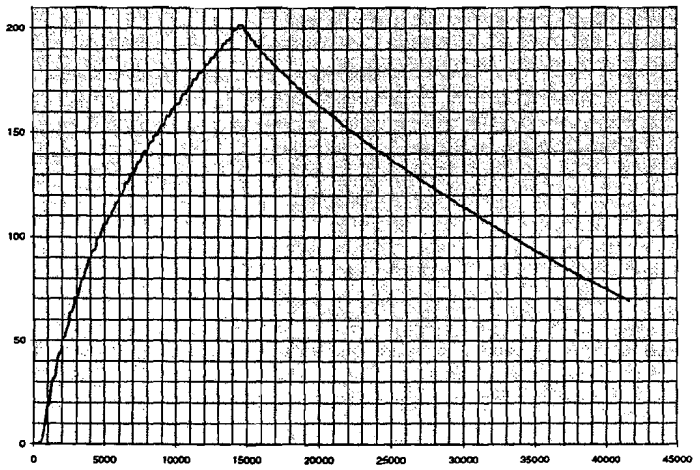


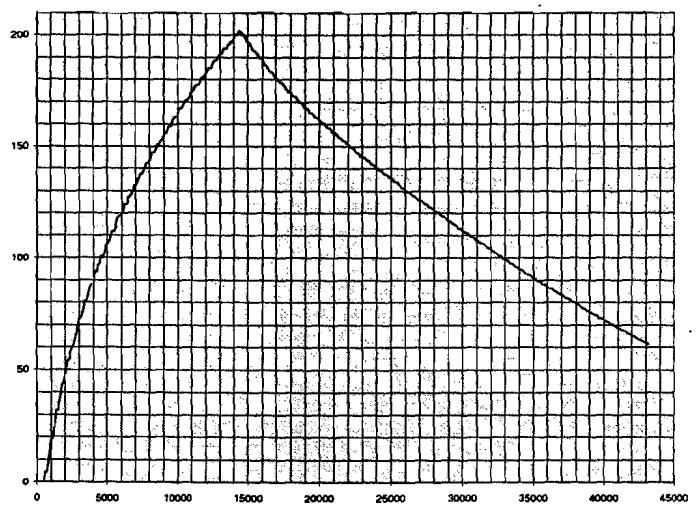




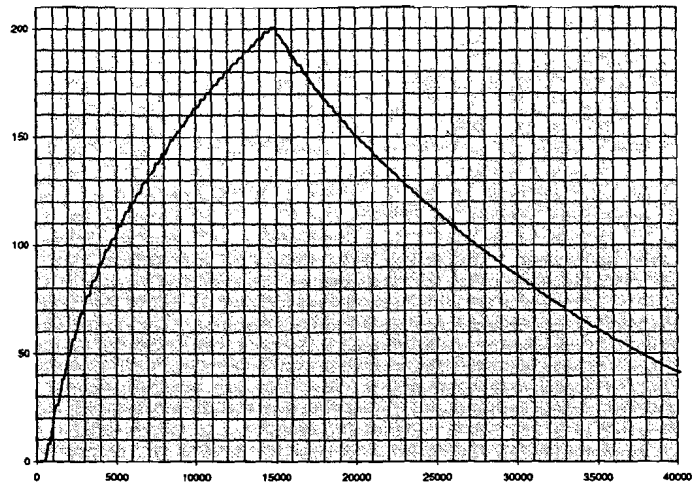
Three measurements for a valve control signal of 206. (U206.XLS)

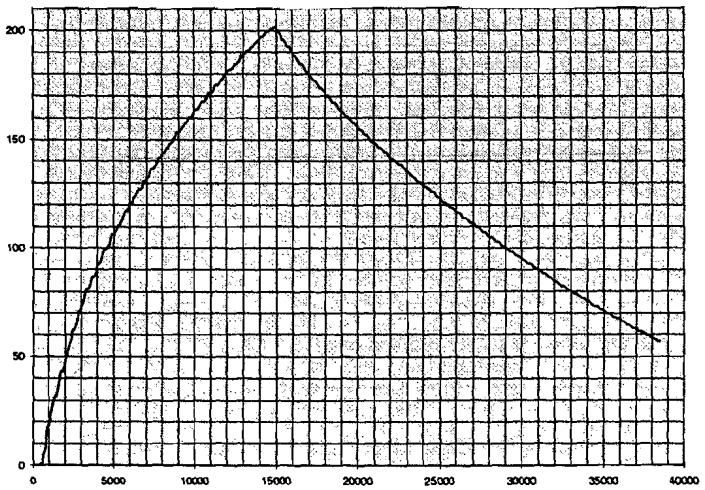


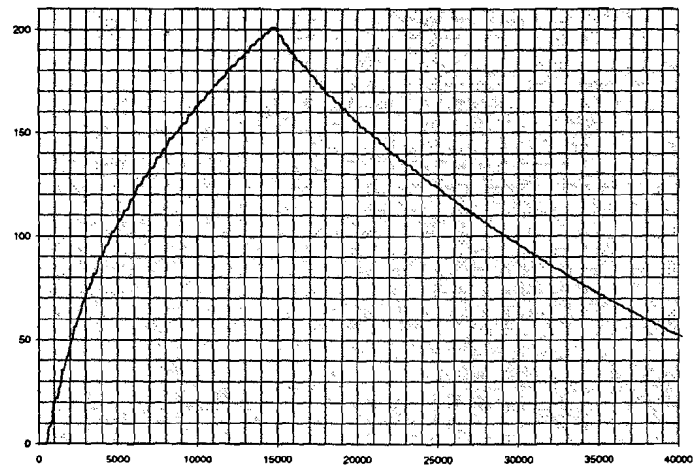




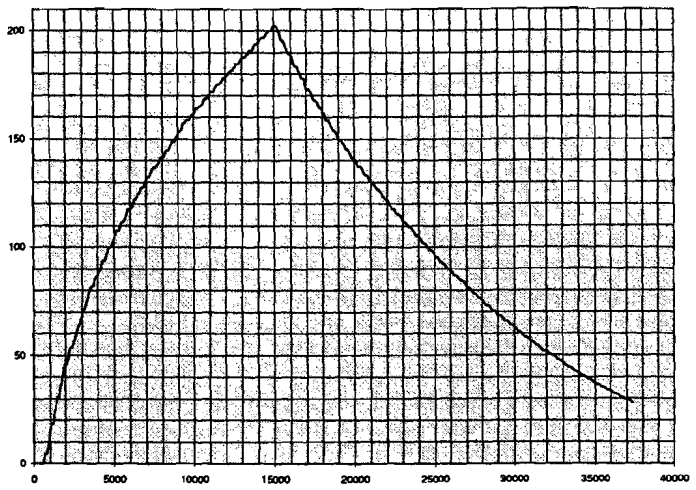
Three measurements for a valve control signal of 205. (U205.XLS)

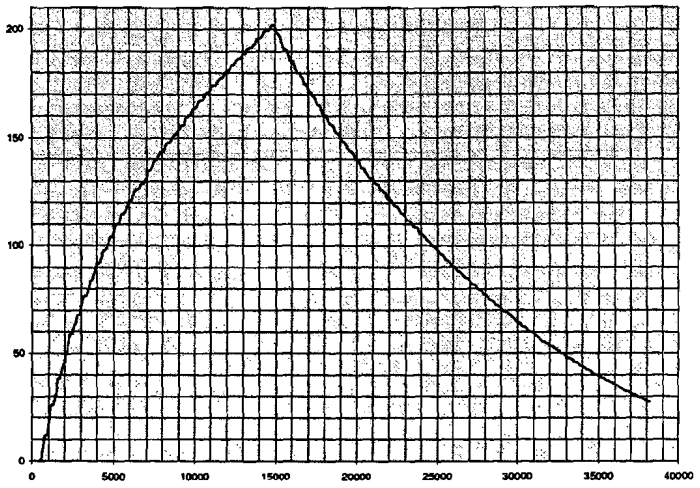


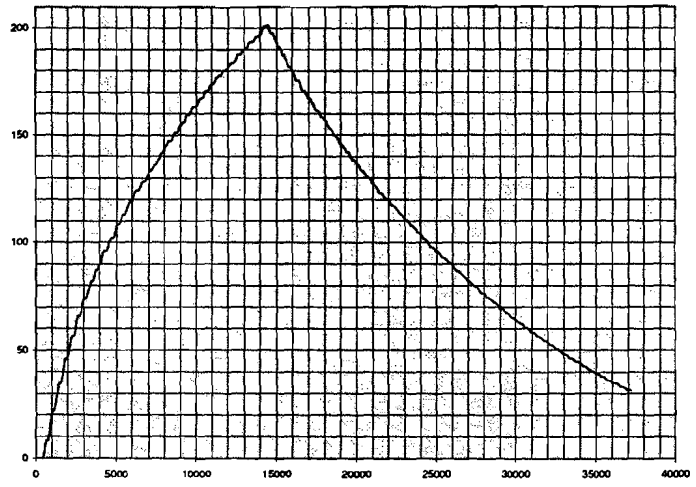




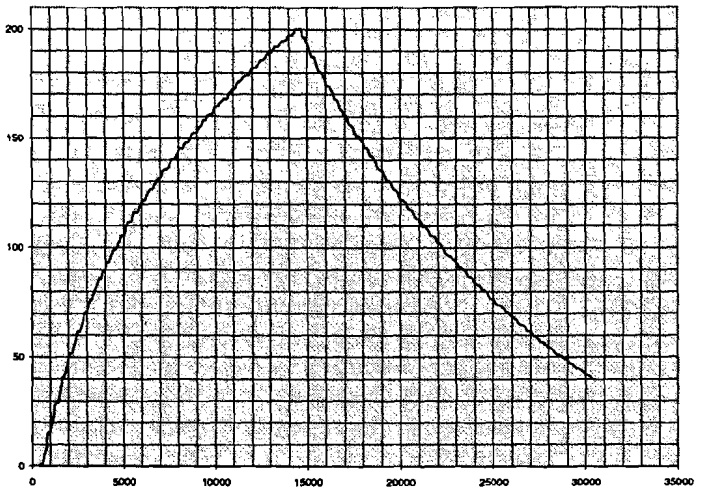
Three measurements for a valve control signal of 204. (U204.XLS)

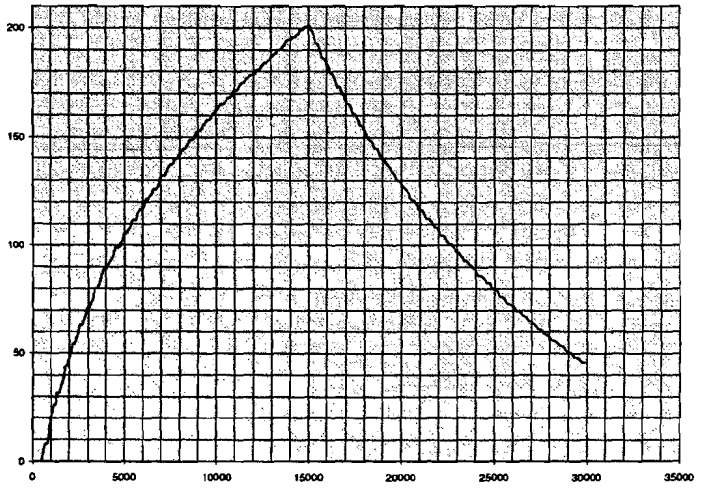


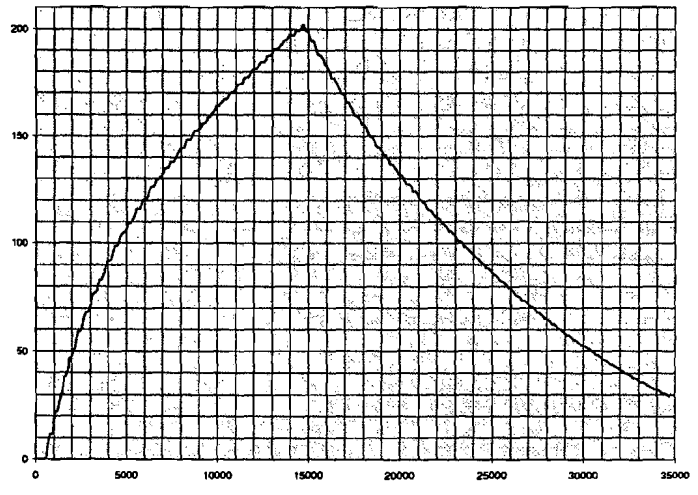




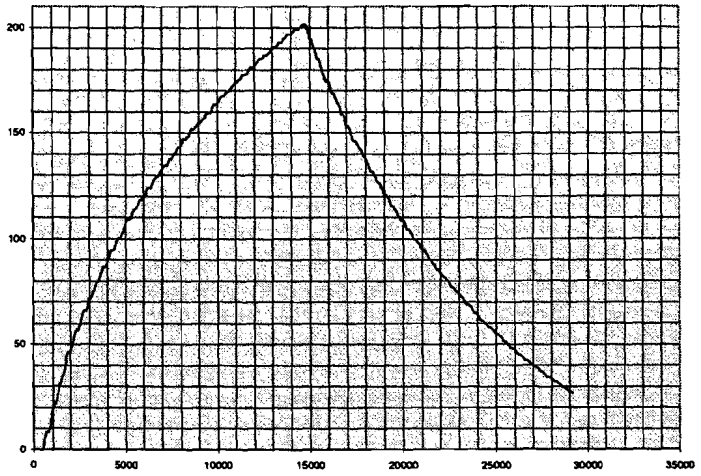
Three measurements for a valve control signal of 203. (U203.XLS)

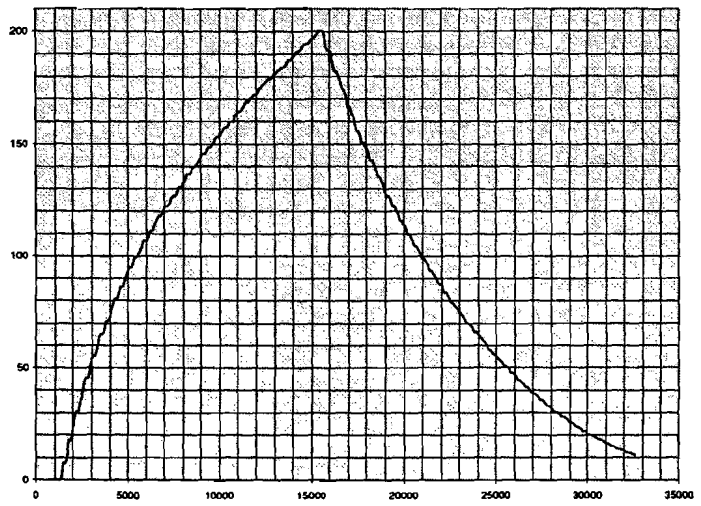


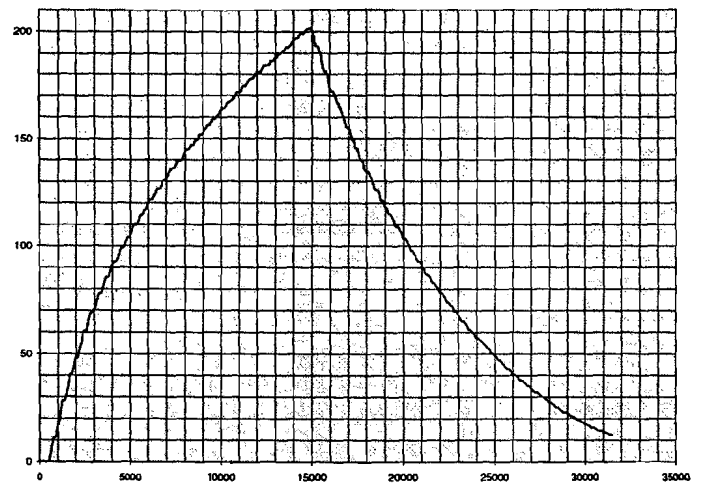




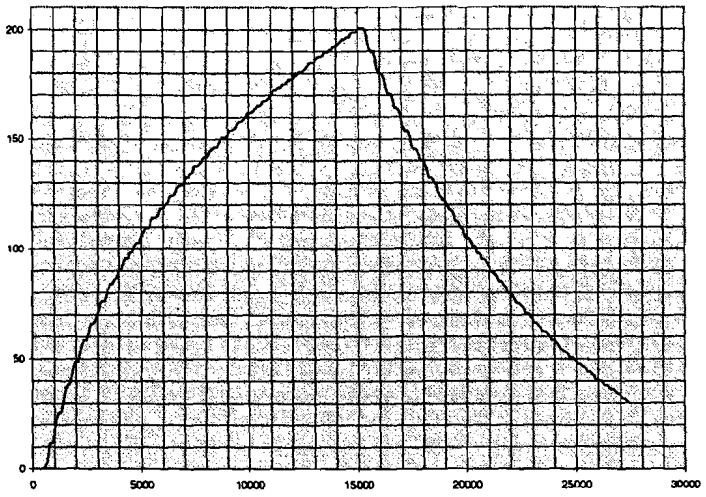
Three measurements for a valve control signal of 202. (U202.XLS)

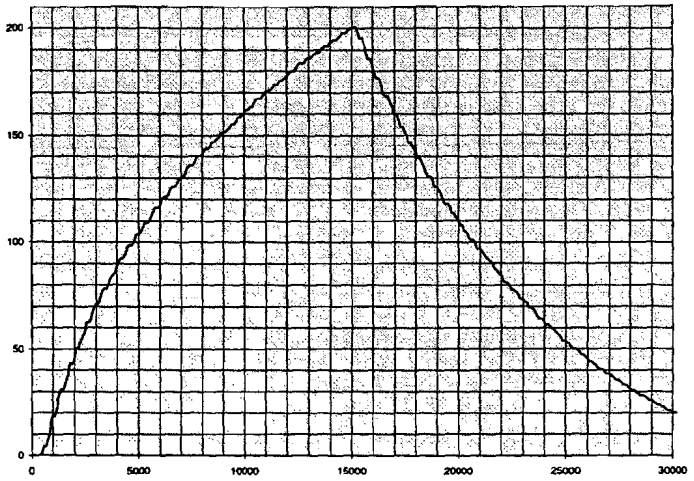


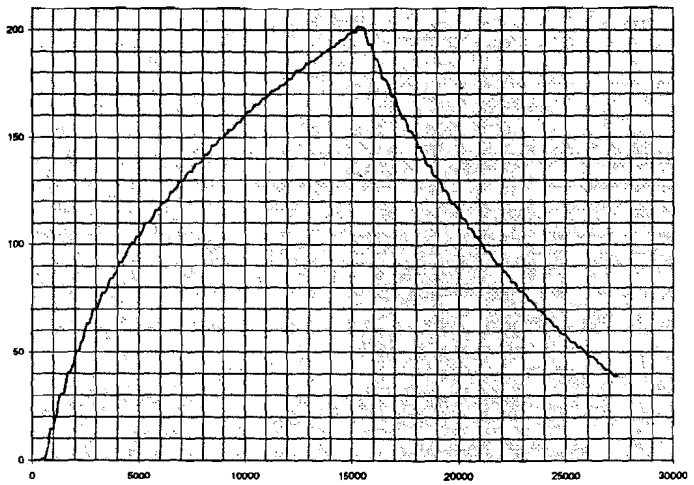




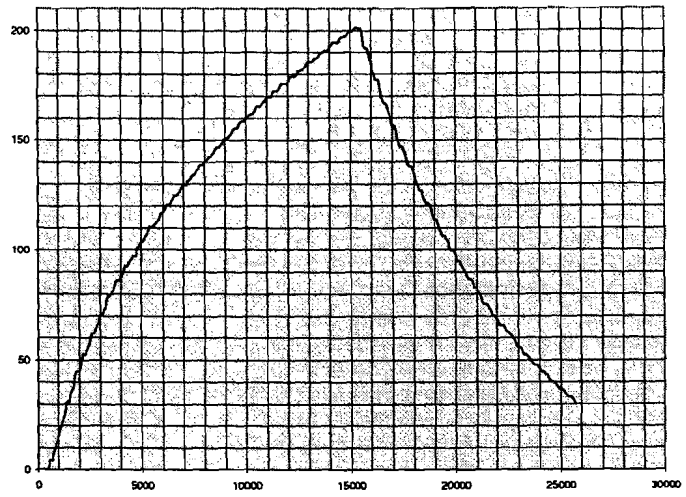
Three measurements for a valve control signal of 201. (U201.XLS)

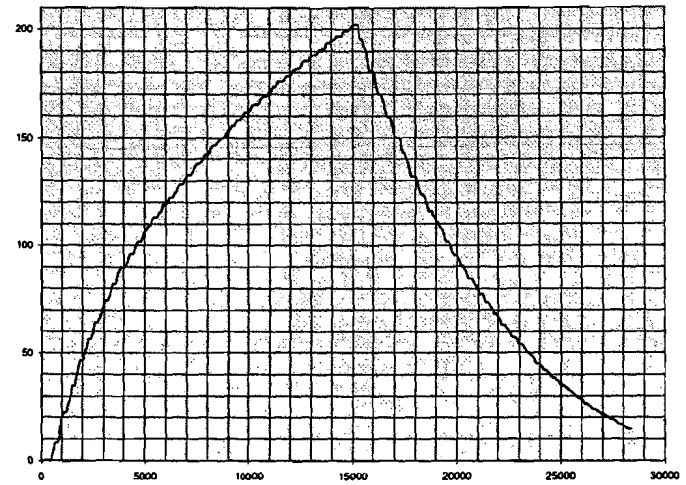


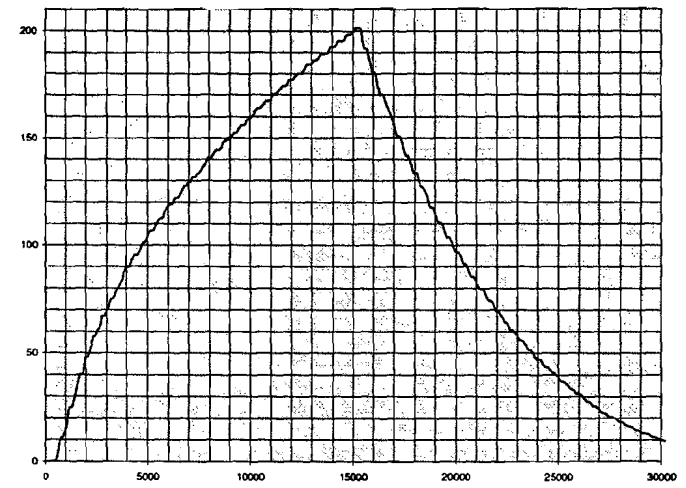




Three measurements for a valve control signal of 200. (U200.XLS)







Three measurements for a valve control signal of 199. (U199.XLS)

Appendix E Error calculations

Example 1:

The required sample frequency is calculated when a pressure error P_E of 1 mmHg is allowed.
Measurement properties are:
Initial pressure : 180 mmHg
End pressure : 15 mmHg
Measurement time : 45 seconds

This results in: $$\alpha = \frac{(P_{INITIAL} - P_{END})}{T_{MEASUREMENT}} = \frac{(180-15)}{45} \approx 3.67\, mmHg/s$$

For the minimum error of 1 mmHg this results in:

$$T = \frac{P_{E,\,MIN}(nT)}{\left(\frac{P_1}{T_{deflate}} - \alpha\right)}$$

$$T_{deflate} \approx 5s$$

This results in $T \approx 31ms$.
Because the maximum error is also 1 mmHg, The frequency for this error should also be calculated according to:

$$f_{s,\min} = \frac{1}{T} = \frac{\alpha}{P_{E,\,MAX}}$$

This leads to $T \approx 3.67s$.
The minimum sample frequency should be at least 32.33 Hz to assure a worst case error of 1 mmHg.

Example 2:

When a frequency of 10 Hz is used the worst case error can be calculated to be:

$$P_{E,\,MIN}(nT) = -\alpha \cdot T + \frac{P_1}{T_{deflate}} \cdot T = -3.67 \cdot 0.1 + \frac{180}{5} \cdot 0.1 \approx 3.2 mmHg$$

$$P_{E,\,MAX} = \alpha \cdot T = 3.67 \cdot 0.1 \approx 0.37 mmHg$$

Of course the performance decreases when delays are introduced. The limited resolution of data acquisition and signal generation can also degrade performance.

Appendix F Input file syntax and commands

Manual

to accompany the program cuffctrl, version 0.1

Program installation

The program runs on a 80486 based PC with an MSDOS operating system version3.0 or higher.
The program files can be installed in any directory.
This is done by copying the listed files to the directory:

cuffctrl.bat
meas.dat
leader.exe
dialog.exe

Program start

The program is executed when the command

cuffctrl

is entered at the commandline. The current working directory should be the directory in which the program files reside.

Script file

The script contains the description of a measurement that the program should execute.
The script is stored in the file

meas.dat

which should be located in the same directory as the program files (the working directory).

Script layout:

{[comment]}
{< initial setting>[comment]
{[comment]}}
{< measurement setting>[comment]
{[comment]}}

[ ] means optional.
< > means required.
{ } means repeated once or more times.

Script conventions:

All lines, which are no comment, have to be ended by a semicolon. ';'
Each separate command or parameter has to be separated by a colon. ':'

The last line in the file is a hash (#).

Script language explained:

comment

Comment starts with an asterisk and is followed by an arbitrary number of characters. The program ignores all characters beyond the asterisk on the same line.

example:
* this is comment

initial setting

Initial settings are:

<measurement name>[:"MEASUREMENT NAME"]<;>
<k value:FLOATING POINT NUMBER;>
<ti value:FLOATING POINT NUMBER;>
<td value:FLOATING POINT NUMBER;>

Values that are out of range will be automatically set to the maximum or minimum value in the specific range.

example:
measurement name:Meting 500ml;
k value:1.75;

measurement setting

Measurement settings are:

<display trace:TRACE NUMBER:VISIBILITY:ACTIVATION CONDITION;>
<inflate:INITIAL PRESSURE:ACTIVATION CONDITION;>
<deflate:DEFLATE TIME:END PRESSURE: ACTIVATION CONDITION;>
<set inflate actuator:INFLATE ACTUATOR VALUE;>
<set deflate actuator:DEFLATE ACTUATOR VALUE;>
<set dump actuator:DUMP ACTUATOR VALUE;>
The parameters (printed in capitals) have the following meaning:

TRACE NUMBER can be any number between 0 and TRACERANGE.
In Version 0.1: TRACERANGE=4.

VISIBILITY can be
'1' (visible) or

'0' (invisible).
Note that invisible traces will also be stored.

ACTIVATION CONDITION can be
'immediate' (immediate execution of the command)
'timed:TIME' (execution after a lapse of TIME seconds)
'status:STATUS' (execution after the status condition STATUS is reached)

In Version 0.1 the following values are implemented: STATUS =
554 (cuff is inflated to specified initial pressure)
557 (specified deflate time is reached)
559 (the program is idle)

INITIAL PRESSURE can be any integer value within 0 and 300 mmHg.

DEFLATE TIME can be any number of seconds between 0 an 180.

END PRESSURE can be any number between 0 and 300 mmHg.

INFLATE ACTUATOR VALUE can be any number between 0 and 255.

In Version 0.1:
0 == off
other == on

DEFLATE ACTUATOR VALUE can be any number between 0 and 255.

DUMP ACTUATOR VALUE can be any number between 0 and 255.

In Version 0.1:
0 == open
1 == closed

Runtime commands

Runtime commands are used to control the script execution by the user.
At runtime the following keys can be used:

start scriptfile	s
write measurement data to disk and erase the previous data	w
freeze the measurement data display (toggle function)	f
pause the script at the next untimed line	p
show display markers (toggle function)	m
Move marker 1 left	{
Move marker 1 right	}
Move marker 2 left	[
Move marker 2 right	]
Next trace marker	+
Previous trace marker	-
Quit to the operating system	q

Implementation choices:

In version 0.1 there are 4 trace available.
The traces that can be visible on the screen are:

Trace 0 (white):	cuffpressure
Trace 1 (yellow):	controller output (analog valve position)
Trace 2 (green):	controller error
Trace 3 (red):	controller sample time points

The trace can be used to select one of the traces. Trace marker 1 displays the measurement sample value at the location of the marker. Trace marker 2 displays the measurement sample value at the location of the marker and the time difference (seconds) between marker 1 and marker 2. For a more accurate time difference the results should be saved in a file (accuracy: milliseconds).

Results file

Measurements can be stored in a file with the name

results.dat

the results file stores:

Measurement name:
Measurement time:
Measurement date:
K value :
Ti value :
Td value :

Samples

For each sample is stored:

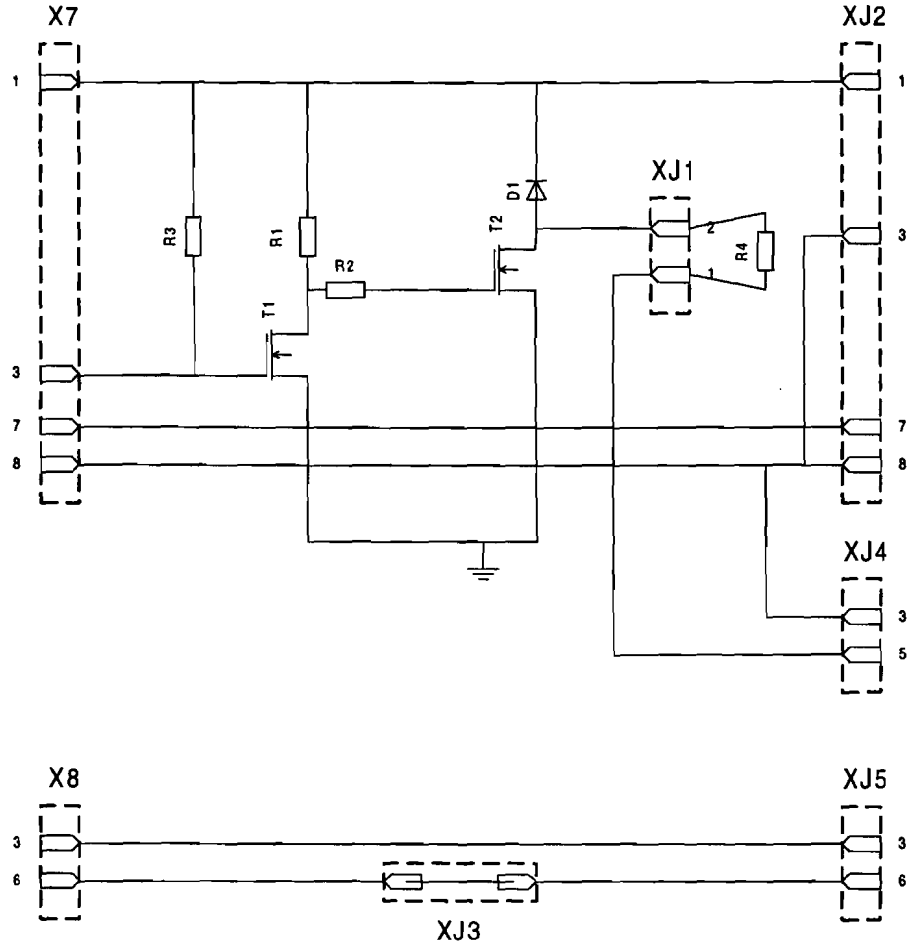
samplenumber
sample time
status
inflate actuator position
deflate actuator position
dump actuator position
cuff pressure
controller sample time
controller output
controller error

An example of a measurement script file could like like this:

```
**********************************************************************
*                                    *
*       Linear cuff deflation for NIBP measurement          *
*                                    *
*         1997 Draeger Medical Electronics Best          *
*                                    *
*                                    *
*  developped by:                              *
*  Jeroen Heesakkers, graduate Technical University Eindhoven    *
*                                    *
**********************************************************************
*
***********************************
*                     *
*  Measurement definition file   *
*                     *
***********************************
*
**********************
* Initial settings *
**********************
*
measurement name:Meting 500ml;
k value:1.75;
ti value:1.75 seconds;
td value:0.001 seconds;
*
*************************
* Measurement settings *
*************************
*
display trace:2:1:immediate;
display trace:0:1:immediate;
inflate:200 mmHg:immediate;
* set dump actuator:0:status:554;
* display trace:1:0:timed:5;
* inflate:200 mmHg:immediate;
deflate:30 seconds:15 mmHg:status:554;
set dump actuator:0:status:557;
* set deflate actuator:0:status:554;
#
```

Appendix G Hardware adapter schematics

For adaptation of the proportional valve to the Dialog 2000, a hardware adapter is built.



Appendix H Measurement data

The measurement data is collected and stored in a Microsoft Excel file. The file contains the following data:

For the feed forward P controller:

File: Pcontrol.XLS

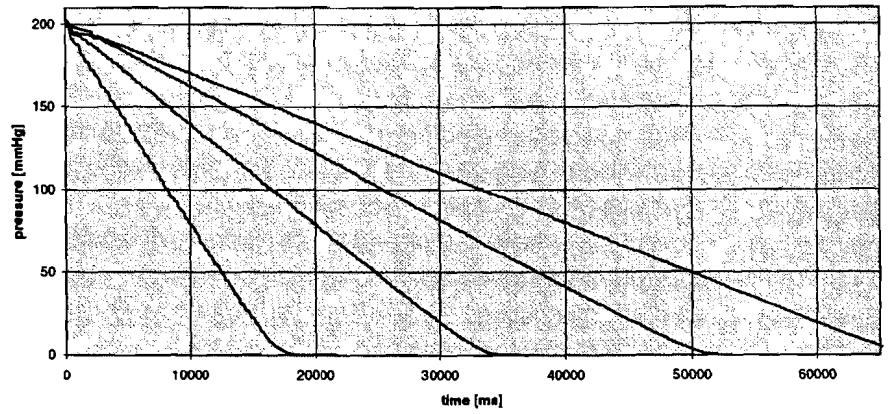
cuff size [mL]	deflate time [s]	template name	comments
500	60	P60s500ml	
60	60	P60s60ml	
30	60	P60s30ml	
	60	COLLECT 60S	collected measurement data for graphic presentation
	60	GRAPH 60S	graphic overview of 60 seconds deflation results
500	45	P45s500ml	
60	45	P45s60ml	
30	45	P45s30ml	
	45	COLLECT 45S	collected measurement data for graphic presentation
	45	GRAPH 45S	graphic overview of 45 seconds deflation results
500	30	P30s500ml	
60	30	P30s60ml	
30	30	P30s30ml	
	30	COLLECT 30S	collected measurement data for graphic presentation
	30	GRAPH 30S	graphic overview of 30 seconds deflation results
500	15	P15s500ml	
60	15	P15s60ml	
30	15	P15s30ml	
	15	COLLECT 15S	collected measurement data for graphic presentation
	15	GRAPH 15S	graphic overview of 15 seconds deflation results
		VERSLAG DATA	collected measurement results of the 500 mL cuff
		GRAPH VERSLAG	graphic overview of the measurement results of the 500 mL cuff

For the feed forward PI controller:

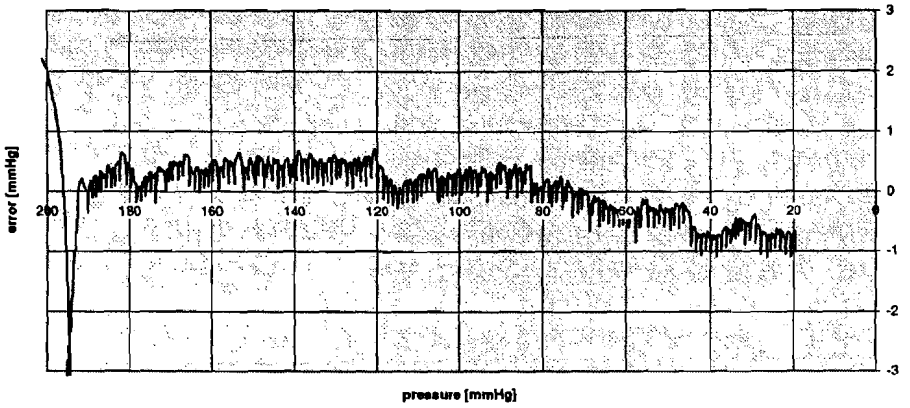
File: Picontrol.XLS

cuff size [mL]	deflate time [s]	template name	comments
500	60	PI60s500ml	
60	60	PI60s60ml	
30	60	PI60s30ml	
	60	COLLECT 60S	collected measurement data for graphic presentation
	60	GRAPH 60S	graphic overview of 60 seconds deflation results
500	45	PI45s500ml	
60	45	PI45s60ml	
30	45	PI45s30ml	
	45	COLLECT 45S	collected measurement data for graphic presentation
	45	GRAPH 45S	graphic overview of 45 seconds deflation results
500	30	PI30s500ml	
60	30	PI30s60ml	
30	30	PI30s30ml	
	30	COLLECT 30S	collected measurement data for graphic presentation
	30	GRAPH 30S	graphic overview of 30 seconds deflation results
500	15	PI15s500ml	
60	15	PI15s60ml	
30	15	PI15s30ml	
	15	COLLECT 15S	collected measurement data for graphic presentation
	15	GRAPH 15S	graphic overview of 15 seconds deflation results
		VERSLAG DATA	collected measurement results of the 500 mL cuff
		GRAPH VERSLAG	graphic overview of the measurement results of the 500 mL cuff

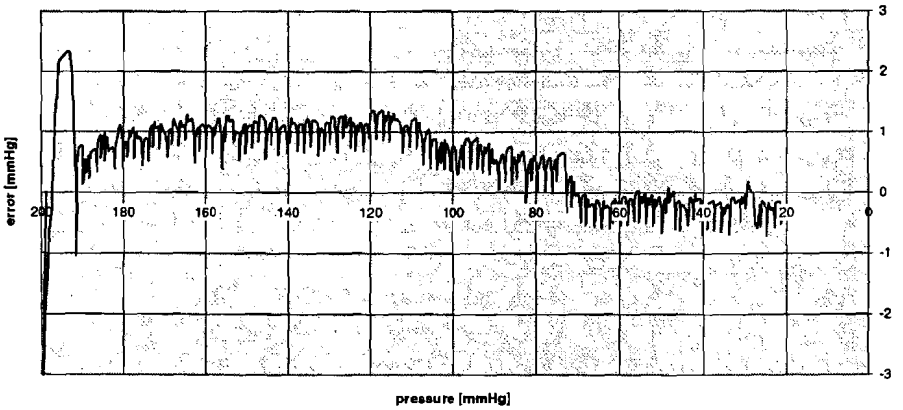
The measurement data of the feed forward P and feed forward PI controller has been included on the next pages.



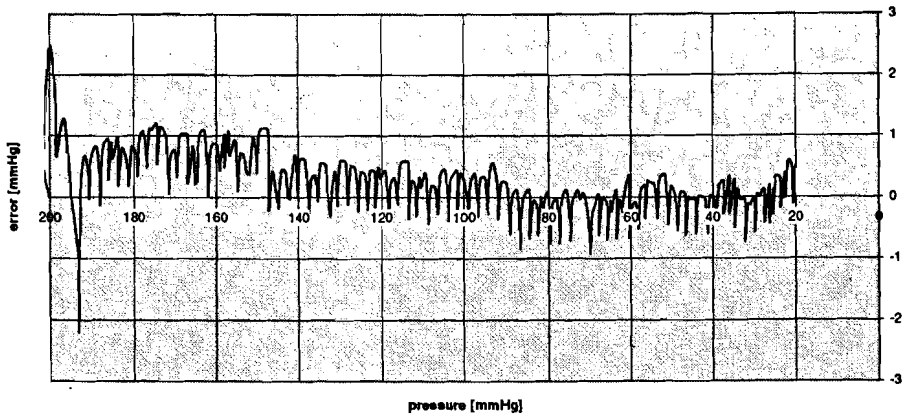
Cuff deflation during 15, 30, 45 and 60 seconds of a 500 mL cuff, using feed forward P control.



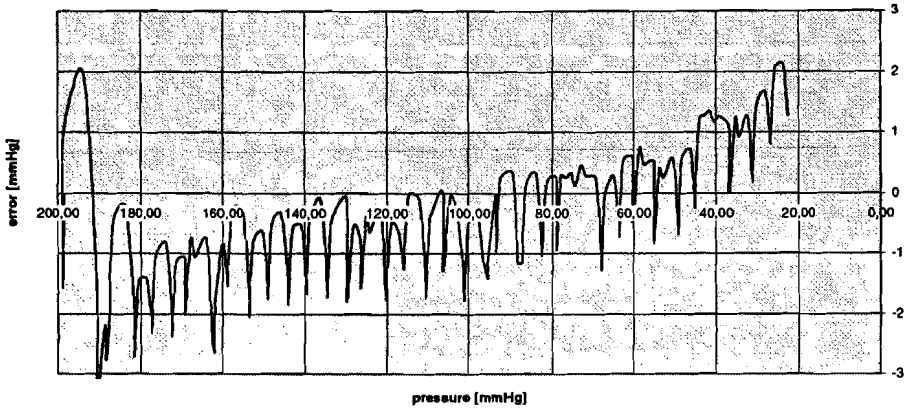
Controller error for 60 seconds deflation of a 500 ml cuff, using feed forward P control.



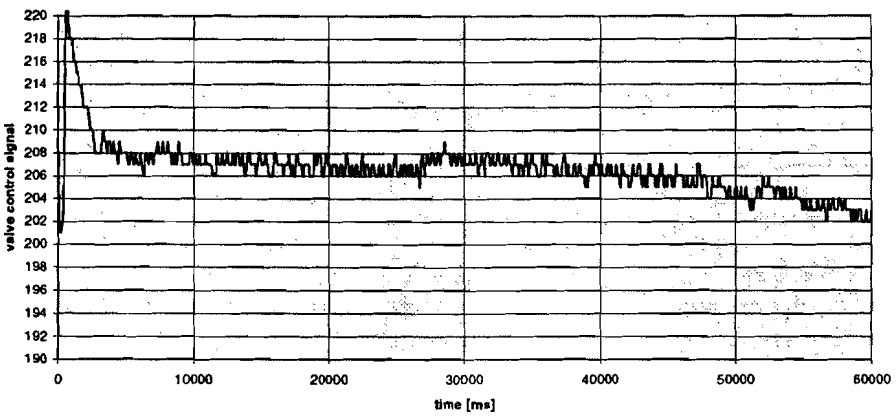
Controller error for 45 seconds deflation of a 500 ml cuff, using feed forward P control.



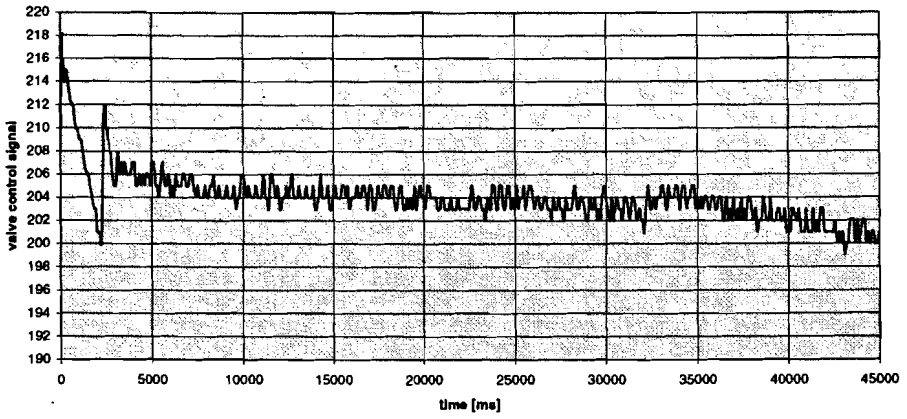
Controller error for 30 seconds deflation of a 500 ml cuff, using feed forward P control.



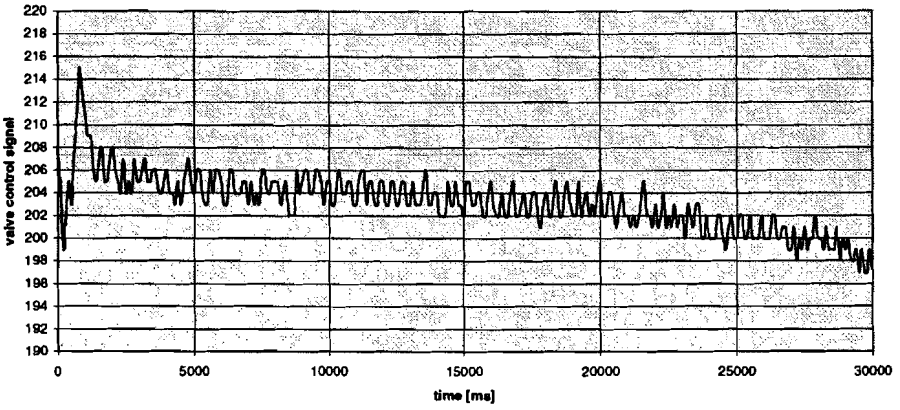
Controller error for 15 seconds deflation of a 500 ml cuff, using feed forward P control.



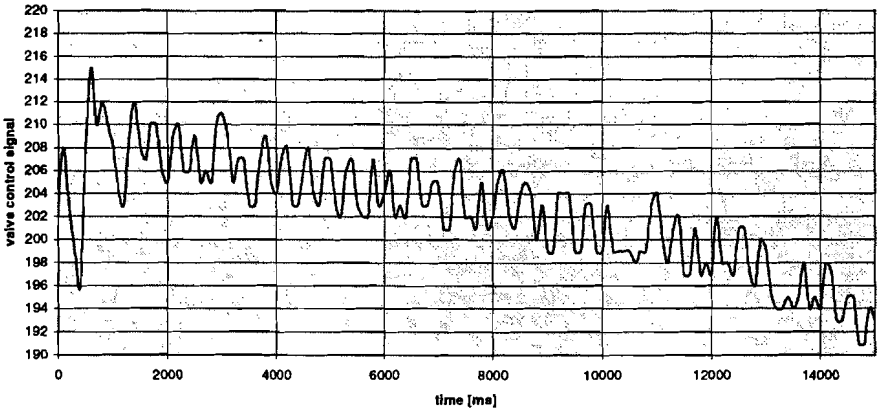
Valve control signal for 60 seconds deflation of a 500 mL cuff, using feed forward P control.



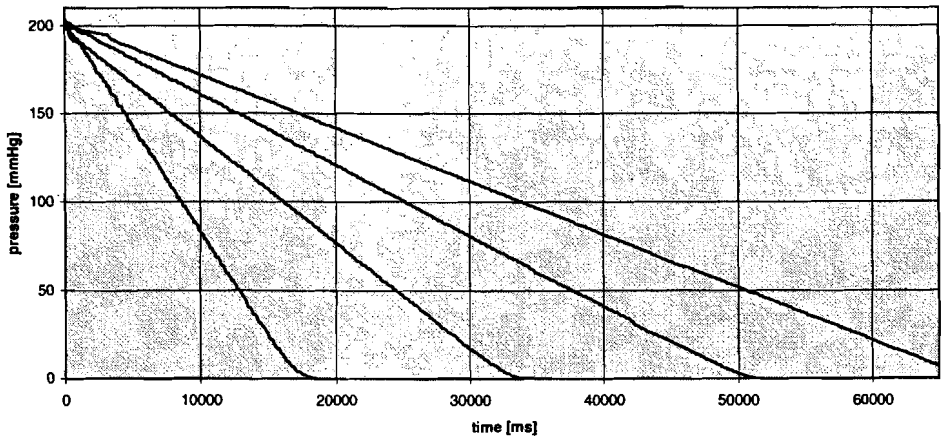
Valve control signal for 45 seconds deflation of a 500 mL cuff, using feed forward P control.



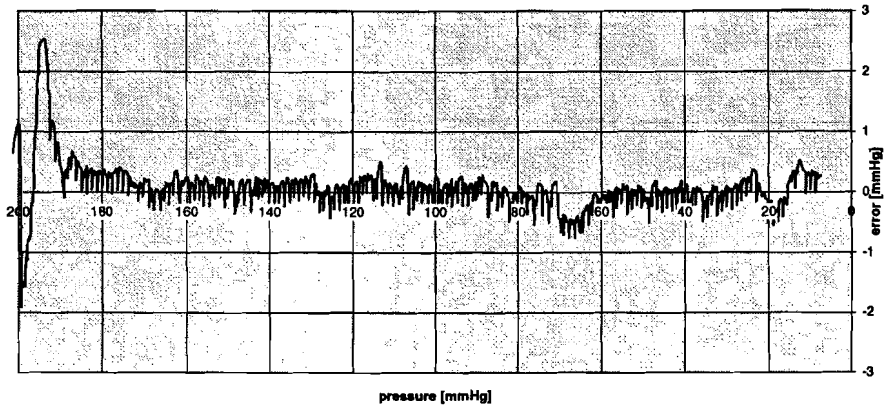
Valve control signal for 30 seconds deflation of a 500 mL cuff, using feed forward P control.



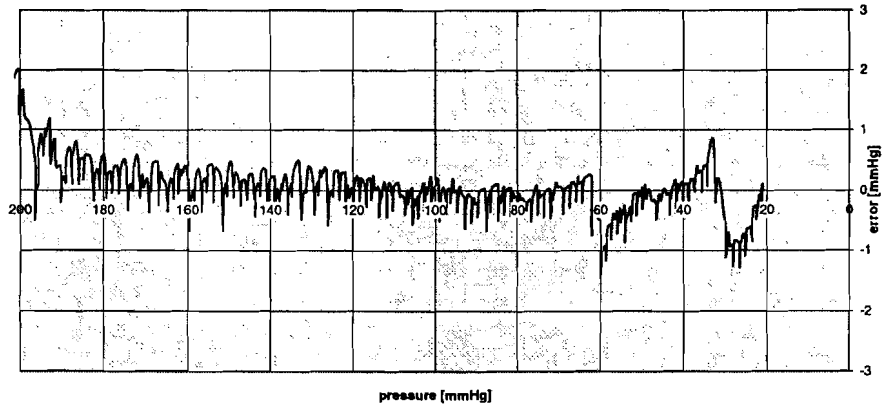
Valve control signal for 15 seconds deflation of a 500 mL cuff, using feed forward P control.



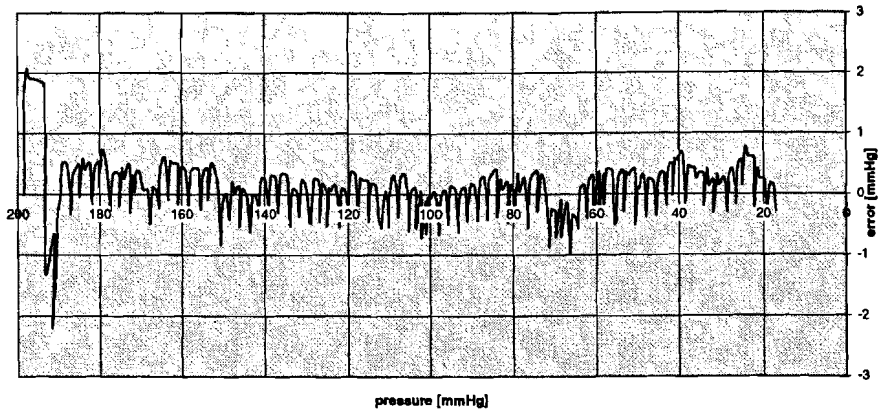
Cuff deflation during 15, 30, 45 and 60 seconds of a 500 mL cuff, using feed forward PI control.



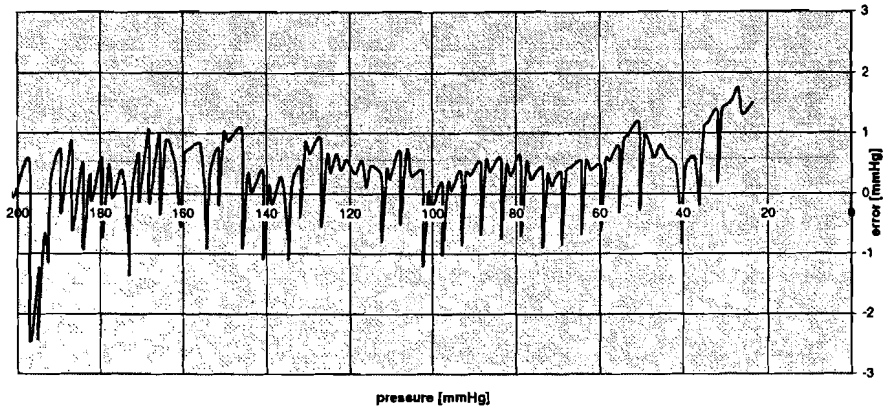
Controller error for 60 seconds deflation of a 500 ml cuff, using feed forward PI control.



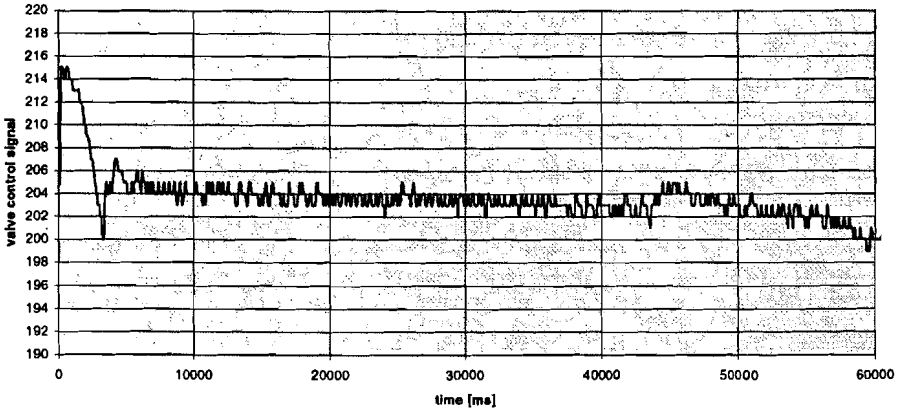
Controller error for 45 seconds deflation of a 500 ml cuff, using feed forward PI control.



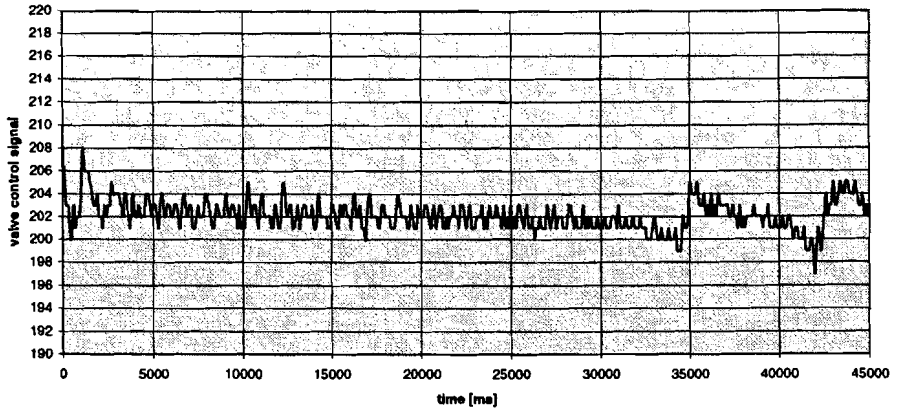
Controller error for 30 seconds deflation of a 500 ml cuff, using feed forward PI control.



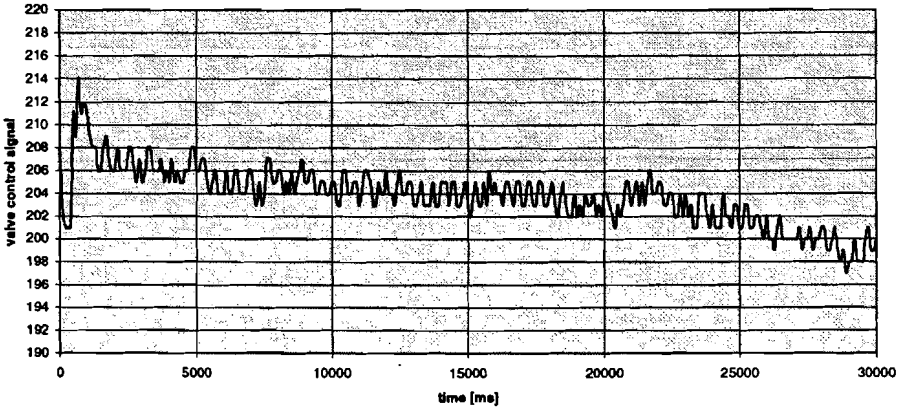
Controller error for 30 seconds deflation of a 500 ml cuff, using feed forward PI control.



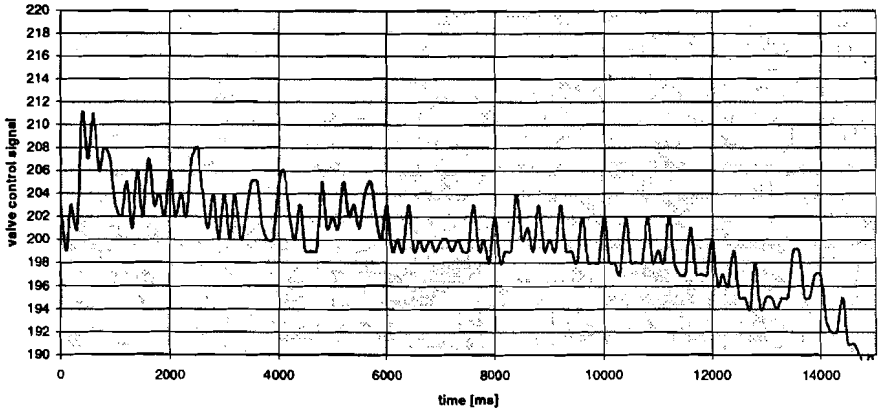
Valve control signal for 60 seconds deflation of a 500 mL cuff, using feed forward PI control.



Valve control signal for 45 seconds deflation of a 500 mL cuff, using feed forward PI control.



Valve control signal for 30 seconds deflation of a 500 mL cuff, using feed forward PI control.



Valve control signal for 15 seconds deflation of a 500 mL cuff, using feed forward PI control.