STUK-YTO-TR 92
OCTOBER 1995

# Experimental testing of an ABB Master application

**P. Haapanen, M. Maskuniitty**
VTT Automation
**J. Korhonen, E. Tuulari**
VTT Electronics

In the Finnish Centre for Radiation and Nuclear Safety
the study was supervised by
**Harri Heimbürger**

This study was conducted at request of
the Finnish Centre for Radiation and Nuclear Safety

FINNISH CENTRE FOR RADIATION AND NUCLEAR SAFETY
P.O.BOX 14, FIN-00881 HELSINKI, FINLAND
Tel. +358-0-759881
Fax +358-0-75988382

ISBN 951-712-071-0
ISSN 0785-9325

Painatuskeskus Oy
Helsinki 1995

*HAAPANEN, Pentti, KORHONEN, Jukka (VTT Automation), MASKUNIITTY, Matti, TUULARI, Esa (VTT Electronics). Experimental testing of an ABB Master application. STUK-YTO-TR 92. Helsinki 1995. 30 pp. + Appendices 24 pp.*

**ISBN** 951-712-071-0
**ISSN** 0785-9325

**Keywords**: Safety, safety analysis, reliability analysis, automation, programmable systems, reactor protection systems, nuclear reactor safety, testing

# ABSTRACT

A prototype dynamic testing harness for programmable automation systems has been specified and implemented at the Technical Research Centre of Finland (VTT). In order to get experience on the methodology and equipment for the testing of systems important to the safety of nuclear power plants, where the safety and reliability requirements often are very high, two different pilot systems have been tested. One system was an ABB Master application, which was loaned for testing from ABB Atom by Teollisuuden Voima Oy (TVO). Another system, loaned from Siemens AG (SAG) by IVO International Oy (IVO), was an application realized with SAG's digital SILT technology. This report describes the experiences gained in testing an APRM pilot system realized with ABB Master technology.

The testing of the pilot application took place in the VTT Automation laboratory in Otaniemi in September—October 1994. The purpose of the testing was not to assess the quality of the pilot system, but to get experience in the testing methodology and find out the further development needs and potentials of the test methodology and equipment.

The experience show that dynamic testing is one feasible way to get more confidence about the safety and reliability of a programmable system that would be hard to achieve by other means. It also shows that more development of the test harness is still needed, especially concerning the comparison of the obtained test response to the expected response provided by the logical model of the system and the user interface of the on-line part of the test harness. Methods for generation of the test cases also need further development eg. for achieving statistical significance for the reliability estimates.

*HAAPANEN, Pentti, KORHONEN, Jukka (VTT Automaatio), MASKUNIITTY, Matti, TUULARI, Esa (VTT Elektroniikka). ABB Master-järjestelmän testaus. STUK-YTO-TR 92. Helsinki 1995. 30 s. + liitteet 24 s.*

**ISBN** 951-712-071-0
**ISSN** 0785-9325

**Avainsanat:** Turvallisuus, automaatio, ohjelmoitavat digitaalijärjestelmät, suojausjärjestelmät, reaktoriturvallisuus, testaus, turvaalisuusanalyysit, luotettavuusanalyysit

# TIIVISTELMÄ

Ohjelmoitavien automaatiojärjestelmien dynaamiseen testaukseen tarkoitettu testiympäristö on määritelty ja toteutettu Valtion teknillisessä tutkimuskeskuksessa (VTT). VTT on testannut kahden järjestelmätoimittajan, ABB Atomin ja Siemens AG:n (SAG), koejärjestelmiä tässä ympäristössä. Koejärjestelmät VTT:n käyttöön ovat toimittajilta lainanneet Teollisuuden Voima Oy (TVO) ja IVO International Oy (IVO). Testausten tavoitteena on ollut kerätä kokemuksia testausmenetelmän ja -järjestelmän soveltuvuudesta ydinvoimalaitosten turvallisuudelle tärkeiden järjestelmien (joiden turvallisuus- ja luotettavuusvaatimukset usein ovat hyvin tiukat) arviointiin. Tämä raportti kuvaa ABB Master-järjestelmällä toteutetun APRM-järjestelmän testausta ja tuloksia.

Testaus suoritettiin VTT Automaation laboratoriossa Otaniemessä syys-lokakuussa 1994. Testauksen tavoitteena ei ole ollut arvioida koelaitteistojen laatua, vaan kerätä kokemuksia testimenettelystä ja löytää menettelyn ja testilaitteiston kehitystarpeita ja -mahdollisuuksia.

Saadut kokemukset osoittavat, että dynaaminen testaus on eräs varteenotettava tapa lisätä uskottavuutta kohdejärjestelmän luotettavuuteen ja turvallisuuteen, mitä muilla keinoilla on vaikeaa saavuttaa. Ne myös osoittavat, että lisäkehitystä edelleen tarvitaan, erityisesti koskien mekanismeja, joilla kohdejärjestelmän testitulosta verrataan sen loogisen mallin antamaan odotettuun vasteeseen. Myös testipenkin on-line osan käyttöliittymää tulisi kehittää käyttäjäystävällisemmäksi. Testitapausten generointi vaatii myös edelleenkehittelyä mm. testien perusteella laadittavan luotettavuusarvion tilastollisen merkitsevyyden saavuttamista varten.

# CONTENTS

# TERMS AND ABBREVIATIONS

| | |
|---|---|
| ABB Master™ | Programmable automation system by ABB Automation |
| ADC | A/D Converter |
| A/D | Analog/Digital |
| APRM | Average Power Range Monitoring system |
| APROS | Advanced Process Simulation System (IVO/VTT) |
| DAC | D/A Converter |
| D/A | Digital/Analog |
| Dynamic testing | Testing of a system by execution of its functioning |
| EXCEL™ | Spreadsheet program by Microsoft® Corporation |
| Expected response | Correct response of the system to a specific test case |
| I/O | Input/Output |
| IVO | IVO International Oy |
| LPRM | Local Power Range Monitoring system |
| RT/SA | Real Time/Structured Analysis |
| RT-SA/SD | Real Time-Structured Analysis/Structured Design |
| Test Harness | (Test environment, test bed, test bench) System or device used for running and automation of tests. |
| Test Oracle | Logical model of the test object used for the calculation of the expected (”correct”) response |
| TVO | Teollisuuden Voima Oy |

# 1 INTRODUCTION

The safety assessment of programmable automation systems can not totally be based on conventional probabilistic methods because of the difficulties in quantification of the reliability of the software as well as the hardware. Additional means shall therefore be used to gain more confidence on the system dependability.

One central confidence building measure is the independent dynamic testing of the completed system. The testing is aimed at demonstrating that the delivered system performs to its specification and meets customer requirements, that there are no functional errors in the software or the hardware and that the system interacts effectively (Abbot 1992). The operation of the system is addressed in realistic situations, with realistic operating conditions, with respect to the required reliability. Testing is intended to demonstrate that in a realistic situation, with real inputs, the system will behave as required over a prolonged period of time. Although the testing can not prove the system to be safe, each successful test case can increase the confidence about safety.

The ultimate goal of dynamic testing would be to reveal all possible faults and errors. If the knowledge about the system internal structure together with some continuity, majority etc. principle does not allow the extension of one single test to cover a wider range of test cases, a “complete” testing is required. This requires all possible input and internal state combinations to be covered. This is in practice not possible, since even in systems with a limited number of inputs and internal states the combination explosion would raise the required number of test cases far beyond any practical limits.

Another important goal is to define a statistically significant set of test cases for the estimation of the system reliability. When the requirements are very high, as is the case eg. for the reactor protection system, even this significance usually is hard if not impossible to fulfil.

In many cases only a limited time period is available for the testing before the system start-up, and this time together with the performance of the testing system set the upper limit for the number of test cases. Thus the practical goal would be to define as many different test cases as can be run during the limited time period available for testing.

In any case a large amount of test cases should be executed in order to get any confidence on the system safety through testing. An automated **test harness** is needed to run the required amount of test cases in a restricted time span. A prototype dynamic test harness was specified and implemented at VTT (Haapanen & Korhonen 1994). This system was used for experimental testing of two representative pilot systems developed by ABB Atom and Siemens AG. The purpose of the testing was not to assess the quality of the pilot system, but to get experience in the testing methodology and find out the further development needs and potentials of the test methodology and equipment. Based on experience gathered the system can later be expanded and completed to a full-scope testing environment and used for testing real safety critical nuclear power plant applications when they eventually arise.

The basic configuration of the test harness is presented in Fig. 1. The central part of the system is the “Test Oracle”, a logical model of the test object used to form the expected, “correct” behaviour of the system output signals for the test signals feeded to the test object. The test data generator is actually an input driver feeding input signal values from a predefined test data file to the test object and the test oracle. The result comparator compares the outputs from the test oracle and the test object. An EXCEL spreadsheet has been used to store the output signal time series from the test object and test oracle are and the comparison is made eg. by drawing charts of the time behaviours. In practice the system is divided into two parts. The on-line part consists of an industrial PC computer with proper I/O devices to feed the input signals to the test object and to read the test object output signals to a data file. The generation of the expected output signals by the test oracle and result comparison are made off-line on separate PC-level computers.

This report describes the testing of a pilot system realized with ABB Master technology. The testing took place in VTT in Otaniemi in September-October 1994.

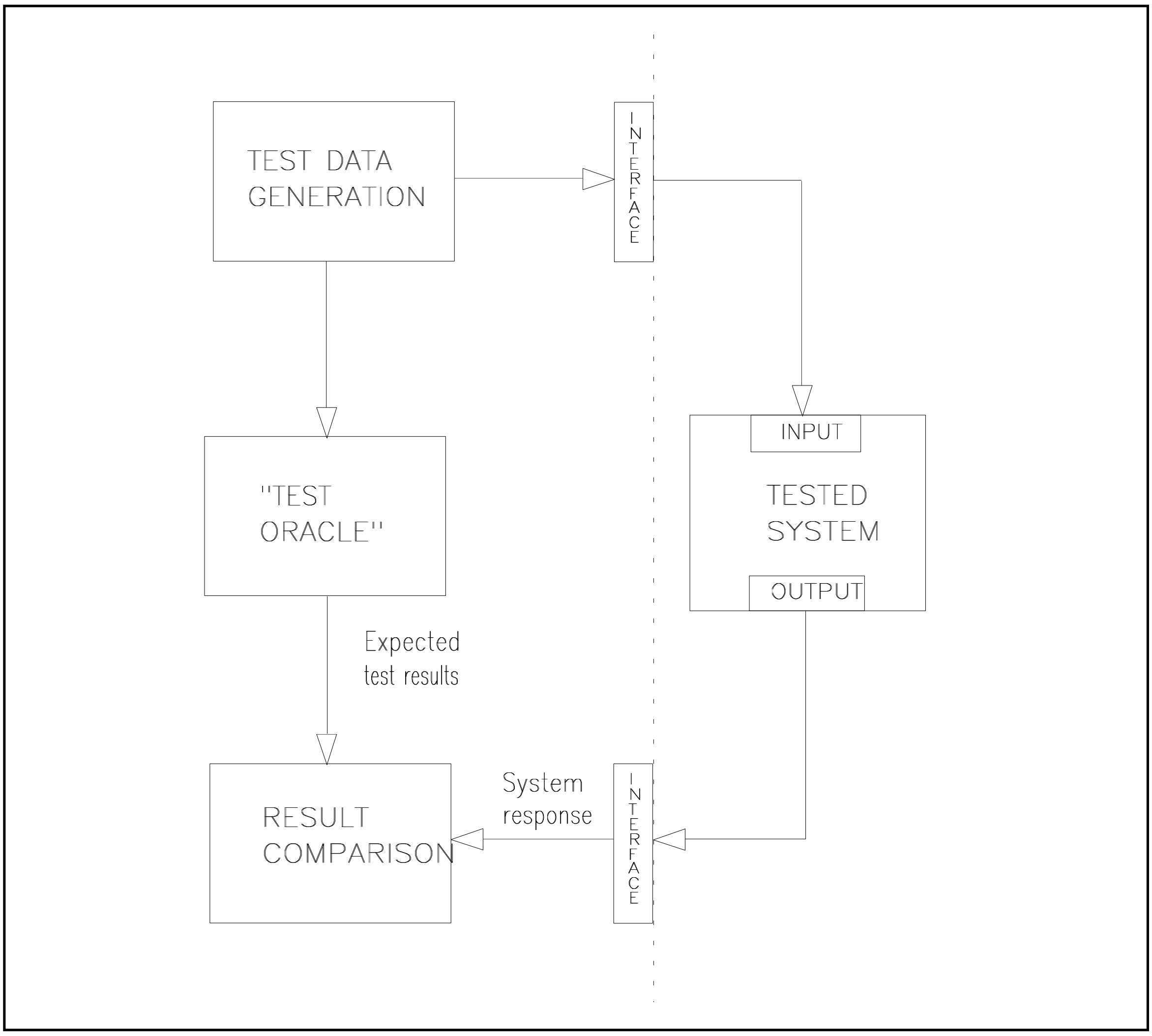


***Figure 1.*** *The principle of the test concept.*

# 2 THE PILOT SYSTEM

*The pilot system was a restricted part of a reactor core power range monitoring system (PRM) configured on an ABB Master automation system. It was based on the specifications of a complete system installed in the Barsebäck plant in Sweden (TVO also is considering replacing the existing electronic APRM system in Olkiluoto plants with similar technology some time in the future). In the following the tested application and the used ABB Master equipment are described.*

## 2.1 The pilot application

The tested pilot application realizes some central parts of an Average Power Range Monitoring (APRM) System for a BWR nuclear power plant. The technical specifications of the Barsebäck APRM system are used as basis for the implementation of the pilot application (ABB 1988, Andersson 1993).

The APRM system surveys the local power distribution and the total thermal power of the reactor core. The reactor core is equipped with local neutron flux sensors (LPRM ~ Local Power Range Monitoring) distributed over the core as shown in Fig. 2 & 3 (an example from the TVO plant). The sensors are located in different radial positions each having 4 sensors at different heights in the core. The APRM signals are used

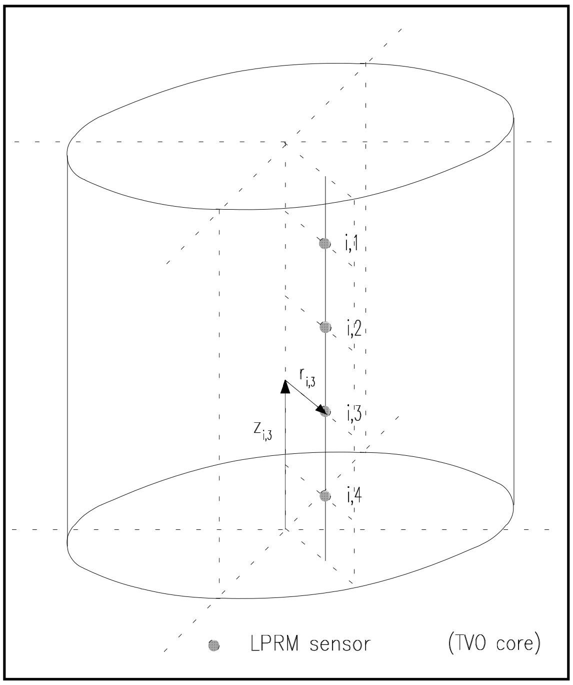


***Figure 2.*** *Location of LPRM sensors in the core.*

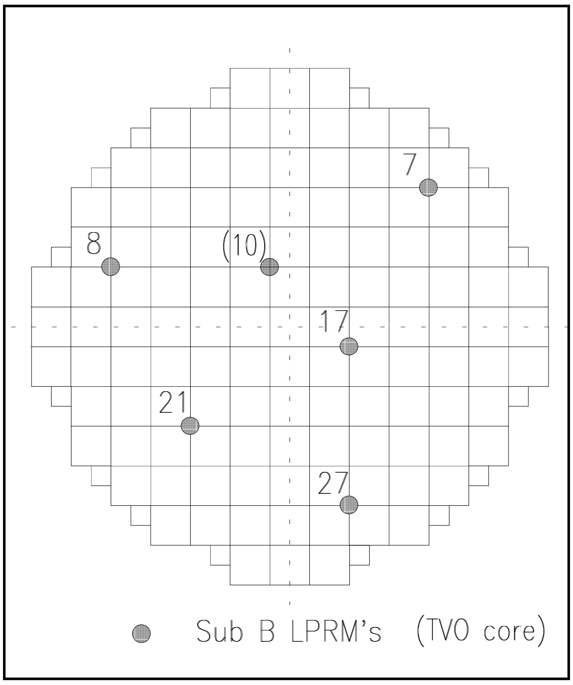


***Figure 3.*** *Radial location of LPRM-sensors of one SUB.*

to produce trip commands for the reactor power set-back and the reactor scram functions[1] if allowed power limits are exceeded. The trip limits depend on the main coolant flow (HC-flow) through the reactor core as indicated in Fig. 4. The system is divided in 4 totally independent subsystems (SUB's) making their own partial trip signals combined then by a voting logic.

The real pilot system implements only one SUB of the APRM system with 20 LPRM measurements and reduced functionality. Only two trip limits, the fast power set-back (E5) limit and fast reactor scram (SS10) limit are realized in the system, and lower filtered limits of the actual system are left out (see Fig. 4). This makes the system static so that the output is all the time

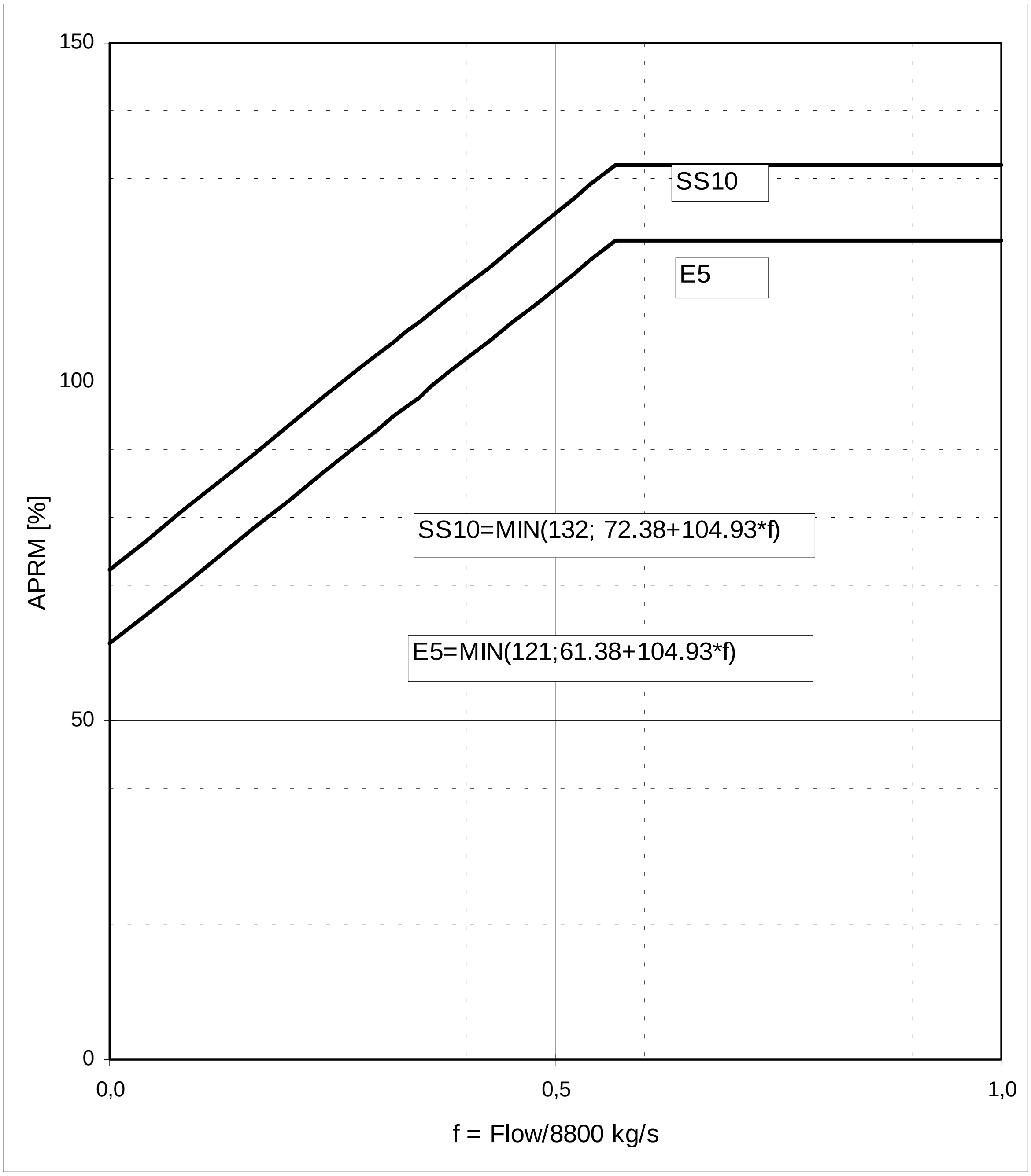


***Figure 4.** Reactor protection system limits SS10 and E5 as functions of the core coolant flow.*

[1] Power set-back = reactor power is lowered by reducing the speed of core coolant circulation pumps.
Reactor scram = reactor is shut down by inserting control rods to the core.

directly determined by the momentary input values without any inherent dynamic behaviour (memory) of the system itself (the filtered limits would have made the system dynamic and thus much more complicated to predict).

The signal levels of the neutron flux sensors are weakening during their lifetime by a factor of ca. 10, and in actual system periodical calibration by increasing the amplification of signals is needed. In the pilot system the amplification coefficients were constants and the influence of their change on system behaviour was not tested. The basic functions included in the pilot system, as well as the logical model, are shown in Fig. 5. The tasks of the system are (the tasks are shown as bubbles in the Fig. 5, the arrows are information flows):

1) to calculate the average power range value (APRM) from 20 local power range values:

$$APRM = \frac{1}{20}\sum_{i=1}^{20} LPRM_i$$

2) to compare the calculated APRM value to HC-flow dependent trip limits (E5 & SS10) and to give the trip signal if either of these is exceeded.

The trip signals are called “reactor-scram”-signal and “power set-back”-signal. The system has thus only 2 outputs. The output signal is ‘1’ if the corresponding trip limit is surpassed and ‘0’ otherwise (De-energized trip condition is used in the acutal system i.e. the signal values are inverted).

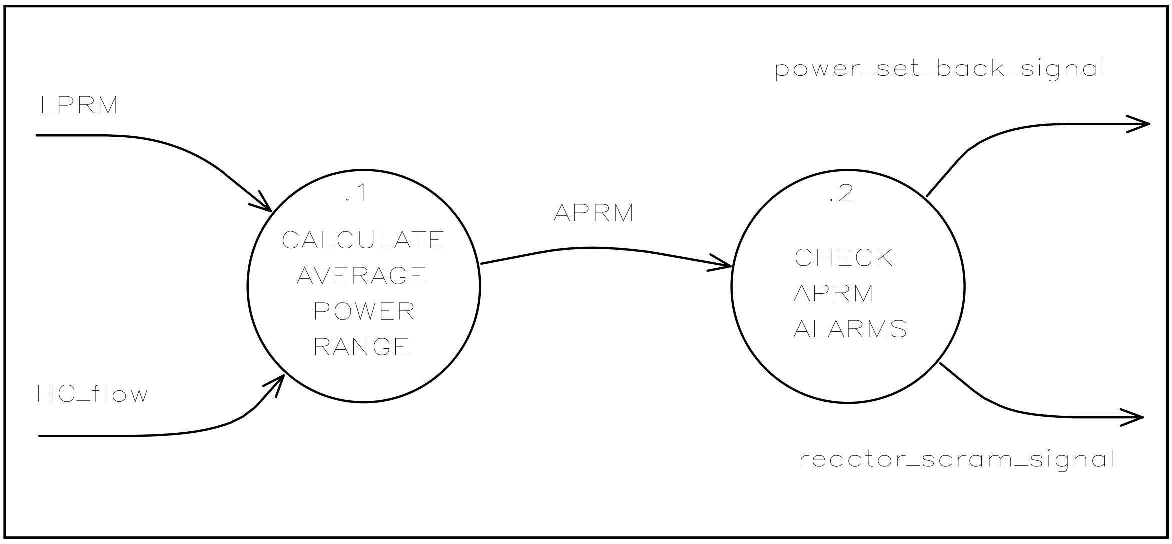


***Figure 5.*** *High-level logical model of the APRM system.*

## 2.2 The pilot equipment

The pilot application was implemented on a small ABB Master system, which consisted of a process control unit ABB MasterPiece 200 (MP200) and a programming aid MasterAid 215.

MP200 is a complete process station unit for logic sequencing, data processing, arithmetic, reporting, logging of process data, weighing, motor speed control, positioning, regulatory control, adaptive control, optimizing, etc. The unit contains all that is needed for the entire control task, e.g. a process database which contains all the information on the process signals and objects connected to it. The hardware composition of the pilot MP200 unit is presented in Fig. 6.

MasterAid 215 is a computer based unit for configuring, application programming, documenting, testing and commissioning of an ABB Master system. The system uses a specialized programming language AMPL (ABB MasterPiece Language) for the development of the application programs. These are developed by selecting and interconnecting function blocks from the system library. A comprehensive set of function blocks from simple logic gates up to loop, motor and valve controllers is available in the Master system.

The program is loaded from a PROM memory (read only) of the MP 200. The state of the incoming analog signals are updated in the data base of the MP 200. The application program could not be modified in this application, but some parameters had to be defined to the read-write memory (RAM) of the unit. These are the amplification factors of the data base modules of the neutron flux and recirculation flow signals. The amplification factors depend on the process for which the system is applied.

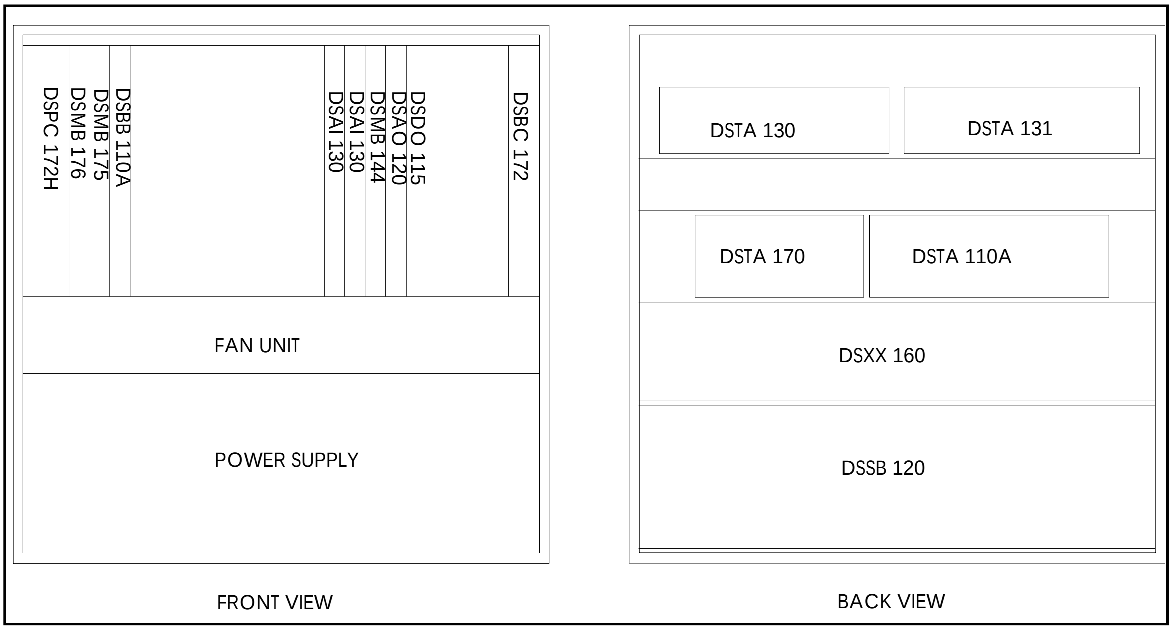


***Figure 6.*** *The components of the pilot MP200 unit.*

# 3 TEST CASES

As the source of the test data in this limited experimental testing a simple reactor core model acquired from TVO was used. This model provides plant data that is not real but realistic enough for testing of the functionality and correctness of the pilot application. The core model is implemented on EXCEL and is presented in Fig. 7.

The model calculates the total thermal power of the reactor core (APRM) as the function of the core coolant flow (HC-flow) and the control rod position using a point kinetic neutron flux model. The model contains no core thermohydraulics so the void feedback was not included. The individual local power range monitoring system (LPRM) measurement signal values are calculated by multiplying the total thermal power by constant radial and axial flux shape factors for each sensor. The radial shape factors RSH is calculated as:

$$RSH_{i,j} = 1—0{,}35*R_{i,j}^{2},$$

where $R_{i,j}$ is the relative distance of the i,j-sensor from the vertical central axis of the core (i indicates the radial and j the axial position of the sensor).

For the axial shape factors ASH the following values are used:

$ASH_{i,1}$ = 0.6, upper
$ASH_{i,2}$ = 0.8, upper middle
$ASH_{i,3}$ = 1.0, lower middle
$ASH_{i,4}$ = 0.8, lower

Individual sensor value is then calculated as:

$$LPRM_{i,j}(t) = APRM(t)*RSH_{i,j}*ASH_{i,j}$$

The model calculates the values of all LPRM signals at constant time intervals; in the actual calculation a 100 ms time step was used. The calculated values are scaled to a 0—250 % relative scale and stored as ASCII values (CSV form) in the test data table as illustrated in Fig. 8. The core coolant flow scaled to 0—8 800 kg/s is also stored to the same table. As the pilot system uses a 20 ms time step, the actual values feeded to the test object are interpolated from the data table.

Two different sets of basic transient cases were calculated using the EXCEL model. In one set the core coolant flow was kept at a constant value

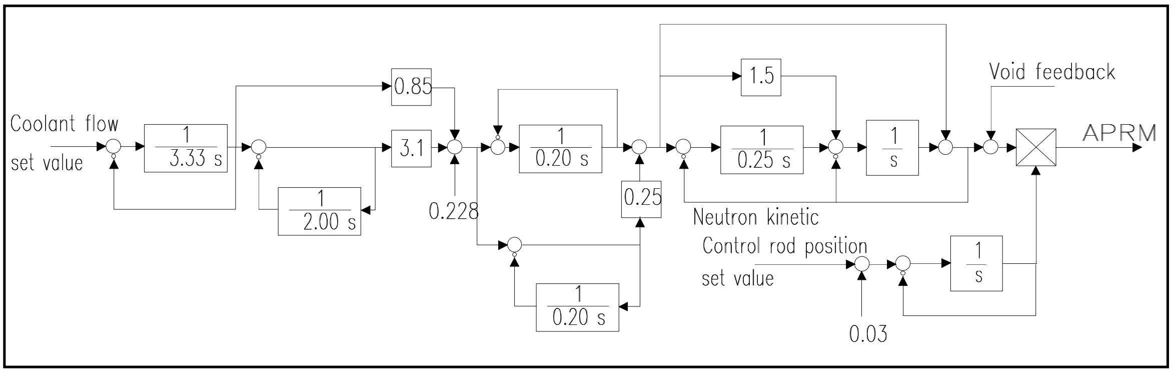


***Figure 7.*** *Point kinetic model of the reactor core (TVO).*

and the control rods were moved so much out of the core, that the average thermal power slightly exceeded the reactor scram limit SS10 (“rod transients”). This set included eight (denoted as F3—F10) cases with different flow values ranging from 31 to 100 % of the nominal value 8 800 kg/s. The other set actually included only one case, where the control rods were kept at constant insertion depth and the coolant flow increased from 57 % to 98 % of nominal value (“flow transient”; FF). The initial control rod position was adjusted so that the thermal power was 100 %. On the flow/flux-diagram this corresponds the normal operation point near the turning point of the flow/flux limit border. The reactor protection system was switched off during the transients so that the actual power limitation and reactor trip limits could be exceeded. Otherwise the protection system — if operating correctly — would have decreased the reactor power immediately after the power surpasses first time the power set-back limit E5, and the operation of the reactor scram at limit SS10 would never been tested.

These nine (9) data sets served as base cases for the testing of the application. More actual test cases were then generated by adding pseudo-random signal noise on the calculated LPRM-signals. The maximum amplitude of the noise was adjusted to be 4 % of the nominal power and the dominating frequency around 0.4 Hz corresponding the actual situation in an operating BWR-plant.

## 3.1 Test cases without noise

The input-files for test-cases without noise have 100 rows. The rows contains the HC-flow value and 20 LPRM detector values. The columns of the input data table contain the momentary input signal values with 100 ms time samples. As the input signals to the model and the pilot system are updated with 20 ms sample intervals, the intermediate values are interpolated from the 100 ms values during the execution of the actual test runs.

Test data values are scaled so that LPRM values between 0 and 1 correspond to 0—250% of nominal power (i.e. when all LPRM values are at 100 % the calculated APRM value corresponds the nominal thermal power of the reactor) in actual LPRM, HC-flow values between 0 and 1 correspond to 0—8800 kg/s in actual HC-flow.

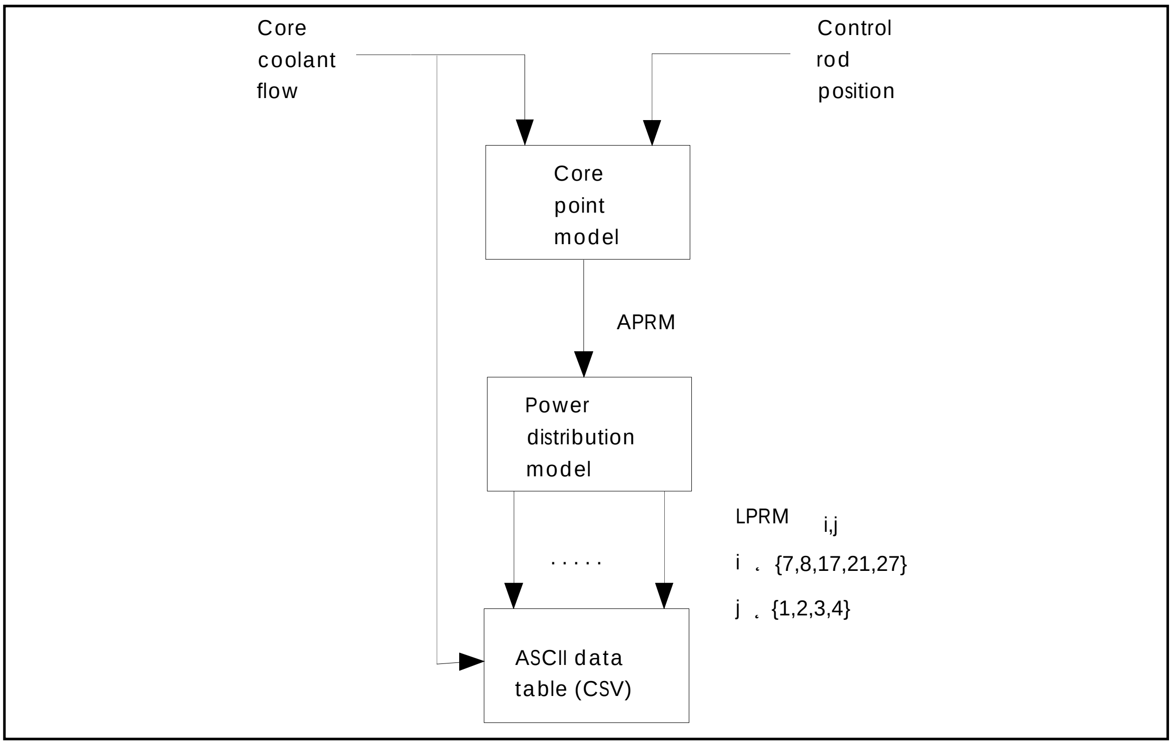


***Figure 8.*** *The principle of making the test data table.*

***Table I.*** *A part of an input data file without noise.*

| HC-flow | LPRM7.1 | LPRM7.2 | LPRM7.3 | LPRM7.4 | LPRM8.1 | LPRM8.2 | .... |
|---|---|---|---|---|---|---|---|
| 0.9 | 0.278644 | 0.371525 | 0.464406 | 0.371525 | 0.282752 | 0.377002 | .... |
| 0.9 | 0.287839 | 0.383785 | 0.479731 | 0.383785 | 0.292082 | 0.389443 | .... |
| 0.9 | 0.296115 | 0.394819 | 0.493524 | 0.394819 | 0.300480 | 0.400640 | .... |
| 0.9 | 0.303563 | 0.404750 | 0.505938 | 0.404750 | 0.308038 | 0.410718 | .... |
| 0.9 | 0.310266 | 0.413688 | 0.517110 | 0.413688 | 0.314840 | 0.419787 | .... |
| 0.9 | 0.316299 | 0.421732 | 0.527165 | 0.421732 | 0.320962 | 0.427950 | .... |
| 0.9 | 0.321729 | 0.428972 | 0.536215 | 0.428972 | 0.326472 | 0.435296 | .... |
| 0.9 | 0.326616 | 0.435487 | 0.544359 | 0.435487 | 0.331431 | 0.441908 | .... |
| 0.9 | 0.331014 | 0.441351 | 0.551689 | 0.441351 | 0.335894 | 0.447858 | .... |
| 0.9 | 0.334972 | 0.446629 | 0.558286 | 0.446629 | 0.339910 | 0.453214 | .... |
| .... | .... | .... | .... | .... | .... | .... | .... |

(100 lines altogether)

***Table II.*** *A part of an input data file with noise.*

Basename of the input file is: f9

Output voltages, cycle time (ms) is : 20

| Cycle | LPRM7.1 | LPRM7.2 | LPRM7.3 | LPRM7.4 | LPRM8.1 | .... | Flow |
|---|---|---|---|---|---|---|---|
| 1 | 2.78630 | 3.71430 | 4.64470 | 3.71430 | 2.82780 | .... | 9.0012 |
| 2 | 2.78140 | 3.83150 | 4.63740 | 3.80460 | 2.94020 | .... | 9.0012 |
| 3 | 2.80340 | 3.91450 | 4.78140 | 3.76310 | 2.76920 | .... | 9.0012 |
| 4 | 2.87420 | 3.78020 | 4.79610 | 3.63370 | 2.64960 | .... | 9.0012 |
| 5 | 2.66670 | 3.98290 | 5.20880 | 3.68500 | 2.75700 | .... | 9.0012 |
| 6 | 2.69110 | 3.97310 | 5.23080 | 3.62880 | 2.69350 | .... | 9.0012 |
| 7 | 2.74970 | 3.87060 | 5.20630 | 3.88280 | 2.82780 | .... | 9.0012 |
| 8 | 2.76920 | 4.10260 | 5.11360 | 3.77050 | 2.65690 | .... | 9.0012 |
| 9 | 2.89380 | 4.08790 | 4.99150 | 3.61420 | 2.98170 | .... | 9.0012 |
| 10 | 3.01100 | 4.16360 | 4.91330 | 3.88770 | 2.72280 | .... | 9.0012 |
| 11 | 3.11110 | 3.99760 | 5.01100 | 3.76310 | 2.74730 | .... | 9.0012 |
| 12 | 3.18930 | 3.93160 | 5.16970 | 3.62150 | 2.87420 | .... | 9.0012 |
| .... | .... | .... | .... | .... | .... | .... | .... |

(5*500 lines altogether)

## 3.2 Test cases with noise

The test data input files with noise have 2 500 rows containing five consecutive test runs generated so that in each single 500 rows long test run different pseudorandom noise values have been added to the same LPRM base data. The base data containing signal values with 100 ms samples is firstly interpolated to contain 20 ms samples and then the noise is added to each value. The rows contain the 20 LPRM values and the HC-flow value scaled directly to voltage values betveen 0—10V in LPRM value corresponding to 0—250% in actual LPRM signal level, 0—10V in HC-flow value corresponds to 0—8800 kg/s in actual HC-flow.

Fig. 9 and 10 give examples of the input signals with and without noise in two different test cases. In Fig. 9 the HC-flow has been kept at a constant value 0.9 and control rods are moved, in Fig. 10 control rods are kept at constant position and HC-flow increased. Only 3 LPRM signals and the

Flow = 0.9, control rod 1 --> 1.35

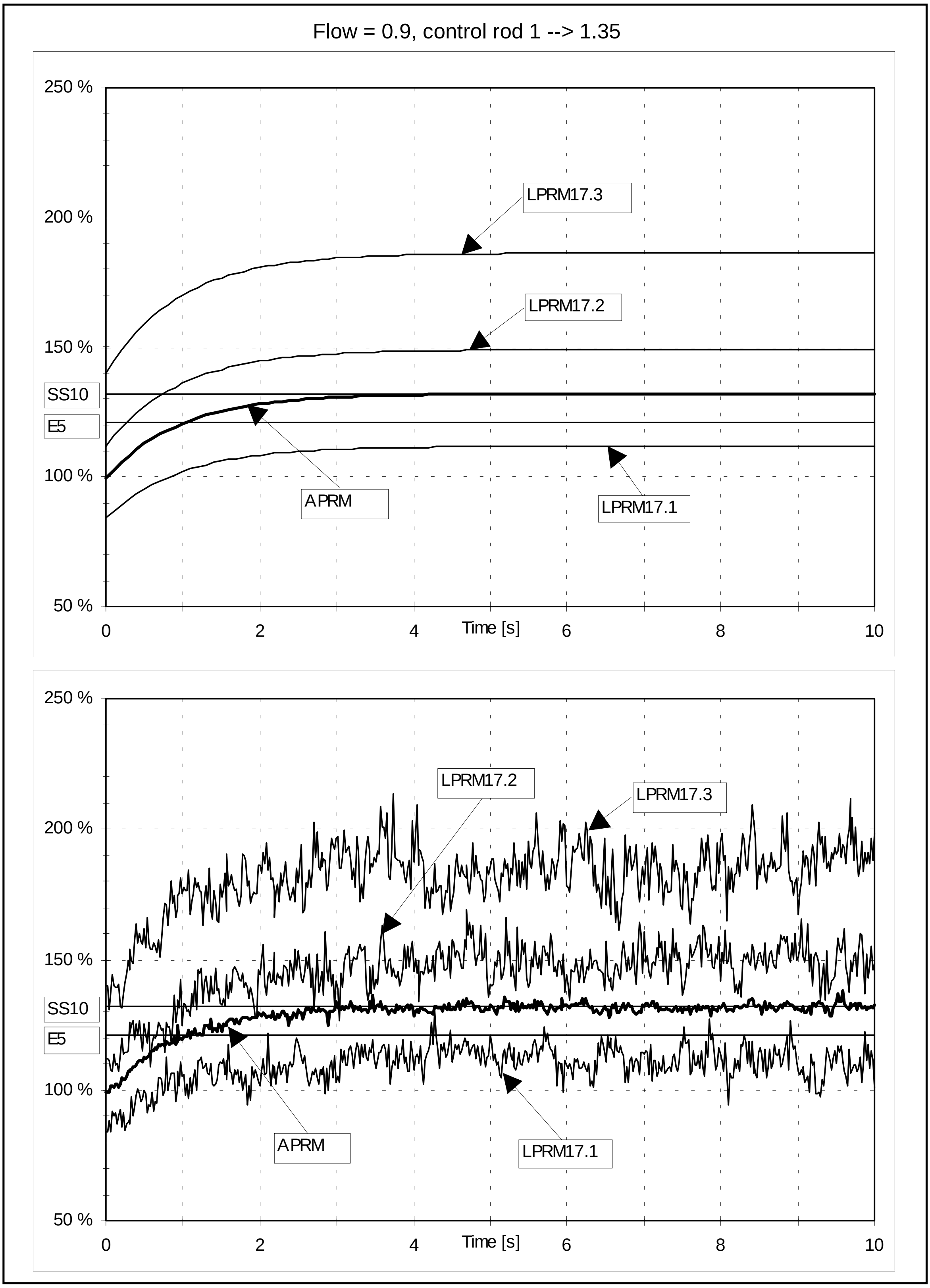


***Figure 9.*** *Time series of three neutron flux sensor signals in a control rod transient.*

APRM signal (corresponding the average of all LPRM values) are presented in order to keep the graphs readable. Upper part of the graphs present the test data without noise, in the lower part the first 500 rows long test run with noise is presented.

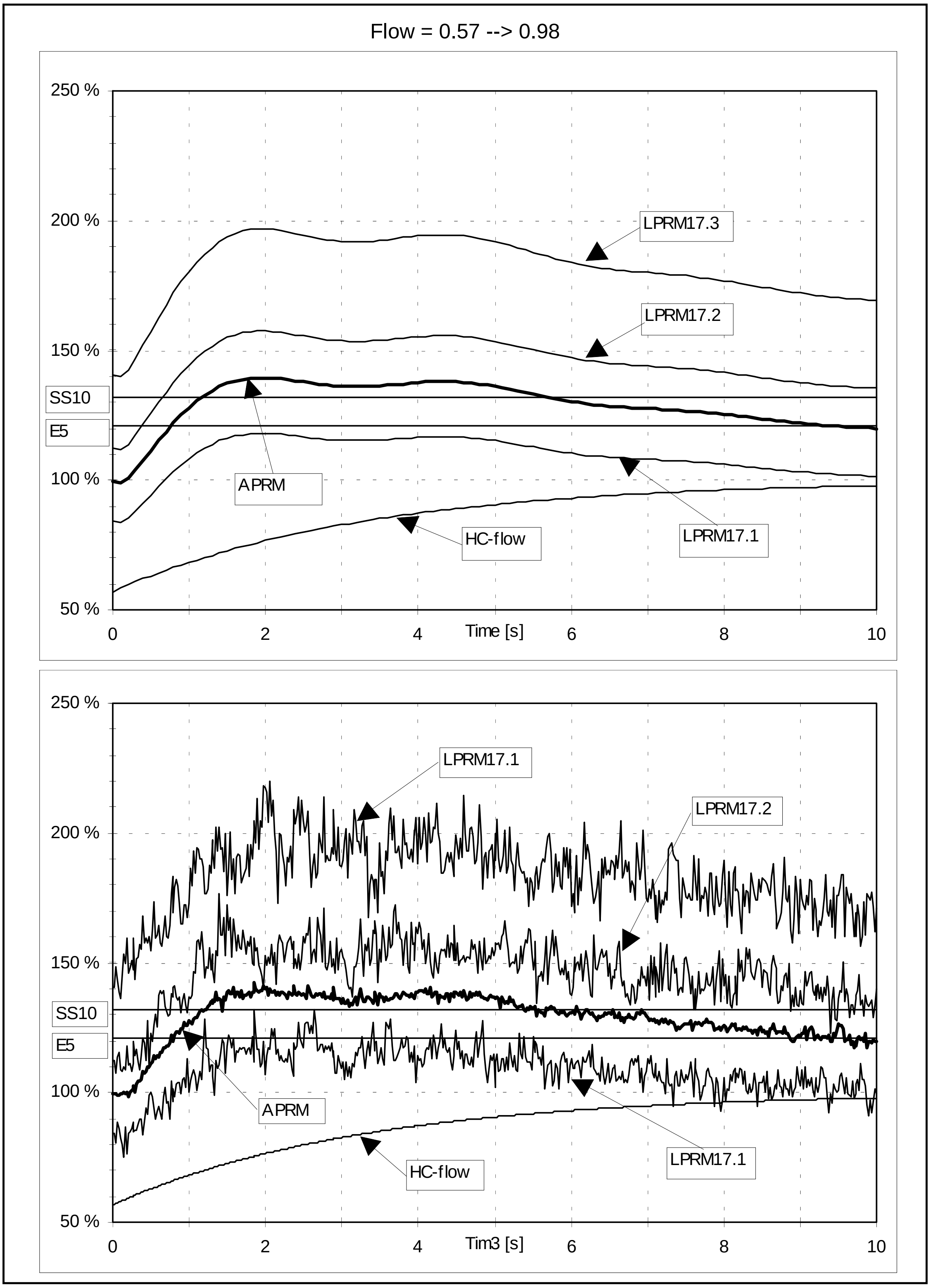


***Figure 10.*** *Time series of three local neutron flux sensor signals in an flow transient.*

# 4 TEST ORACLE

## 4.1 Development of the logical model

The development of the test oracle was based on the requirements specification of the Power Range Monitoring of the ABB-pilot system (ASEA ATOM 1988). The details of the requirements were further discussed with TVO.

The test oracle consist of a logical model of the system together with C-coded input- and output-functions. The logical model was developed by using PROSA structured analysis design tool (Prosa 1989). ReaGeniX code generator was then used for automatically generating C-code from this logical model (ReaGeniX Programmer 1994). The development of the logical model was straightforward because the state-behaviour of the modelled system is very limited.

The first version of the test oracle was quite large as it modelled the whole system requirement specification including for example flow-filtering and alarms for local power range values. However, the actual version of the APRM system turned out to be a simplified version of the APRM system containing only the non-filtered trip limits SS10 and E5 (see Fig. 4). Therefore the test oracle was adapted to correspond the reduced system. The data flow and state transition diagrams of the logical model are presented in App. A.

## 4.2 Validation of the logical model

The logical model was in the early phase of the project validated by comparing results obtained from it to results obtained by manually executing the specification. Errors were immediately corrected to the logical model. Later the validation method was augmented with automatic comparison of the test results. In the comparison the output of the logical model was automatically compared to the “correct” results (see Fig. 11). The test system also reported of any differences between the results. This automated comparison speeded up the testing of the logical model significantly.

Finally the logical model was compared to the model of the reactor protection system also included in the TVO’s EXCEL-model. This model was also used for generating the actual test cases for ABB-pilot. Comparisons show that the responses of the logical model correspond precisely to responses of the EXCEL-model.

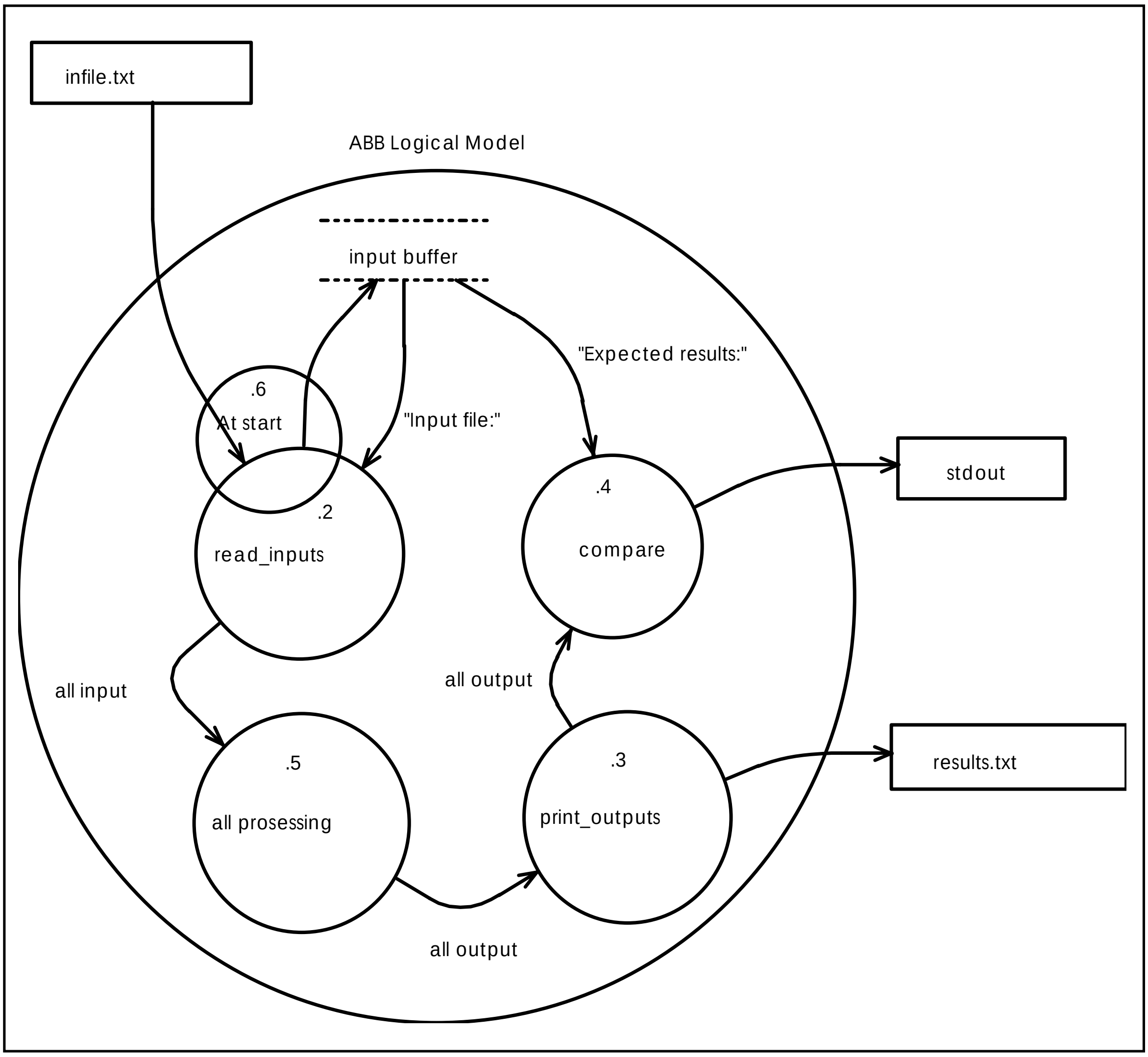


***Figure 11.*** *Validation testing of the logical model.*

# 5 EXPECTED RESPONSE GENERATION

The generation of the expected response for the test cases was done by executing the test cases in the logical model in a similar manner as the validation of the model was done. The actual logical model of the pilot system was enhanced with some new functions that were used for reading the inputs from an input file (infile.txt in Fig. 11 ) and storing the response of the model into an output file (results.txt in Fig. 11).

The test input data files were read as text files (see Tab. I and II) to the PC-computer where the logical model was executed. The output values of the logical model were stored into the output file each time when the calculated APRM value crosses one or the other of the trip limits i.e. the status of either of the trip signals change. The output file contains the number of the cycle and the exact event time of the change and the states of the trip signals at that moment of time. The file also gives the exact value of the APRM signal at the event time.

An example of the logical model output file in a rod transient without noise is given in Tab. III.

As can be seen from Tab. III, only two events are recorded:

- Power set-back limit E5 has been exceeded on cycle nr. 70 at 1.4 seconds after the start of the transient; APRM value was 121.018 %
- Reactor scram limit SS10 has been exceeded on cycle nr. 219 at 4.38 seconds after the start of the transient; APRM value was 132.005 %

The output file for test cases including noise is in principle similar, but if the power is increased slowly (as was case in the specified test cases), much more events are recorded as the APRM value exceeds and the again falls below the trip limits due to the noise. An example in this case is given in Tab. IV. In this special case the file contains 83 events; the number of events in all recorded cases varied from 34 to 134. The responses of the logical model are presented in graphical form in App. B together with the actual responses of the pilot system.

As there were no timers in the logical model, the execution of test cases was totally independent of real-time. This means that a new input was fed into the model as soon as the response from the previous input was received. Thus the execution rate was tens of times faster than in the pilot system.

***Table III.*** *Response of the logical model to a rod transient without noise, HC-flow = 0.7.*

```
Output # 70: time: 1400 ms

          APRM: 121.018163

          APRM alarms, h1 (E5):       1

          APRM alarms, h2 (SS10):     0

Output # 219: time: 4380 ms

          APRM: 132.005300

          APRM alarms, h1 (E5):       1

          APRM alarms, h2 (SS10):     1
```

***Table IV.*** *Response of the logical model to rod transient with noise, HC-flow = 0.7.*

```
Output # ; 65; time ;1300; ms APRM ;122.253000; ss10:  ;0; E5:    ;1;

Output # ; 68; time ;1360; ms APRM ;117.863500; ss10:  ;0; E5:    ;0;

Output # ; 69; time ;1380; ms APRM ;121.984000; ss10:  ;0; E5:    ;1;

Output # ; 72; time ;1440; ms APRM ;119.349875; ss10:  ;0; E5:    ;0;

Output # ; 73; time ;1460; ms APRM ;121.978250; ss10:  ;0; E5:    ;1;

Output # ;162; time ;3240; ms APRM ;132.008750; ss10:  ;1; E5:    ;1;

Output # ;170; time ;3400; ms APRM ;131.068375; ss10:  ;0; E5:    ;1;

Output # ;177; time ;3540; ms APRM ;132.722750; ss10:  ;1; E5:    ;1;

Output # ;179; time ;3580; ms APRM ;130.286875; ss10:  ;0; E5:    ;1;

Output # ;192; time ;3840; ms APRM ;132.566875; ss10:  ;1; E5:    ;1;

........

(83 lines altogether)
```

# 6 TESTING ARRANGEMENT

The connection of the test harness to the test object is presented in Fig. 12. The 21 analog output signals from the test harness (test input) and 2 binary output signals from the pilot system back to the harness were hard wired to the terminal block located in the backside of the MP200. The analog signals were connected through D/A converter channels directly to the analog inputs of the test object as 0 - 10 V signals. The two binary output signals from the pilot system are connected to the test harness binary inputs through opto-isolators in order to prevent electrical interferences between systems and adapt the different voltage levels. To avoid electrical disturbances, the MP200 was connected to an uninterruptible power source (UPS) during the tests.

The analog inputs to the test object consist of twenty LPRM signals and the core coolant flow signal. Test object produces two binary output signals, the reactor scram signal SS10 and the power set-back signal E5.

The original test data tables from the ECXEL model contained the input data with 100 ms sample intervals and the individual LPRM values were listed in numerical order. These tables were arranged off-line to separate files each containing the time series of one input signal as presented in Fig. 13. These files (21 pieces, 20 LPRM files and one HC-flow file) were stored in the on-line test harness computer, which did the scaling to proper values for the D/A conversion, necessary interpolation to the desired sample intervals and possible addition of pseudorandom noise to the APRM signals and finally wrote the data to the input registers of the D/A converters. The on-line part of the test harness operated in real time updating the signal values with 20 ms time intervals.

The on-line system sensed the status of the pilot system output E5 and SS10 continuously with 0.01 ms time resolution and saved the exact time of the event always when one of the signals changed its state.

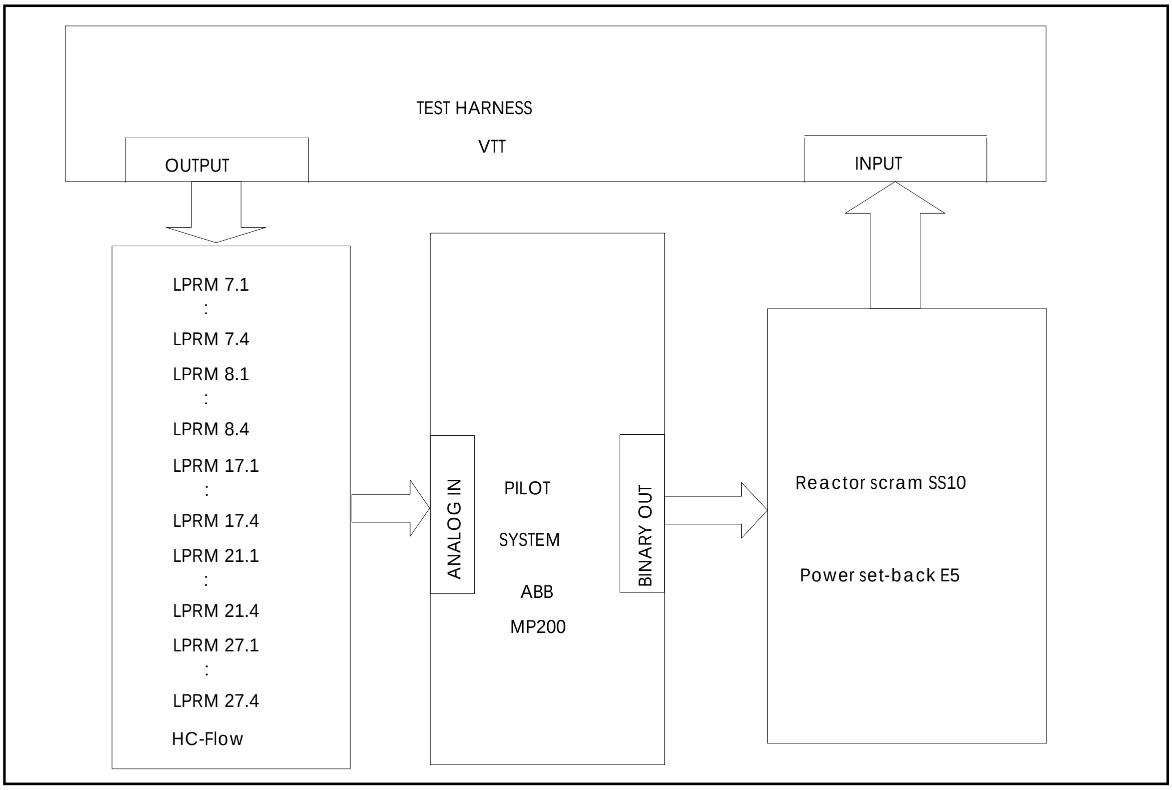


***Figure 12.*** *Configuration of the test arrangement.*

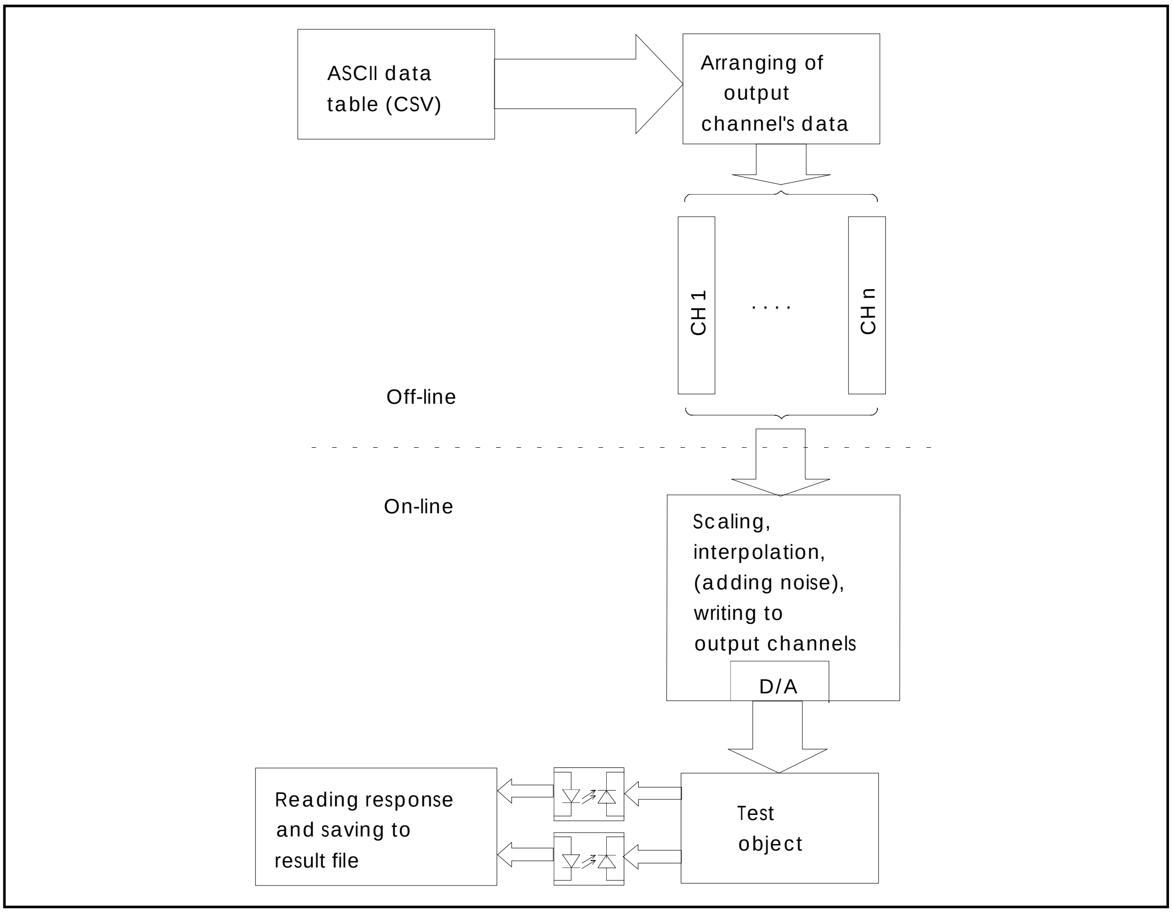

ASCII data table (CSV)
Arranging of output channel's data
CH 1
. . . .
CH n
Off-line
On-line
Scaling, interpolation, (adding noise), writing to output channels
D/A
Test object
Reading response and saving to result file

***Figure 13.*** *Preparing and feeding of the test data to the test object and reading the system response.*

# 7 TEST RESULTS

*The pilot system output was compared to the logical model output ("the correct response") manually by comparing the output file listings. This was considered to be the most suitable way to do the comparison as the number of the test cases is small. The test input data and the responses of the pilot system and the logical model are also presented in graphical form in App. B, which makes the comparison easier especially in the case of test signals containing noise.*

## 7.1 Test cases without noise

An example of the pilot system output file in a test case without noise is given in Tab. V. The table contains the cycle index, the elapsed time from the beginning of the transient and the status of the binary inputs to the test harness recorded each time when one of the inputs has changed its state. The last bit in the digital output byte in Tab. V is the status of the power set-back signal E5 and the second last the status of the reactor scram signal SS10. The six most significant bits are non-relevant in this case.

The corresponding output file of the logical model in the same test case is presented in Tab. VI.

The response of the pilot system resembles the logical model predictions quite well in most test cases, although there seems to be a tendency of the pilot system events (event is a change in the state of one of the trip signals) to become registered somewhat later than the logical model has predicted. The event times predicted by the logical model and actually recorded in the tests and their time differences are presented in Tab. VII.

***Table V.*** *Response of the pilot system to rod transient without noise, HC-flow = 0.9.*

```
Index :  55 elapsed time (ms) : 1093.90  digital input : 11111101
Index : 331 elapsed time (ms) : 6633.20  digital input : 11111111
```

***Table VI.*** *Response of the logical model to rod transient without noise, HC-flow = 0.9.*

```
Output # 52: time: 1040 ms
 APRM: 121.100210
 APRM alarms, h1 (E5):    1
 APRM alarms, h2 (ss10):  0
Output # 293: time: 5860 ms
 APRM: 132.000670
 APRM alarms, h1 (E5):    1
 APRM alarms, h2 (ss10):  1
```

***Table VII.*** *Event times and time differences in transients without noise [ms].*

| | E5 | | | SS10 | | |
|---|---|---|---|---|---|---|
| Transient | Model | Pilot | Difference | Model | Pilot | Difference |
| F3 | 1560 | 1577.43 | 17.43 | 5080 | 5232.64 | 152.64 |
| F4 | 1560 | 1604.29 | 44.29 | 4880 | 4903.81 | 23.81 |
| F5 | 1580 | 1629.47 | 49.47 | 4600 | 4669.07 | 69.07 |
| F6 | 1580 | 1625.27 | 45.27 | 6040 | 6544.62 | 504.62 |
| F7 | 1400 | 1447.42 | 47.42 | 4380 | 4506.96 | 126.96 |
| F8 | 1220 | 1270.70 | 50.70 | 4080 | 4210.32 | 130.32 |
| F9 | 1040 | 1093.90 | 53.90 | 5860 | 6633.20 | 773.20 |
| F10 | 760 | 783.92 | 23.92 | 3580 | 3663.54 | 83.54 |
| FF | 800 | 828.90 | 28.90 | 1200 | 1228.81 | 28.81 |
| FF | 9400 | 9627.62 | 227.62 | 5740 | 5828.07 | 88.07 |

Some of the time differences (1 or even two 20 ms time steps) can be explained by the asynchronous operation of the pilot system and the test harness and the settling times of the D/A and A/D converters in the signal path. If the time moment when the test harness writes a value to the D/A input register takes place after the time moment the pilot system reads the corresponding input channel the potential event can be discovered by the pilot system at the earliest at the next time step.

The larger time differences can be explained by the inaccuracies in the analog signal path from the test harness to the test object. Small differences in the data feeded to the logical model and the test object were caused e.g. by the use of uncalibrated analog output boards in the test harness, the limited resolution of the D/A and A/D conversions (e.g. 12 bit converter in the test harness) etc. These differences could be avoided by feeding the logical model with the actual measured values from the test object using a standard EXCOM-protocol. The smaller is the time gradient of the input signals at the time of

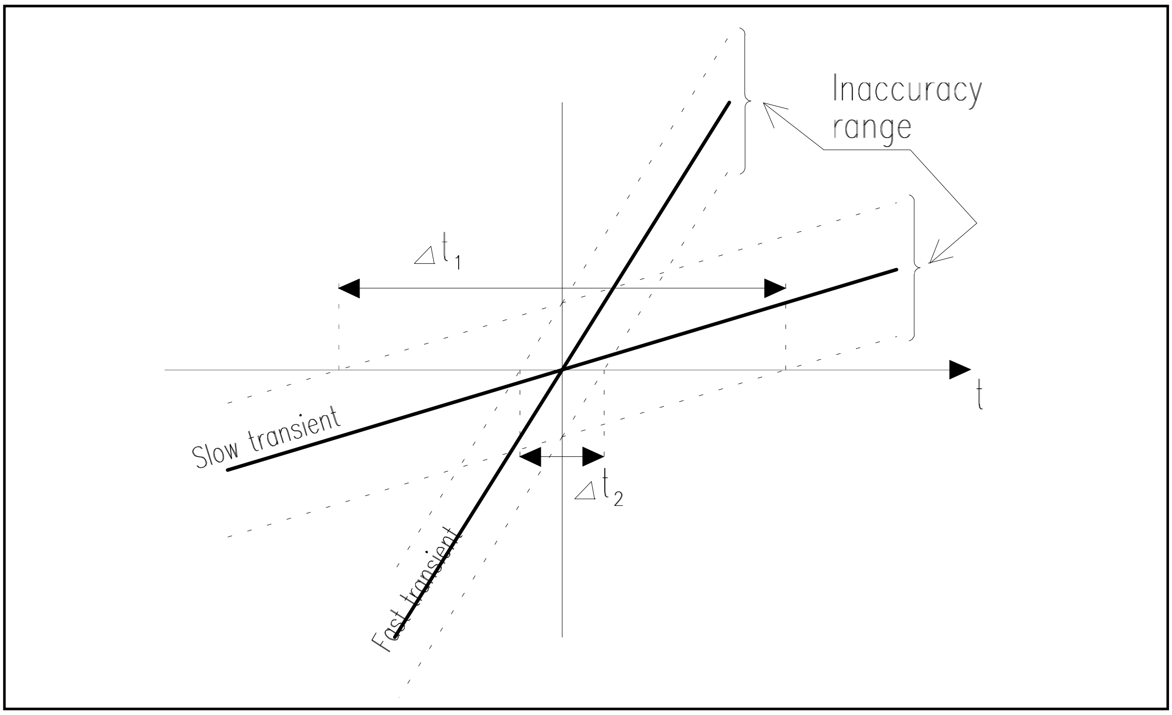


***Figure 14.*** *Timing errors caused by quantification inaccuracies.*

***Table VIII.*** *Response of the pilot system to rod transient with noise, HC-flow = 0.9.*

```
Index :    48 Total time (ms) :   945.08    digital input :11111101
Index :    50 Total time (ms) :   984.88    digital input :11111100
Index :    54 Total time (ms) :  1065.07    digital input :11111101
Index :   151 Total time (ms) :  3004.78    digital input :11111111
Index :   156 Total time (ms) :  3104.62    digital input :11111101
Index :   160 Total time (ms) :  3184.76    digital input :11111111
Index :   164 Total time (ms) :  3264.61    digital input :11111101
Index :   173 Total time (ms) :  3444.74    digital input :11111111
Index :   183 Total time (ms) :  3644.55    digital input :11111101
Index :   199 Total time (ms) :  3964.66    digital input :11111111
......
(59 lines altogether)
```

event, the larger is the possible time difference caused by the same signal difference. In extreme cases the time difference may grow to the infinity, that is, an event in one system (pilot or model) is not observed in the other system at all. This principle is illustrated in Fig. 14. Since the test transients were selected so that the reactor scram limit SS10 was just exceeded, the time gradient of the APRM value is much lower at the SS10 limit than as the power set-back limit E5. Therefore also the time differences are bigger in the SS10 signal.

## 7.2 Test cases with noise

An example of the pilot system output file in a test case with noise is given in Tab. VIII. The table is in principle similar than in test cases without noise, but there are much more events recorded. The noise causes the APRM value exceed the trip limit and return back below the limit again several times. The transients are adjusted just to exceed the SS10 limit and therefore there are more changes of state in SS10 signal than in E5 signal.

The corresponding output file of the logical model in the same test case is presented in Tab. IX.

There were problems in comparing results from the ABB-pilot and from the logical model, because the responses are quite different as can be seen from Tab. VIII and IX and also from the graphs in App. B. The main difference is that the logical model predicts much more events than are actually observed in the pilot system. For instance in the selected example cases, the pilot system produced 59 events when the logical model predicted 134 events. This result — that the number of trip limit crossings was much greater in the logical model than in the ABB pilot system — was noticed in all test cases with noise.

This difference appears to be caused by analog signal filtering at the input of the pilot system. The “technical specification for quotation” used as the basis for the logical model did not specify the filtering of the input signals, and therefore the behaviour of the logical model deviates in this respect from the pilot system behaviour. The design documents produced in the later phase of the design process would have defined these filters, but they were not available for this study.

***Table IX.*** *Response of the logical model to rod transient with noise, HC-flow = 0.9.*

```
Output # ; 47; time   ; 940; ms    APRM ;       124.477750; ss10:  ;0;      E5:     ;1;
Output # ; 48; time   ; 960; ms    APRM ;       119.352875; ss10:  ;0;      E5:     ;0;
Output # ; 54; time   ;1080; ms    APRM ;       122.832625; ss10:  ;0;      E5:     ;1;
Output # ; 55; time   ;1100; ms    APRM ;       119.767875; ss10:  ;0;      E5:     ;0;
Output # ; 56; time   ;1120; ms    APRM ;       122.167375; ss10:  ;0;      E5:     ;1;
Output # ;150; time   ;3000; ms    APRM ;       133.681125; ss10:  ;1;      E5:     ;1;
Output # ;151; time   ;3020; ms    APRM ;       131.694000; ss10:  ;0;      E5:     ;1;
Output # ;154; time   ;3080; ms    APRM ;       132.273875; ss10:  ;1;      E5:     ;1;
Output # ;155; time   ;3100; ms    APRM ;       130.879125; ss10:  ;0;      E5:     ;1;
Output # ;159; time   ;3180; ms    APRM ;       133.684500; ss10:  ;1;      E5:     ;1;
......
(134 lines altogether)
```

# 8 CONCLUSIONS

No major problems were found in developing the logical model for the ABB pilot system. Though ReaGeniX is especially intended for embedded systems, it proved to be flexible enough for modelling automation systems, too. The amount of work that was used for model development and testing, was approximately 2,5 manmonths. Compared to the other pilot experiment, the effort was larger although the model is much simpler, but this can be explained by two reasons. Firstly, the persons involved in the development had to be changed so that creation of the model and testing of the model were done by different persons. Although the change of personnel probably increased the correctness of the model, it also increased the amount of work. Secondly, the model that was once created and tested, had to be updated due to the simpler implementation of the pilot system. It later turned out that the many features of the original specification had been omitted in the pilot system, and therefore the missing features had also to be eliminated from the logical model. However, taking into account the previous reasons, the total amount of work can be considered reasonable.

The pilot application was too simple to give a full comprehension about the potentials of the selected modelling methodology, e.g. a corresponding logical model was also included in the EXCEL model used for the development of the test data. One can estimate that the implementation of this ECXEL model of the actual pilot system functionality can not take many manhours of work effort.

Those rather few test-cases used in this comparison show clearly that the responses of the two systems are not identical. However, this does not directly indicate that there were errors in the pilot system or in the logical model, but a closer look at the discrepancies is needed. For example, all differences in test data (e.g. caused by the use of uncalibrated analog output boards and the quantification errors in the D/A- and A/D-conversions) may in some case cause quite large differences in the time behaviour of the system as is shown in Fig. 14. The smaller is the gradient of the signal when it surpasses the trip limit the greater is the time difference between the expected and actual moment of time when the trip occurs. This means that a more intelligent comparison algorithm is needed for distinguishing actual erroneous behaviour of the system from discrepancies caused by quantification error. By using the actual measured data from the test object as input for the logical model the deviations could be minimized.

The ABB-pilot example has shown that dynamic testing is not an easy approach. This is best seen in the interpretation of the test results. Though the amount of test cases was small, the evaluation of two slightly different response turned out to be somewhat problematic. An example of this is noticed in the evaluation of deviating outputs: how great difference is acceptable? The problem is identical to voting methods in diverse systems. Thus it is reasonable to suggest that with a larger system the approach should be more targeted to critical or otherwise interesting portions of the system. It also seems obvious that in later versions of the test harness a more intelligent response comparator will be needed.

Dynamic testing gives valuable information of the time behaviour of the target system. This type of test result can not be achieved by any other means. For instance, analysis of the specification or system design can not fully predict its time related behaviour. Proving of the response times inevitably requires dynamic testing of the real system.

The pilot system included features that were not described in the specification for quotation available for this study. This means that a similar system cannot be reconstructed using only this specification. In this case the pilot system contains probably some sort of filtering of the input signals that is not defined in this specification. The design documents produced in the later phase of the system design process would have defined these filters, but they were not available for this study.  The lack of requirements traceability is a severe drawback: if specification can not be used as a starting point for the logical modelling, all the possible errors that have been made during the early phases of the system development, are missed in dynamic testing. It is a well known fact that the origin of most of the errors lies just in the specification and design phases of the development process. Some references claim that even 80 % of errors stem from specification phase. Thus — according to those references — most of the potential errors can not be found in dynamic testing, if the logical model is based on implementation documentation, i.e. the formal specification.

Therefore it is suggested that the specifications in the future also include features like filtering. The decisions to add new features to the system are perhaps done during implementation (that is, during the development of formal specification). However, there are no obstacles in updating the specifications after this kind of design or implementation decisions have been made. International norms and standards should be developed to give more clear guidance for the production of correct and complete specifications.

Test case preparation and selection should be based on the analysis of the system. Test cases should be targeted to key areas of the system and selected so that statistical analysis of test results is possible.

The experiences gained in this trial testing of one application it is clear that further development of the test harness is still needed, especially concerning the comparison of the actual behaviour of the system with the expected response provided by the logical model. The tested application was of utmost simplicity and more experience is still needed before the full potential of the method and the harness can be evaluated and the development tasks defined. Another, more complicated target system will be tested during the spring 1995. That test will most certainly give a lot of new experience for that purpose.

# REFERENCES

Abott 1992. The role of dynamic testing in the certification of software based safety critical systems. In: IAEA-TECDOC-780, “Safety assessment of computerized control and protection system”, 7 pp. Vienna, 12—16 October 1992.

Anderson J-O 1993. Experience from licensing and installing programmable electronic in the power range monitoring safety system in Barsebäck NPP. EHPG Meeting, Storefjell, Norway, 8—12.3.1993: 1—7 + app. 8 pp.

ASEA ATOM 1988. Barsebäck 1, PRM-electronics. Technical specifications for quotation, RKA 87-463E (Confidential). 17 pp. + app. 27 pp.

Haapanen, Korhonen 1994. Specifications of a dynamic test harness for programmable, safety critical systems. VTT Working Report. 37 pp.+ app. 34 pp.

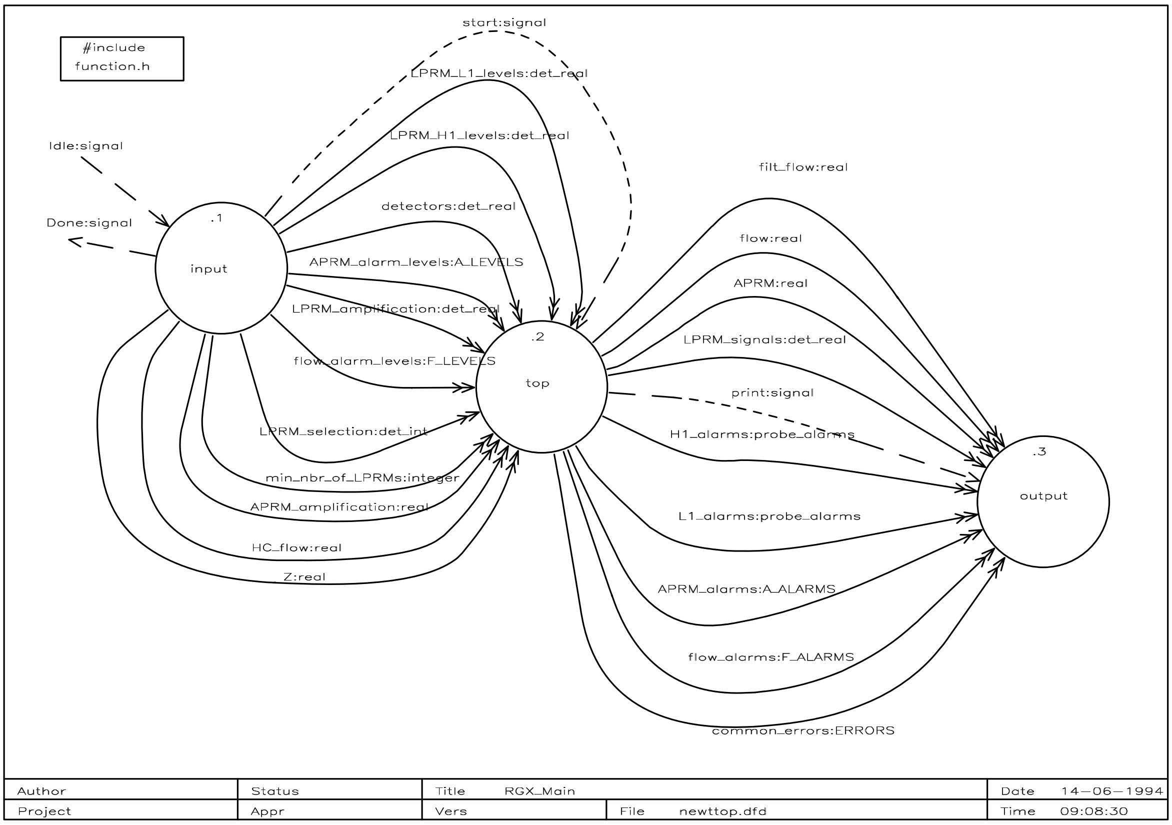
Haapanen, Heikkinen, Korhonen, Maskuniitty, Pulkkinen, Tuulari 1995: Feasibility studies of safety assessment methods for programmable automation systems. Final Report of the AVV project. STUK-YTO-TR 93. In press.

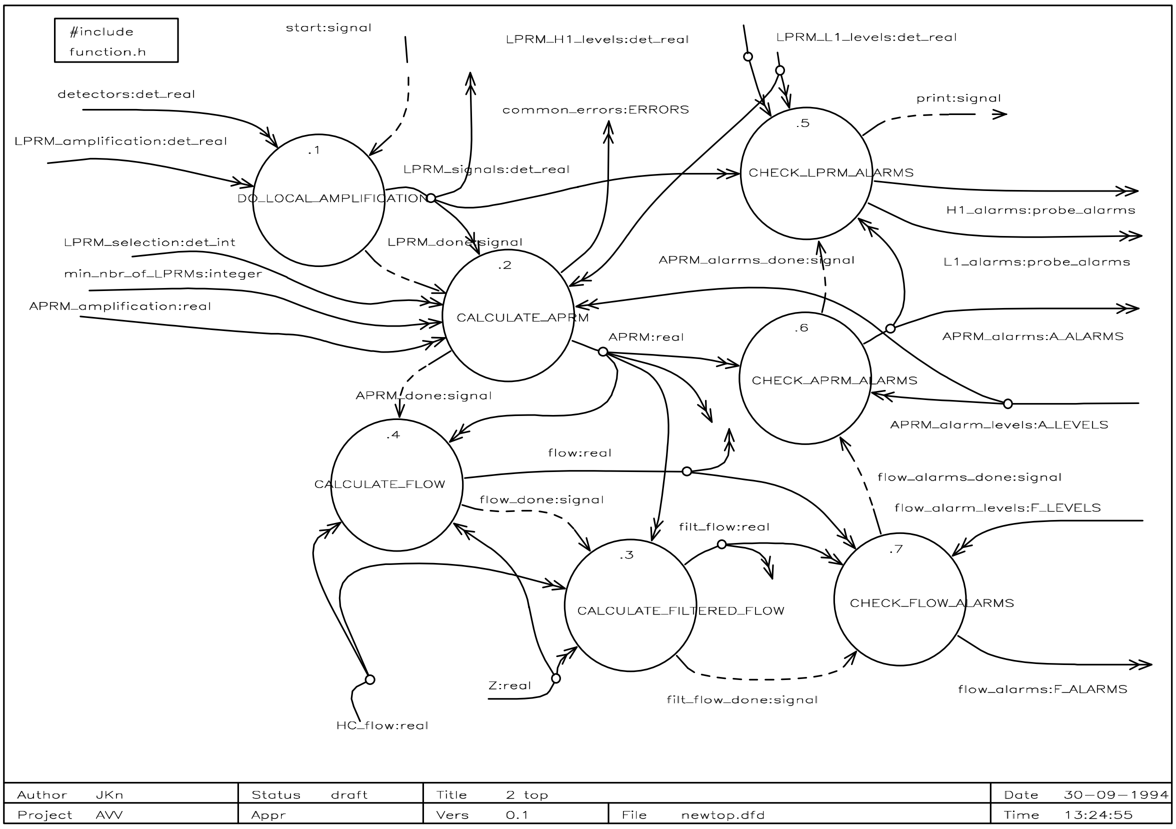
Prosa 1989. Prosa structured analysis. User’s Manual. Insoft Ky. July 1989.

ReaGeniX Programmer 1994. User’s Manual. VTT Electronics.

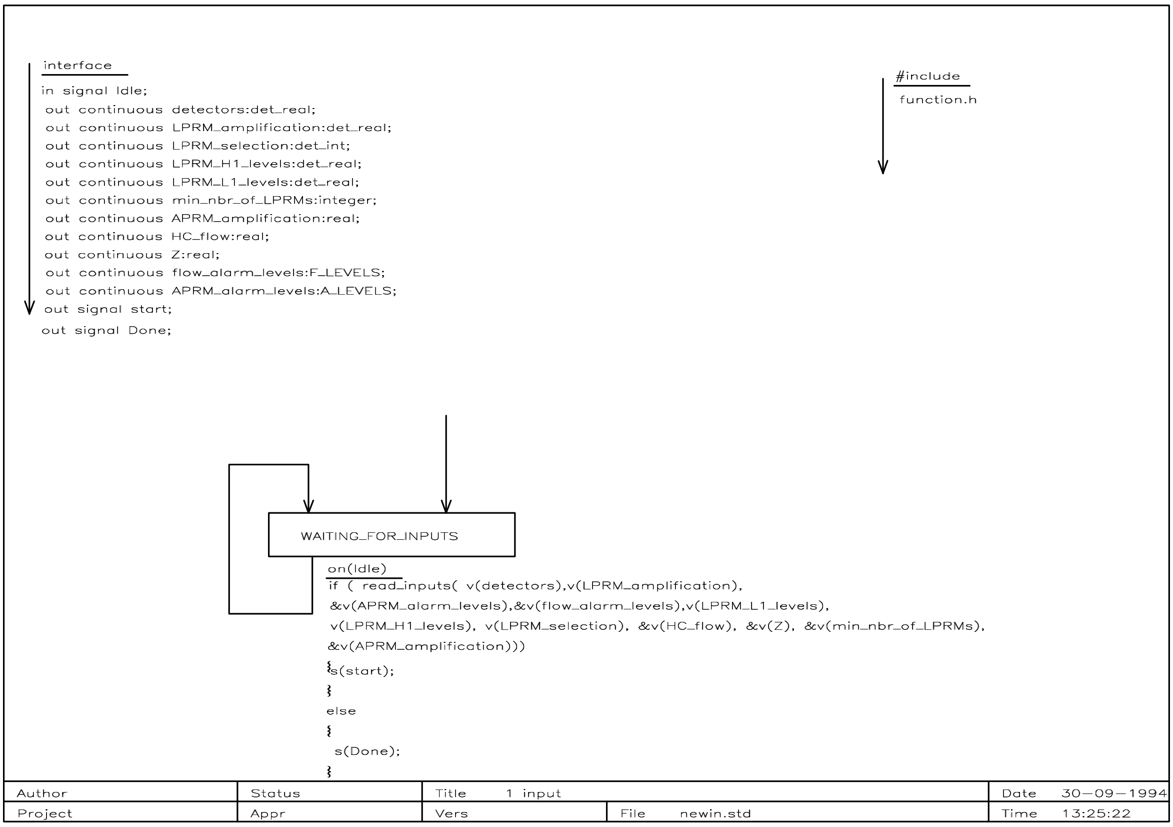
# THE LOGICAL MODEL

# APPENDIX A





Data Flow Diagrams

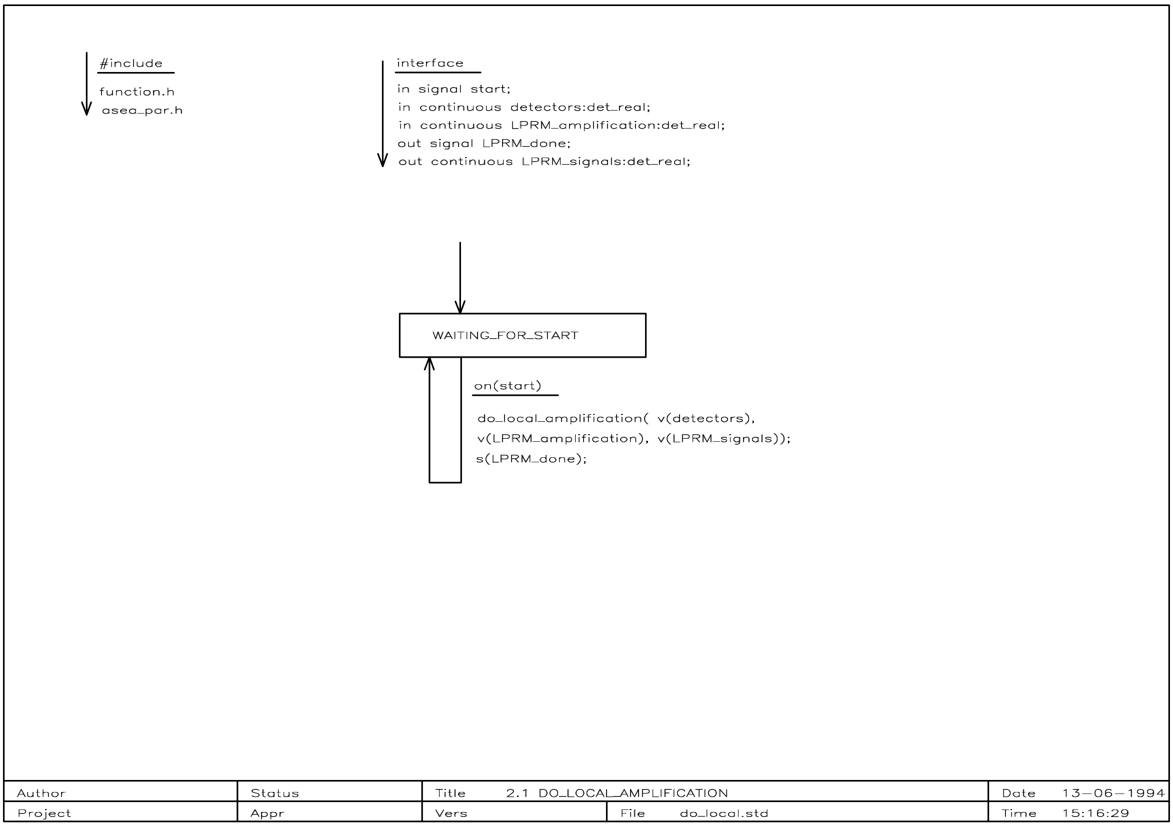


```
interface
in signal Idle;
 out continuous detectors:det_real;
 out continuous LPRM_amplification:det_real;
 out continuous LPRM_selection:det_int;
 out continuous LPRM_H1_levels:det_real;
 out continuous LPRM_L1_levels:det_real;
 out continuous min_nbr_of_LPRMs:integer;
 out continuous APRM_amplification:real;
 out continuous HC_flow:real;
 out continuous Z:real;
 out continuous flow_alarm_levels:F_LEVELS;
 out continuous APRM_alarm_levels:A_LEVELS;
 out signal start;
out signal Done;
```

```
#include
 function.h
```

```
on(Idle)
if ( read_inputs( v(detectors),v(LPRM_amplification),
 &v(APRM_alarm_levels),&v(flow_alarm_levels),v(LPRM_L1_levels),
 v(LPRM_H1_levels), v(LPRM_selection), &v(HC_flow), &v(Z), &v(min_nbr_of_LPRMs),
&v(APRM_amplification)))
{s(start);
}
else
{
 s(Done);
}
```

| | | | | |
|---|---|---|---|---|
| Author | Status | Title 1 input | | Date 30–09–1994 |
| Project | Appr | Vers | File newin.std | Time 13:25:22 |



```
#include
function.h
asea_par.h
```

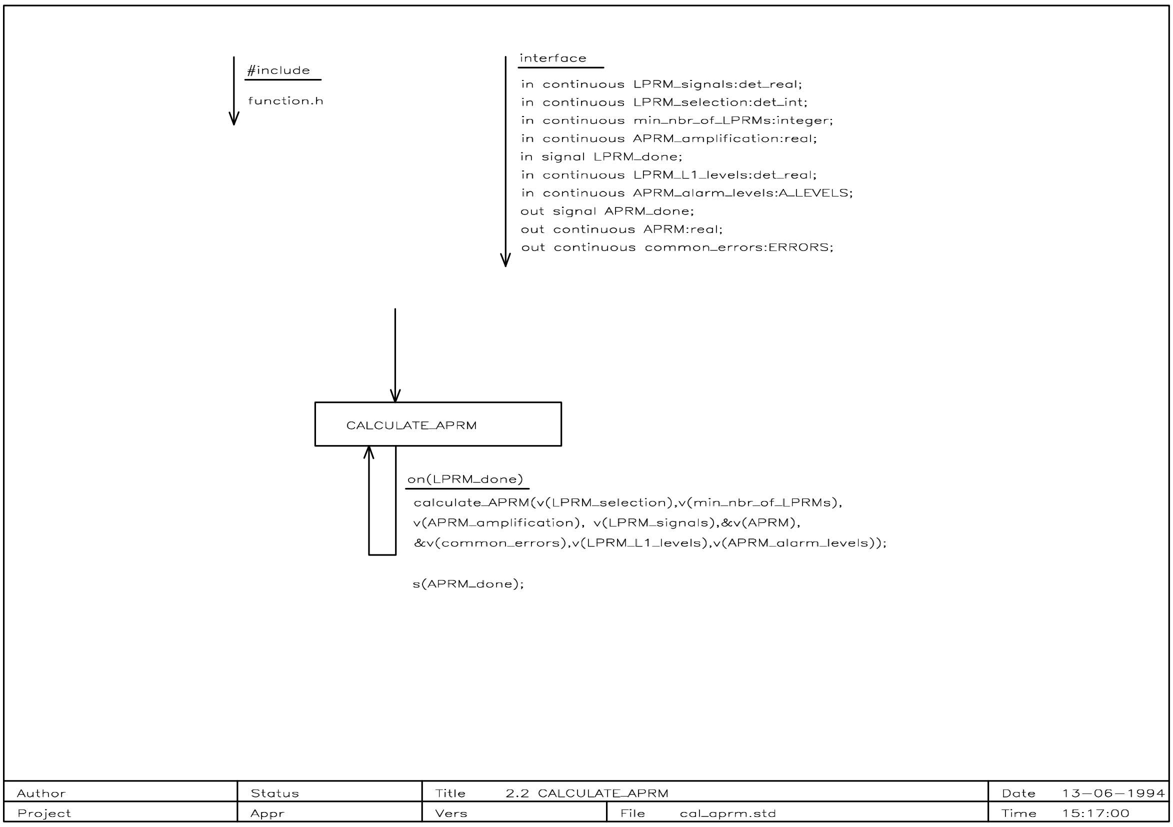
```
interface
in signal start;
in continuous detectors:det_real;
in continuous LPRM_amplification:det_real;
out signal LPRM_done;
out continuous LPRM_signals:det_real;
```

```
on(start)

 do_local_amplification( v(detectors),
 v(LPRM_amplification), v(LPRM_signals));
 s(LPRM_done);
```

| | | | | |
|---|---|---|---|---|
| Author | Status | Title 2.1 DO_LOCAL_AMPLIFICATION | | Date 13–06–1994 |
| Project | Appr | Vers | File do_local.std | Time 15:16:29 |

State Transition Diagrams



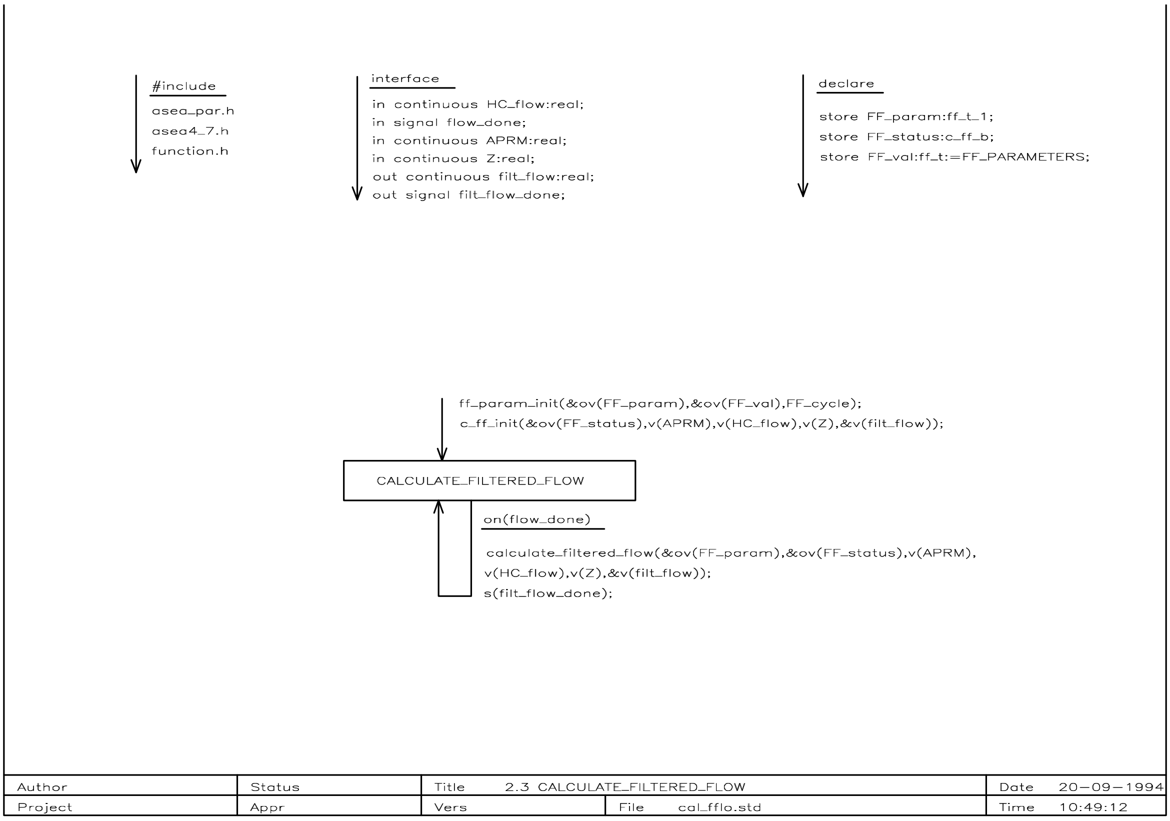
```
#include
function.h
```

```
interface
in continuous LPRM_signals:det_real;
in continuous LPRM_selection:det_int;
in continuous min_nbr_of_LPRMs:integer;
in continuous APRM_amplification:real;
in signal LPRM_done;
in continuous LPRM_L1_levels:det_real;
in continuous APRM_alarm_levels:A_LEVELS;
out signal APRM_done;
out continuous APRM:real;
out continuous common_errors:ERRORS;
```

```
on(LPRM_done)
calculate_APRM(v(LPRM_selection),v(min_nbr_of_LPRMs),
v(APRM_amplification), v(LPRM_signals),&v(APRM),
&v(common_errors),v(LPRM_L1_levels),v(APRM_alarm_levels));

s(APRM_done);
```

| | | | | |
|---|---|---|---|---|
| Author | Status | Title 2.2 CALCULATE_APRM | | Date 13−06−1994 |
| Project | Appr | Vers | File cal_aprm.std | Time 15:17:00 |



```
#include
asea_par.h
asea4_7.h
function.h
```

```
interface
in continuous HC_flow:real;
in signal flow_done;
in continuous APRM:real;
in continuous Z:real;
out continuous filt_flow:real;
out signal filt_flow_done;
```

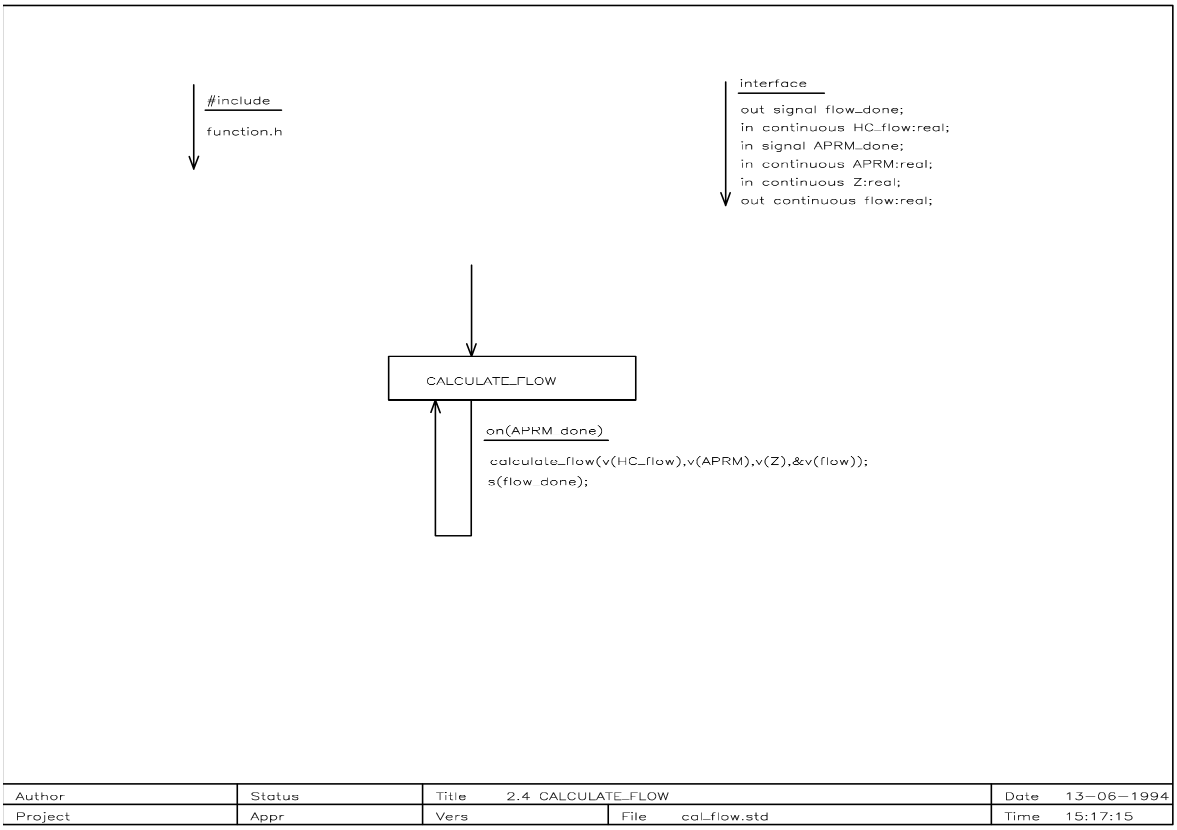
```
declare
store FF_param:ff_t_1;
store FF_status:c_ff_b;
store FF_val:ff_t:=FF_PARAMETERS;
```

```
ff_param_init(&ov(FF_param),&ov(FF_val),FF_cycle);
c_ff_init(&ov(FF_status),v(APRM),v(HC_flow),v(Z),&v(filt_flow));
```

```
on(flow_done)
calculate_filtered_flow(&ov(FF_param),&ov(FF_status),v(APRM),
v(HC_flow),v(Z),&v(filt_flow));
s(filt_flow_done);
```

| | | | | |
|---|---|---|---|---|
| Author | Status | Title 2.3 CALCULATE_FILTERED_FLOW | | Date 20−09−1994 |
| Project | Appr | Vers | File cal_fflo.std | Time 10:49:12 |

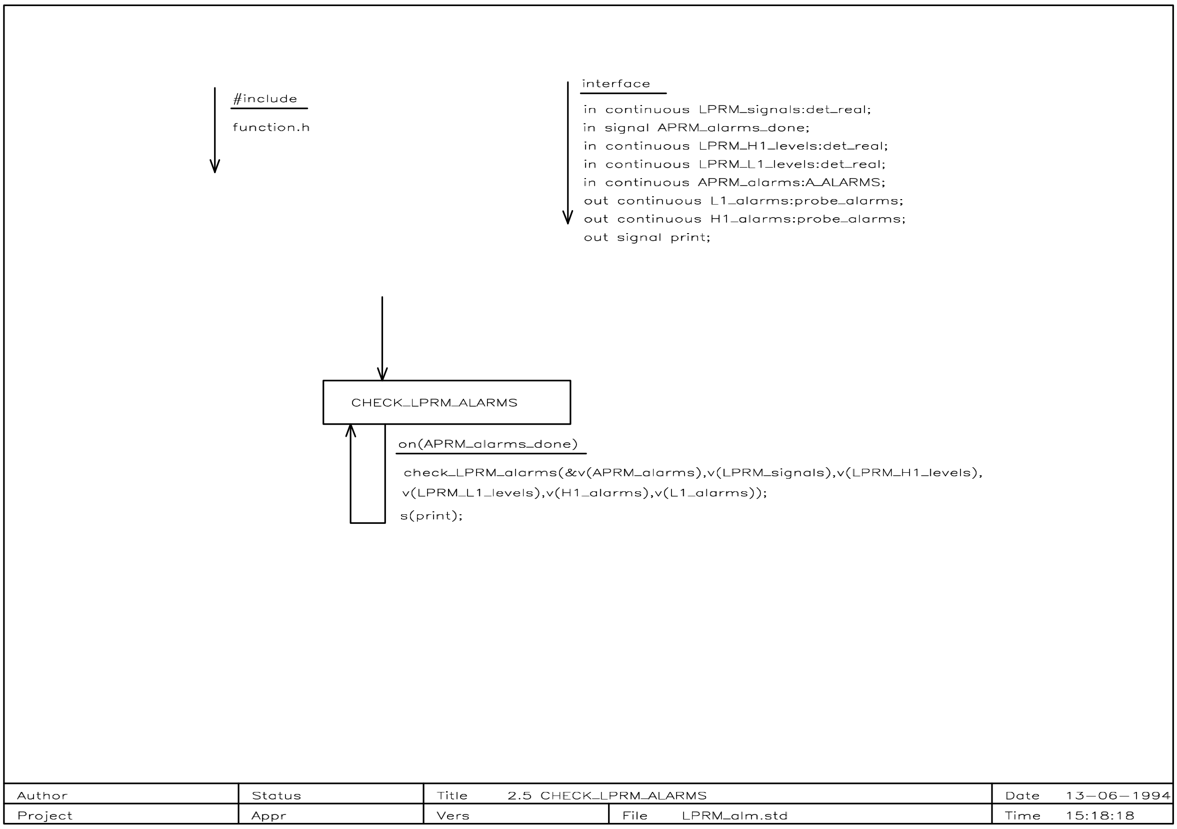
State Transition Diagrams



```
out signal flow_done;
in continuous HC_flow:real;
in signal APRM_done;
in continuous APRM:real;
in continuous Z:real;
out continuous flow:real;
```

```
calculate_flow(v(HC_flow),v(APRM),v(Z),&v(flow));
s(flow_done);
```

| | | | | |
|---|---|---|---|---|
| Author | Status | Title 2.4 CALCULATE_FLOW | | Date 13–06–1994 |
| Project | Appr | Vers | File cal_flow.std | Time 15:17:15 |

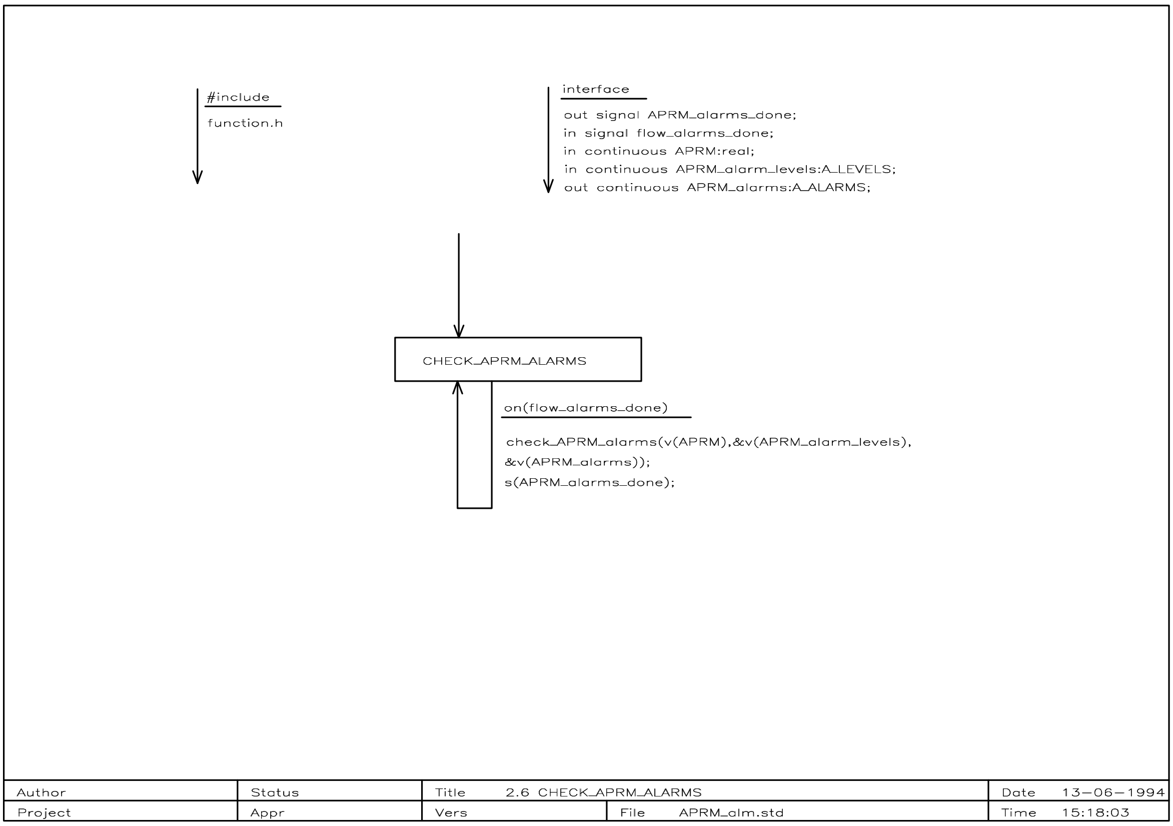


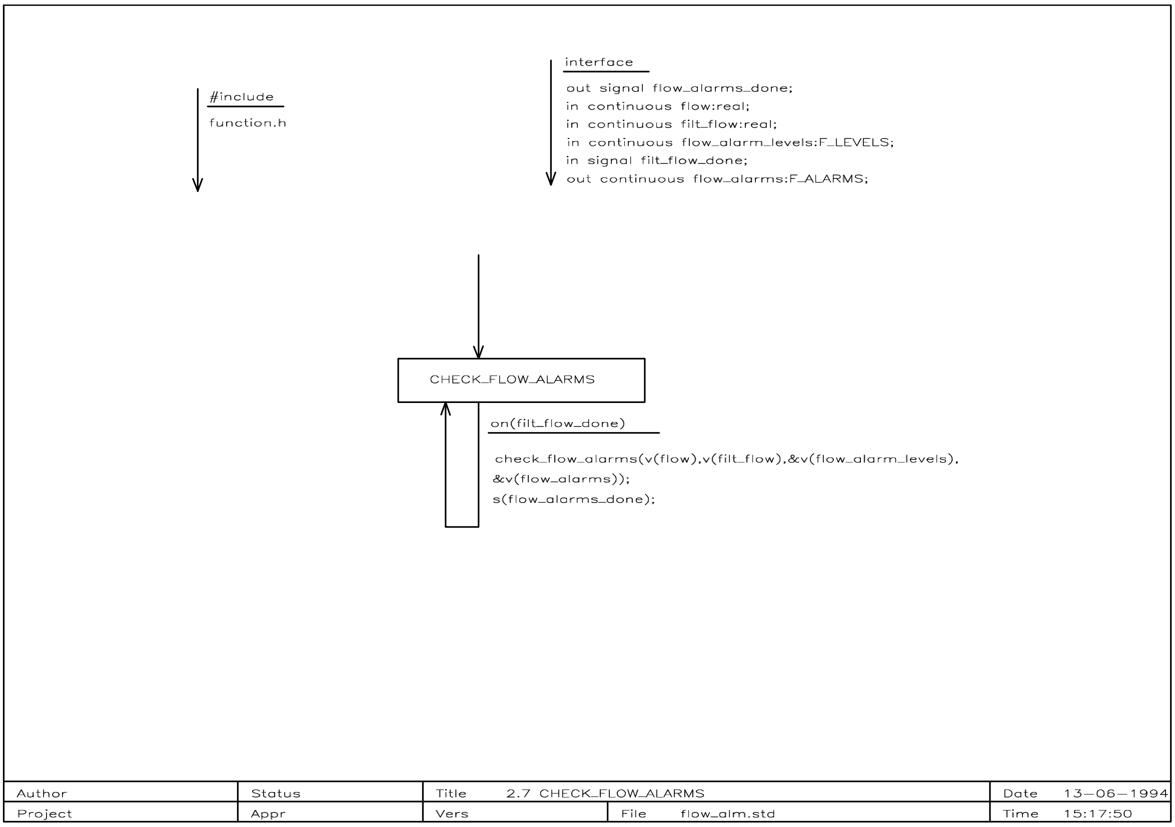
```
in continuous LPRM_signals:det_real;
in signal APRM_alarms_done;
in continuous LPRM_H1_levels:det_real;
in continuous LPRM_L1_levels:det_real;
in continuous APRM_alarms:A_ALARMS;
out continuous L1_alarms:probe_alarms;
out continuous H1_alarms:probe_alarms;
out signal print;
```

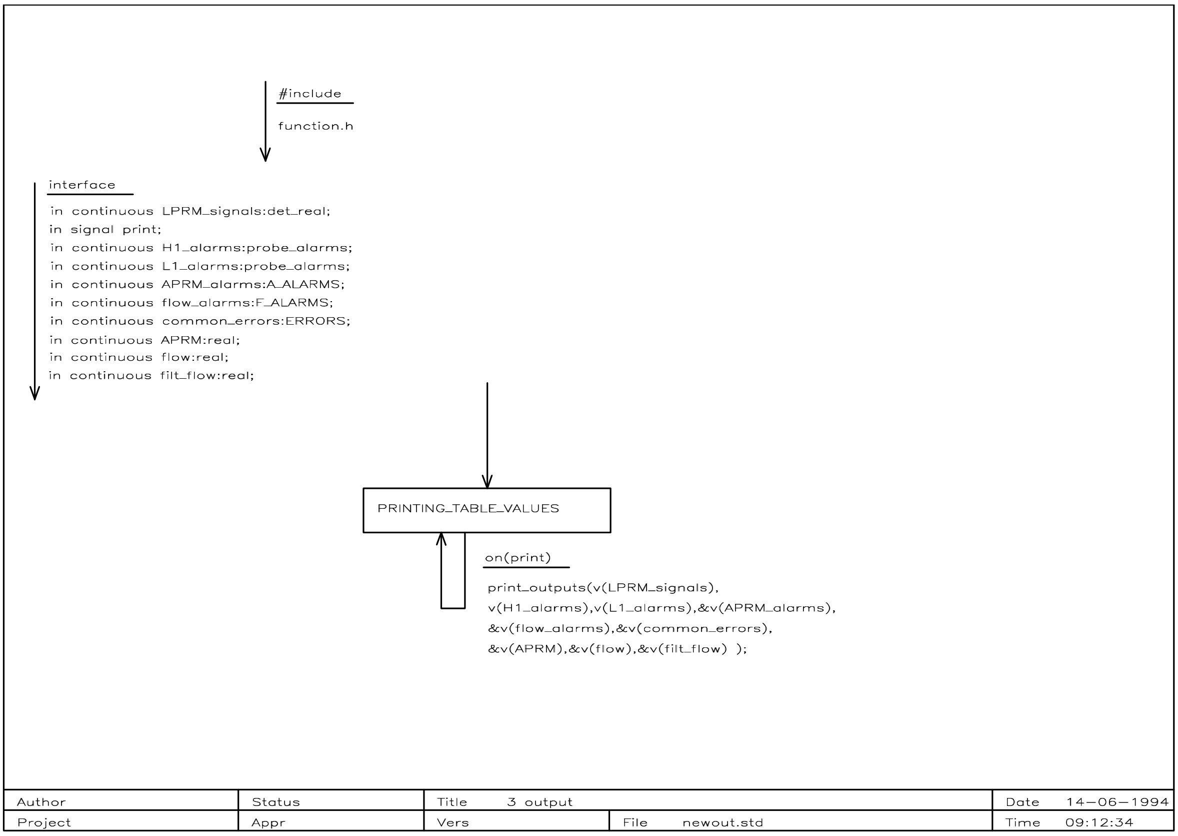
```
check_LPRM_alarms(&v(APRM_alarms),v(LPRM_signals),v(LPRM_H1_levels),
v(LPRM_L1_levels),v(H1_alarms),v(L1_alarms));
s(print);
```

| | | | | |
|---|---|---|---|---|
| Author | Status | Title 2.5 CHECK_LPRM_ALARMS | | Date 13–06–1994 |
| Project | Appr | Vers | File LPRM_alm.std | Time 15:18:18 |

State Transition Diagrams


#include
function.h
interface
out signal APRM_alarms_done;
in signal flow_alarms_done;
in continuous APRM:real;
in continuous APRM_alarm_levels:A_LEVELS;
out continuous APRM_alarms:A_ALARMS;
CHECK_APRM_ALARMS
on(flow_alarms_done)
check_APRM_alarms(v(APRM),&v(APRM_alarm_levels),
&v(APRM_alarms));
s(APRM_alarms_done);
Author
Status
Title 2.6 CHECK_APRM_ALARMS
Date 13-06-1994
Project
Appr
Vers
File APRM_alm.std
Time 15:18:03


#include
function.h
interface
out signal flow_alarms_done;
in continuous flow:real;
in continuous filt_flow:real;
in continuous flow_alarm_levels:F_LEVELS;
in signal filt_flow_done;
out continuous flow_alarms:F_ALARMS;
CHECK_FLOW_ALARMS
on(filt_flow_done)
check_flow_alarms(v(flow),v(filt_flow),&v(flow_alarm_levels),
&v(flow_alarms));
s(flow_alarms_done);
Author
Status
Title 2.7 CHECK_FLOW_ALARMS
Date 13-06-1994
Project
Appr
Vers
File flow_alm.std
Time 15:17:50



```
in continuous LPRM_signals:det_real;
in signal print;
in continuous H1_alarms:probe_alarms;
in continuous L1_alarms:probe_alarms;
in continuous APRM_alarms:A_ALARMS;
in continuous flow_alarms:F_ALARMS;
in continuous common_errors:ERRORS;
in continuous APRM:real;
in continuous flow:real;
in continuous filt_flow:real;
```

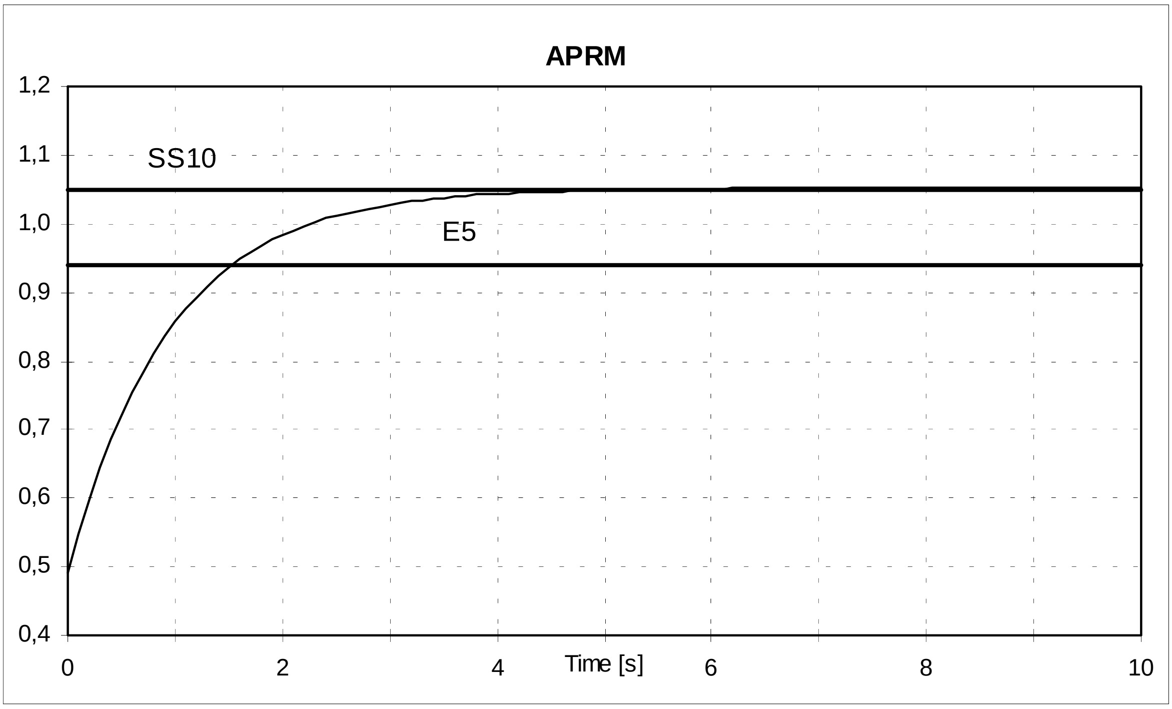
```
print_outputs(v(LPRM_signals),
v(H1_alarms),v(L1_alarms),&v(APRM_alarms),
&v(flow_alarms),&v(common_errors),
&v(APRM),&v(flow),&v(filt_flow) );
```

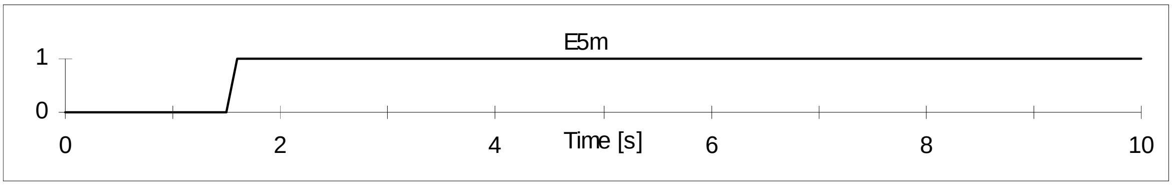
State Transition Diagrams

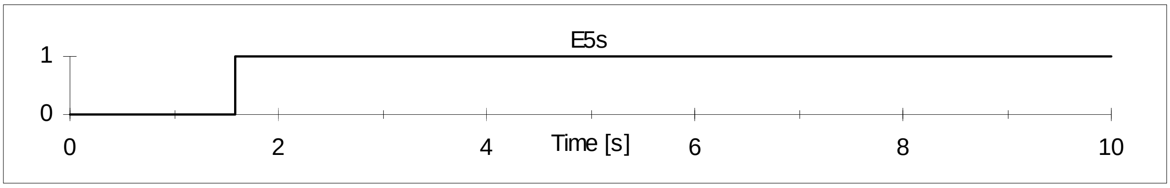
TEST DATA AND RESULTS

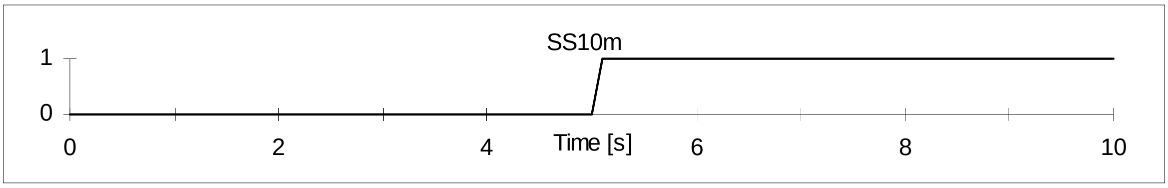
Core flow = 0.31

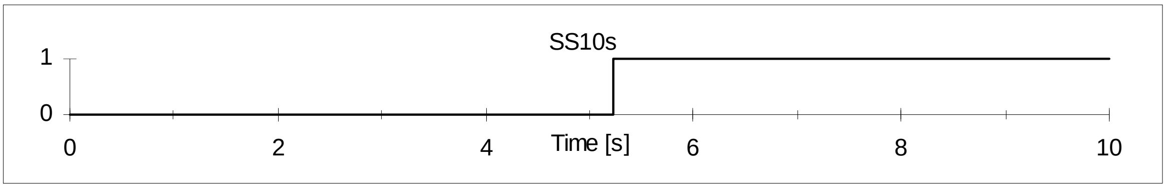
APPENDIX B





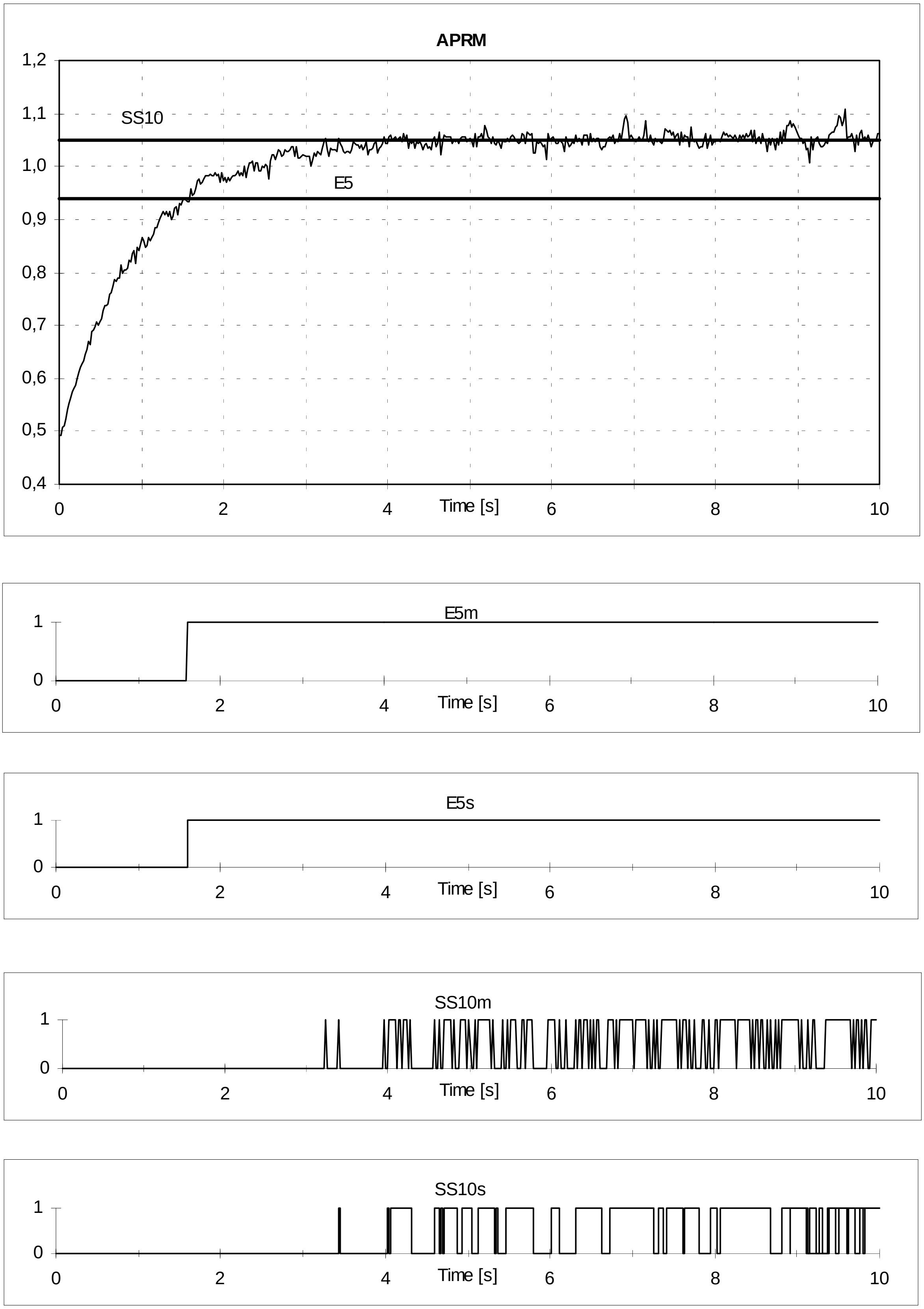






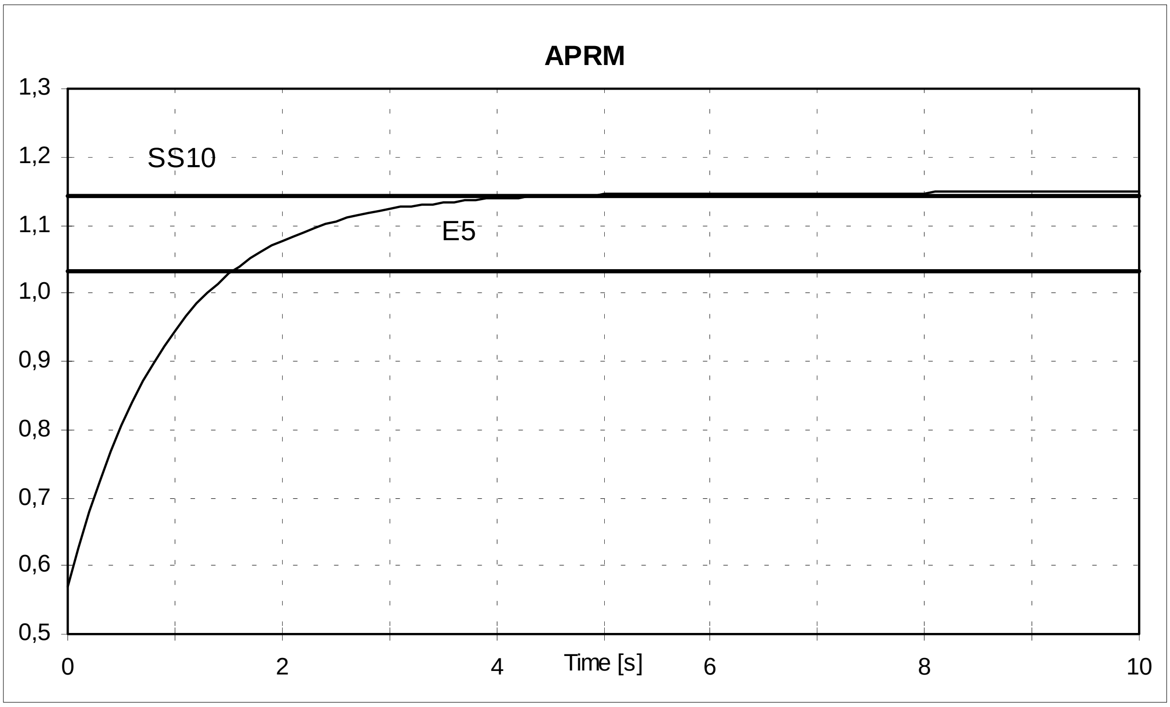
- m refers to the logical **m**odel
- s refers to the pilot **s**ystem

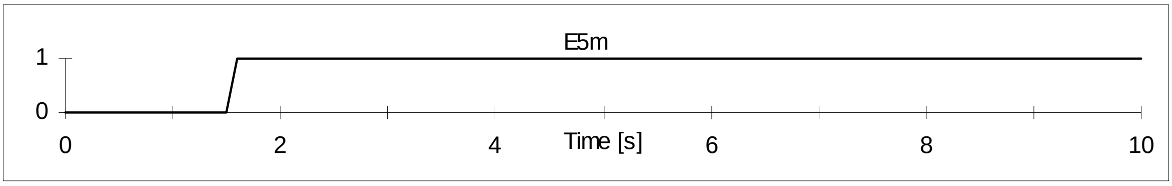
Core flow = 0.31

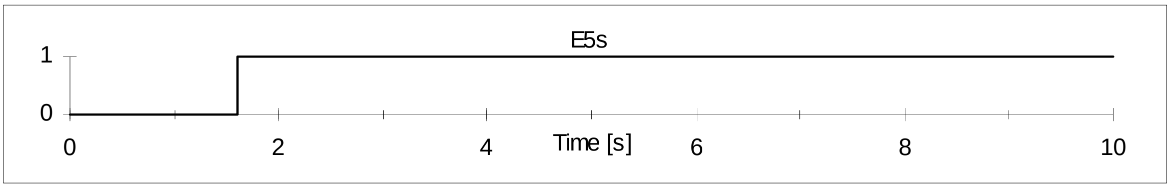


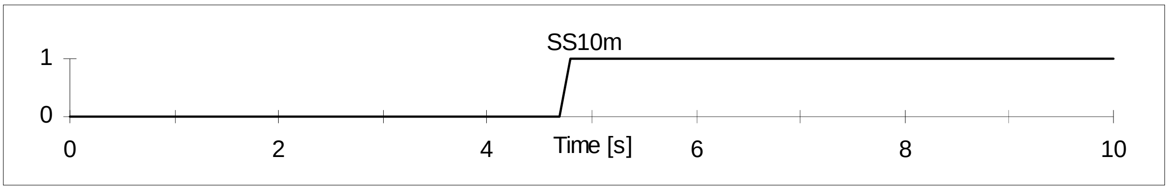
The graphs give the APRM time series calculated from the individual LPRM sensor signals feeded to the logical model and the pilot system and the responses of these.

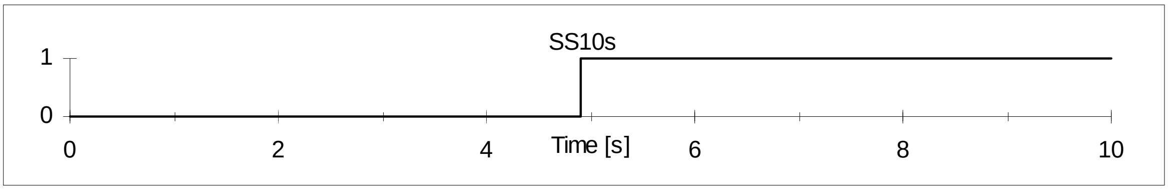
Core flow = 0.4





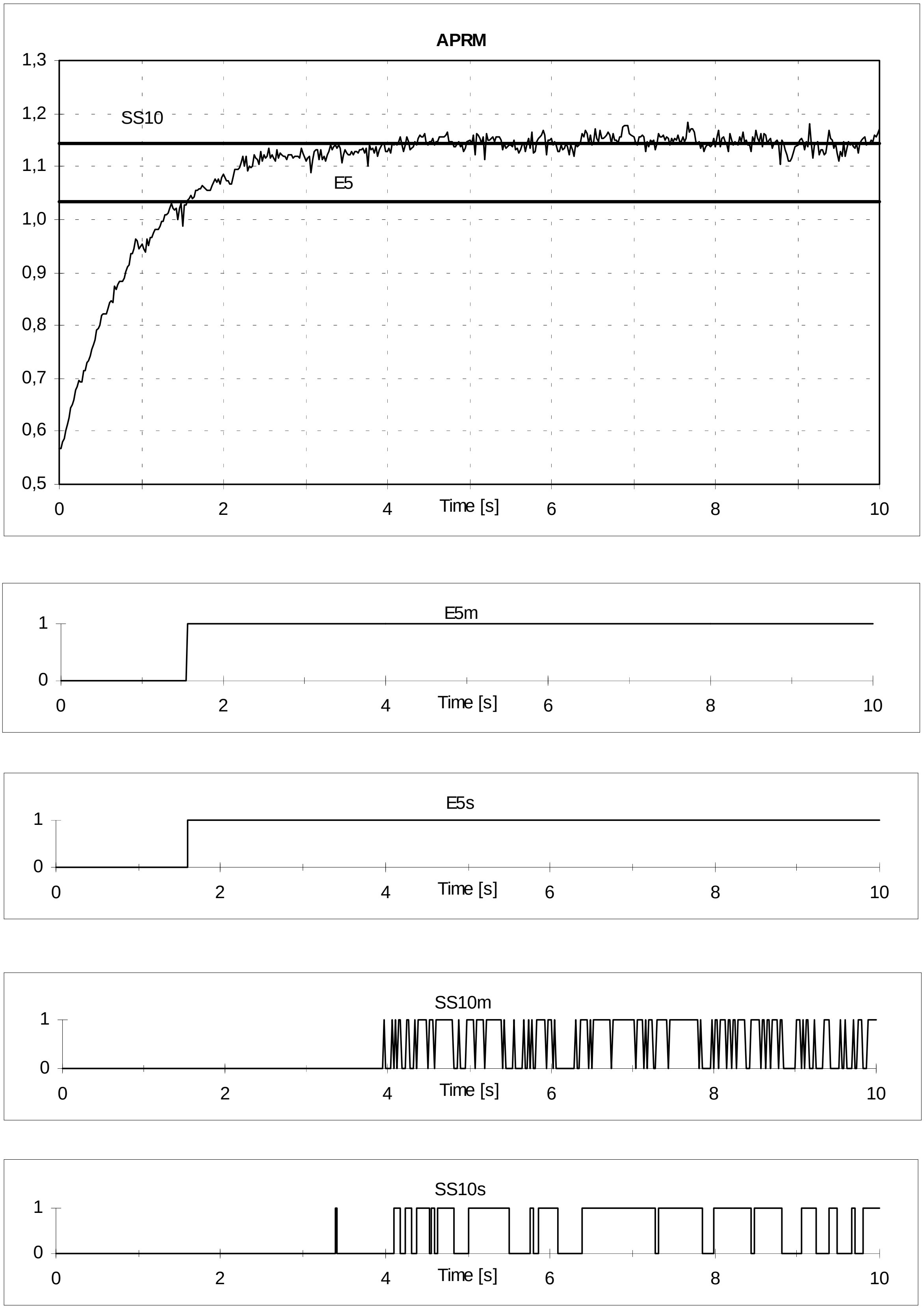






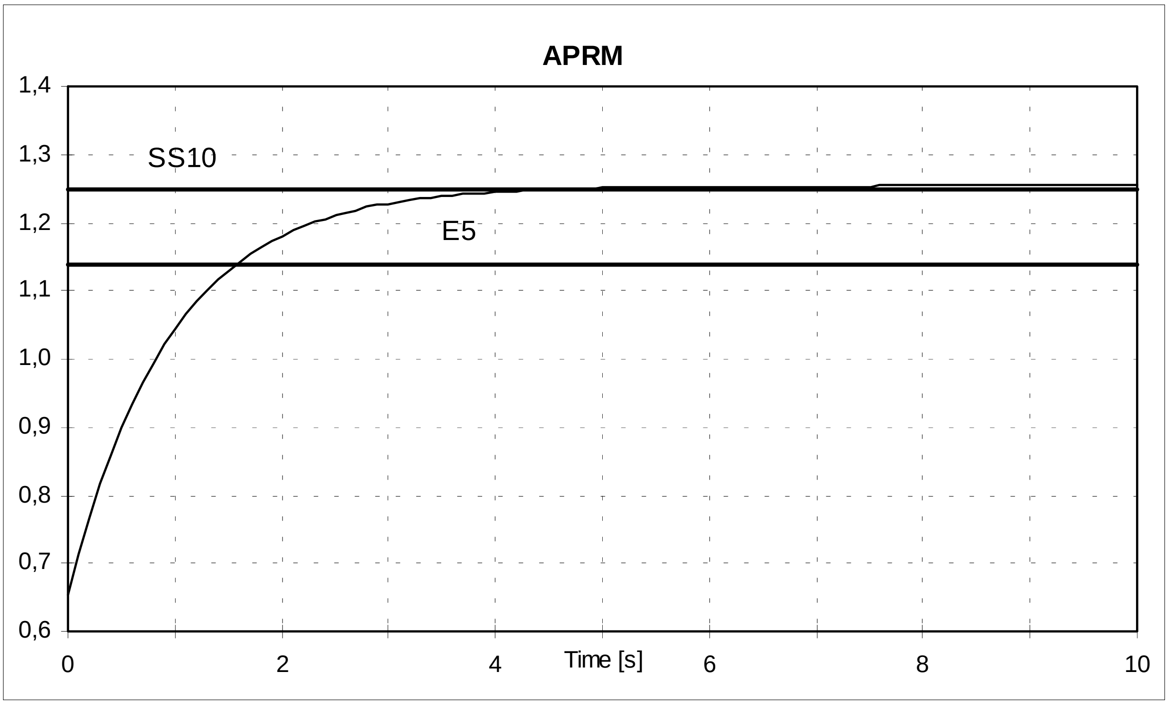
- m refers to the logical **m**odel
- s refers to the pilot **s**ystem

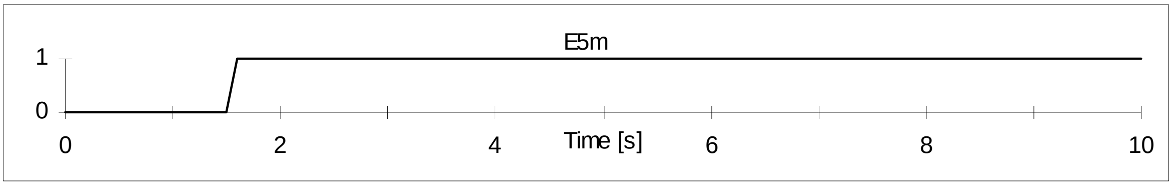
Core flow = 0.4

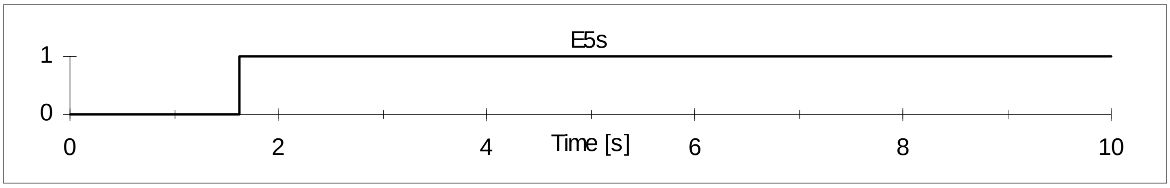


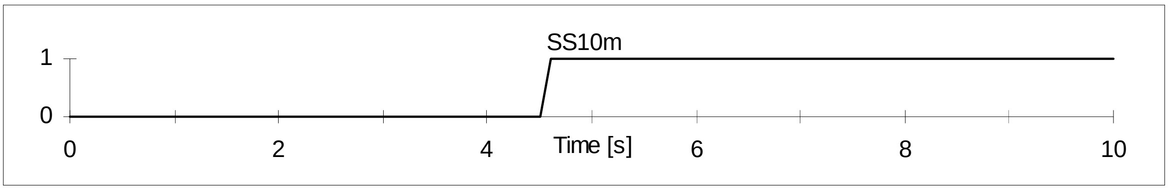
The graphs give the APRM time series calculated from the individual LPRM sensor signals feeded to the logical model and the pilot system and the responses of these.

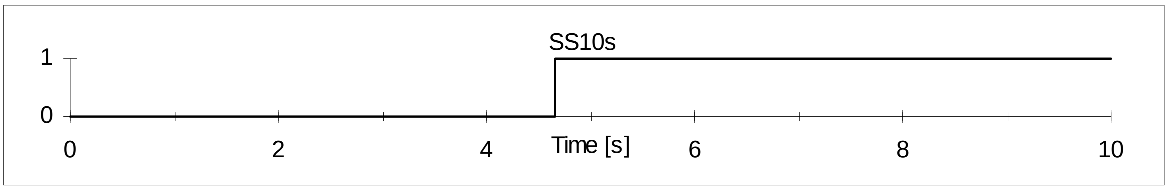
Core flow = 0.5







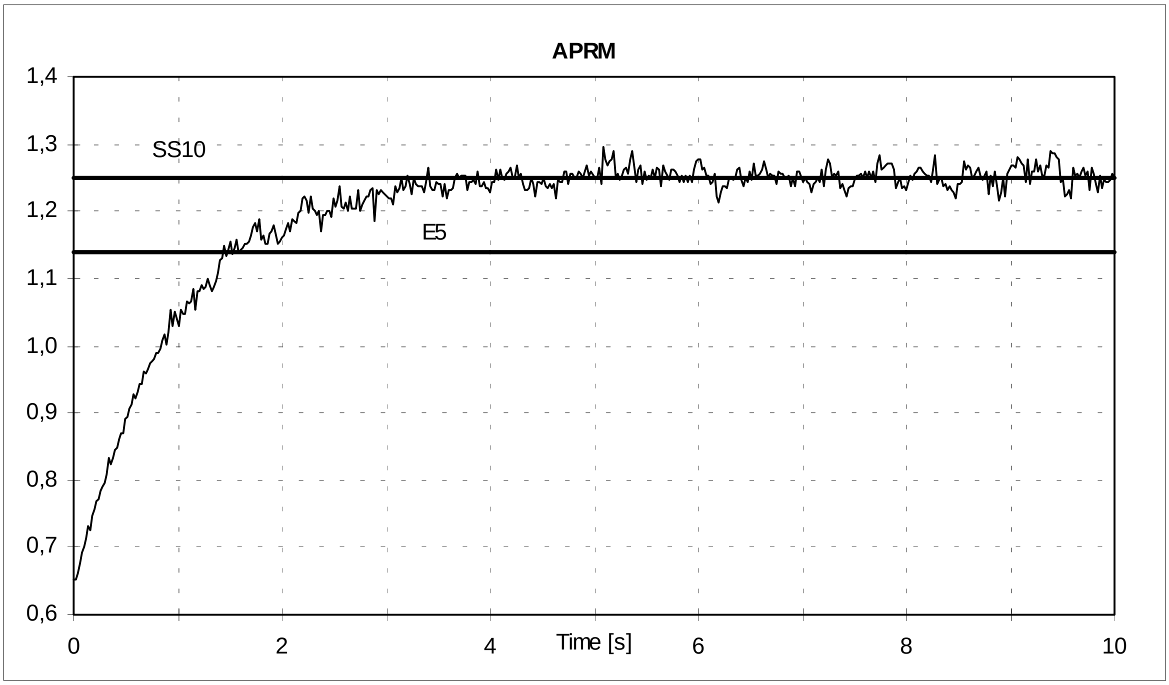


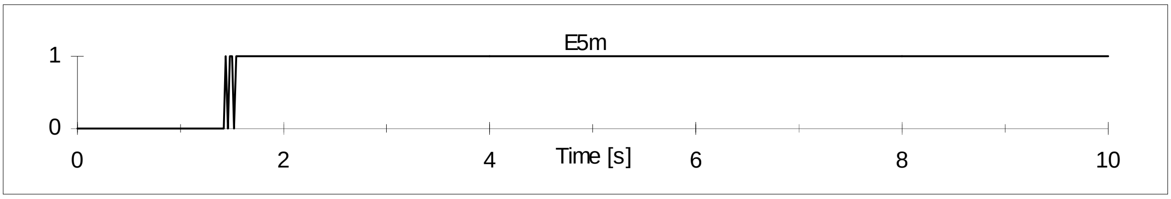


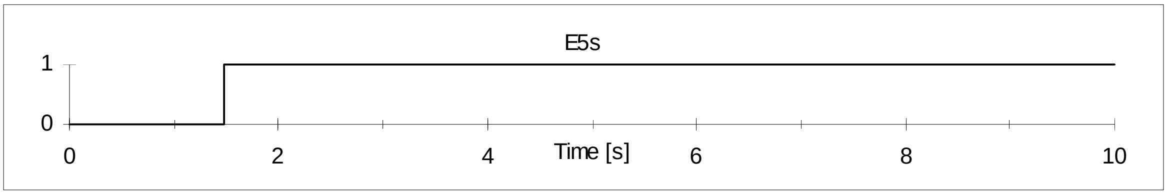
- m refers to the logical **m**odel
- s refers to the pilot **s**ystem

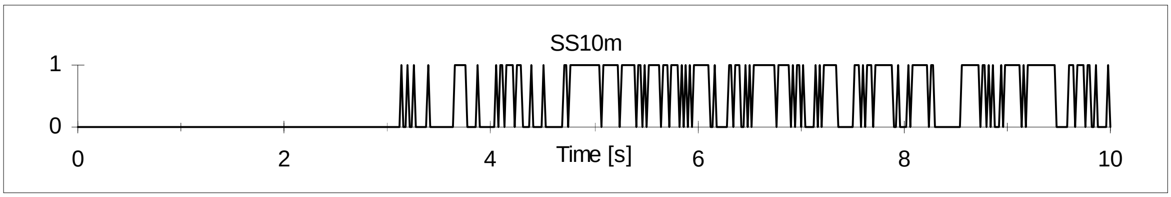
## APPENDIX B

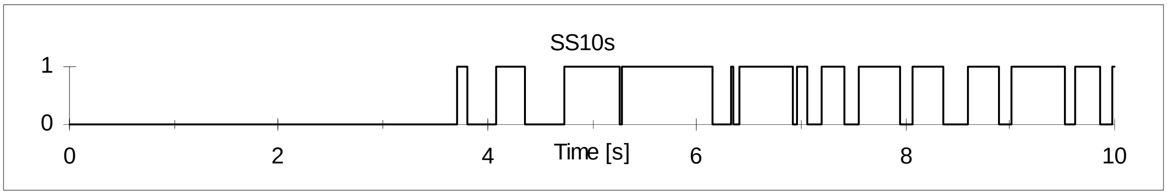
Core flow = 0.5





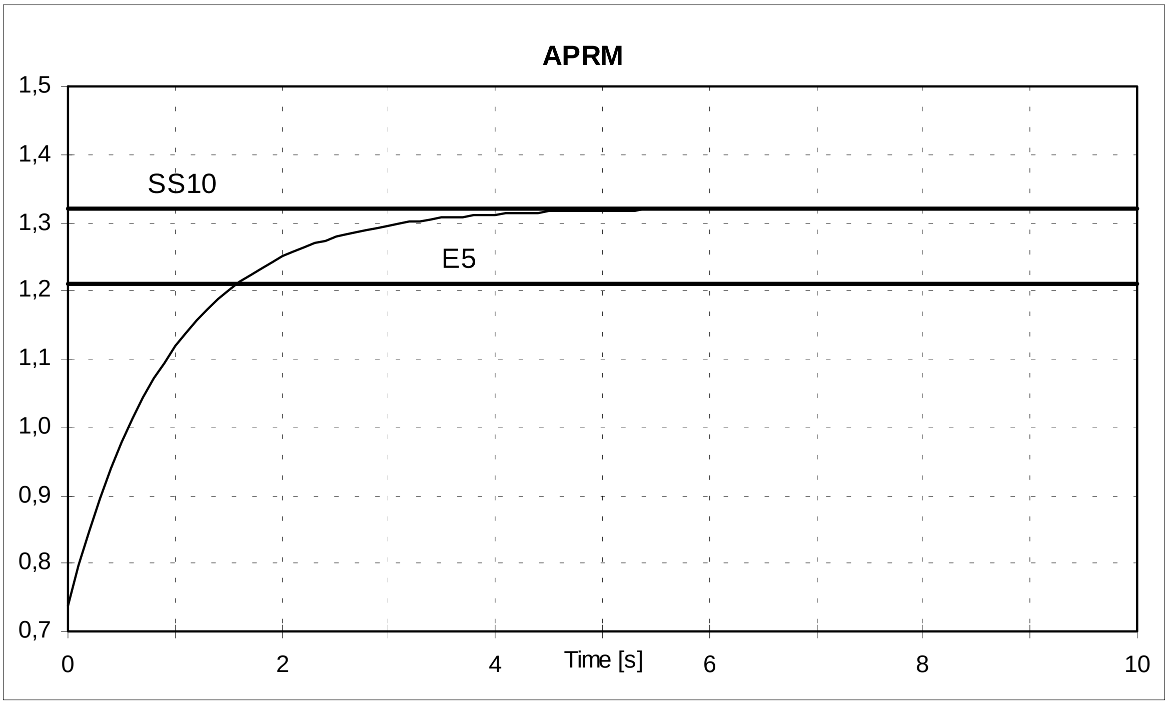


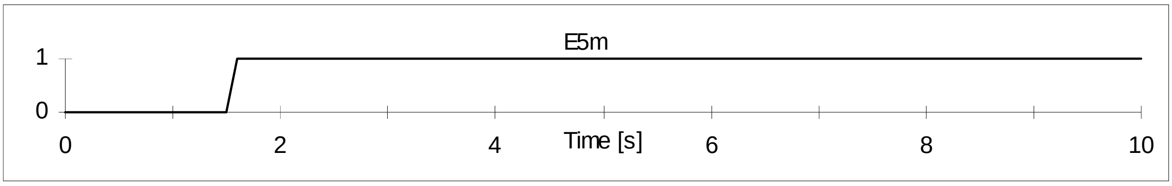


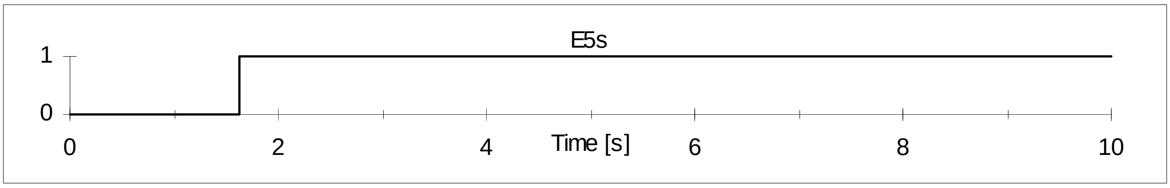


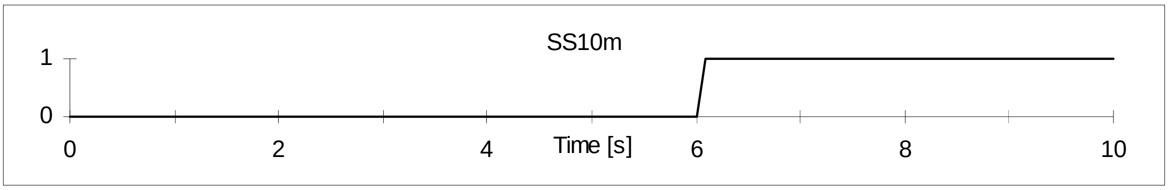
The graphs give the APRM time series calculated from the individual LPRM sensor signals feeded to the logical model and the pilot system and the responses of these.

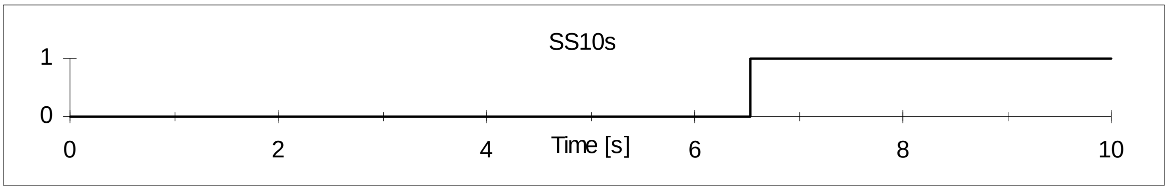
Core flow = 0.6





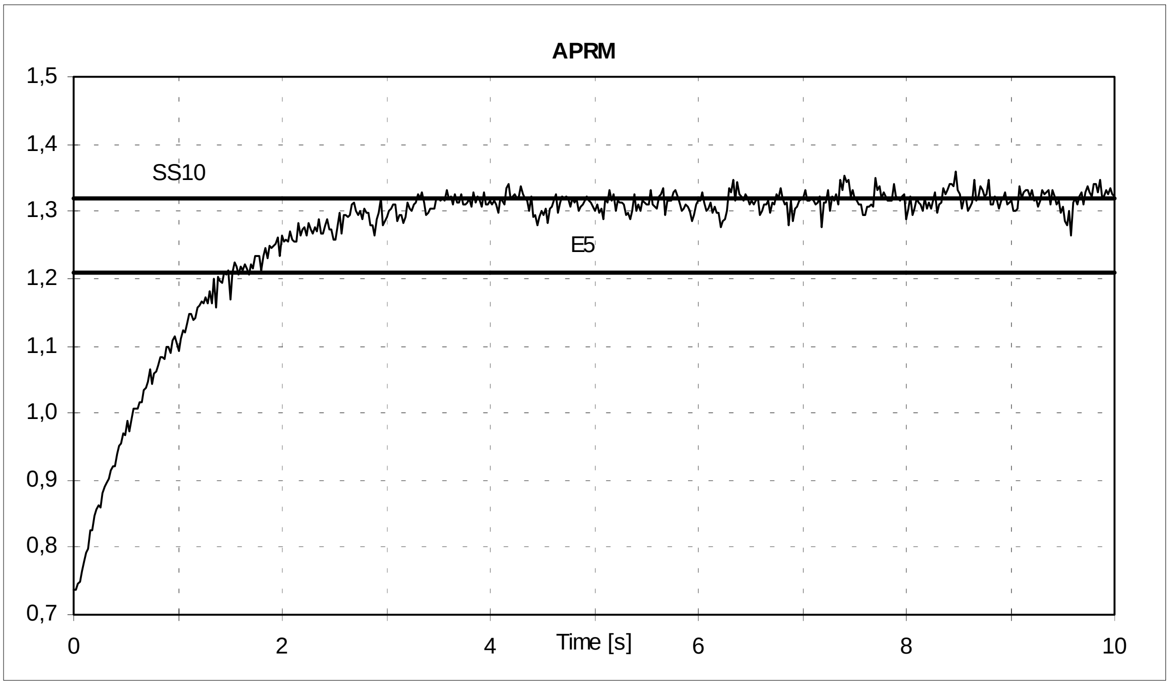


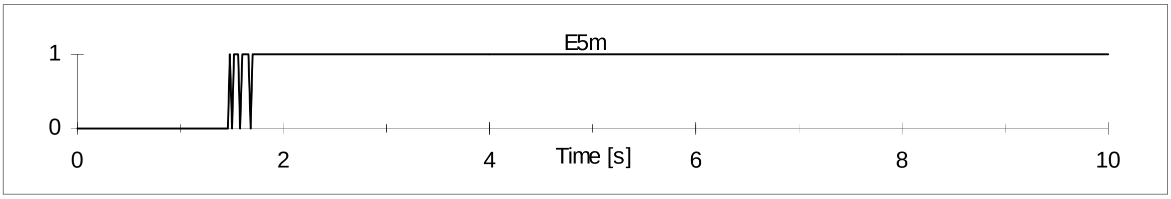


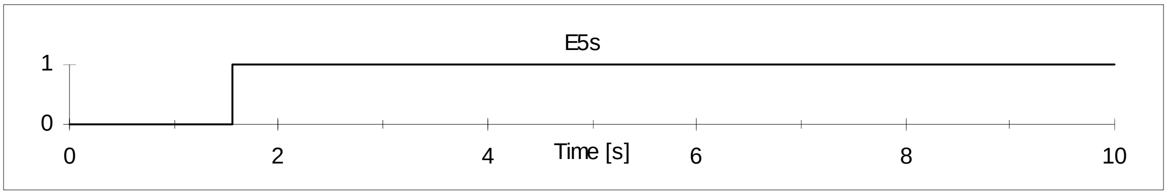


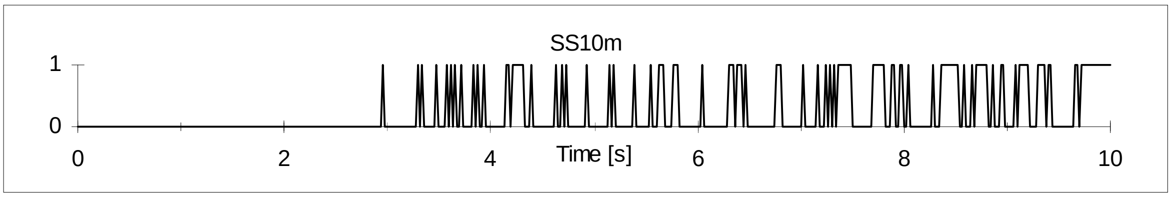
- m refers to the logical **m**odel
- s refers to the pilot **s**ystem

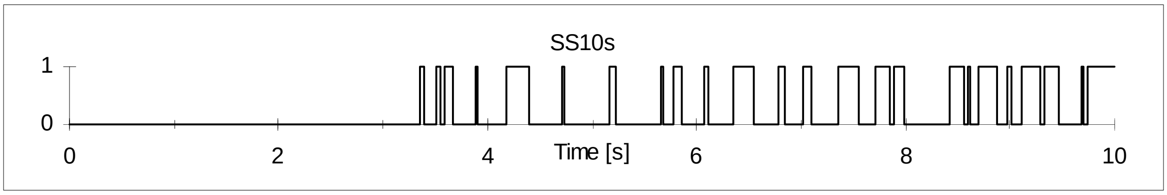
Core flow = 0.6









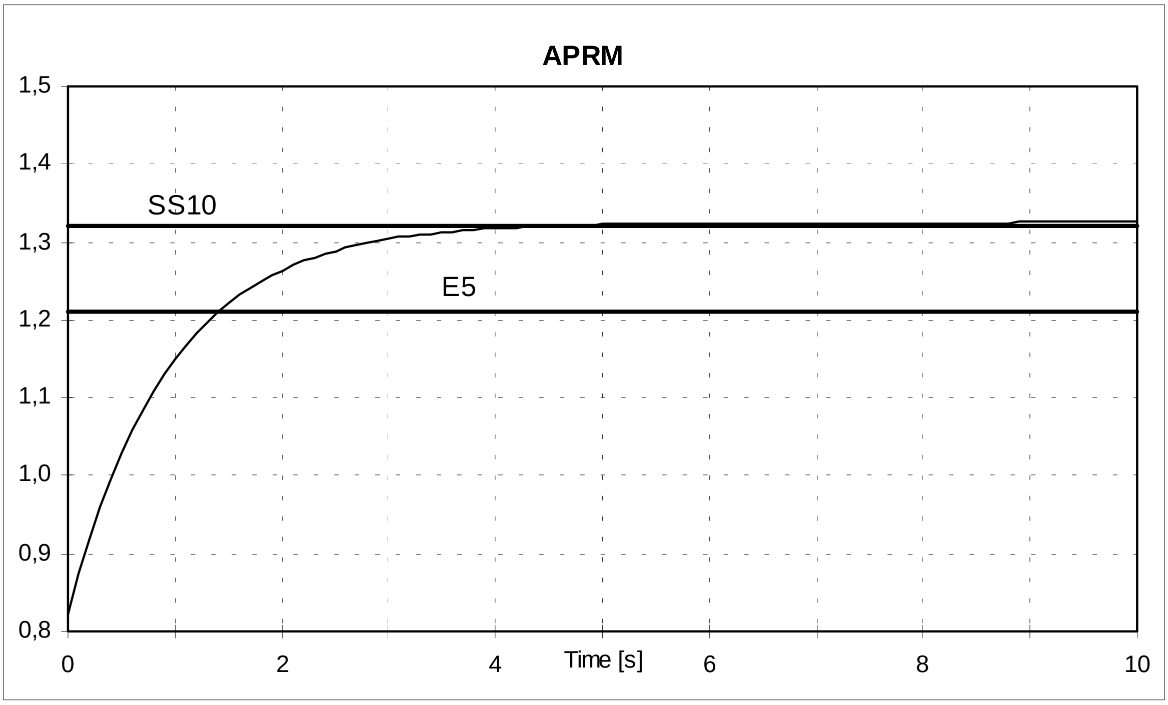


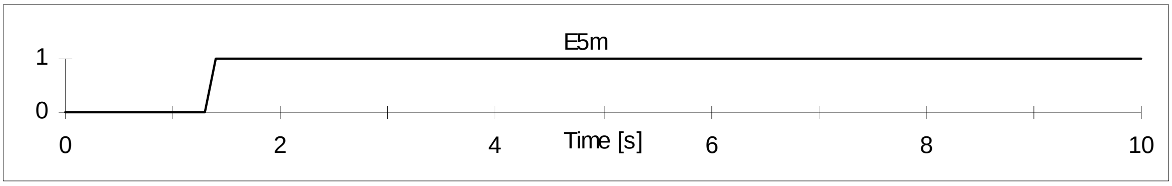
The graphs give the APRM time series calculated from the individual LPRM sensor signals feeded to the logical model and the pilot system and the responses of these.

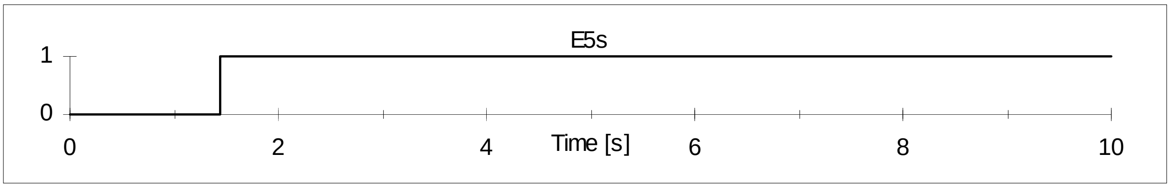
TEST DATA AND RESULTS

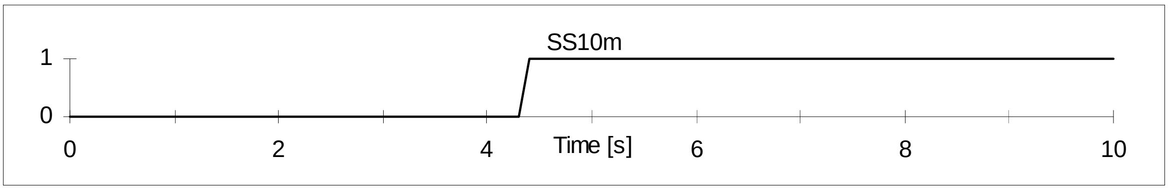
Core flow = 0.7

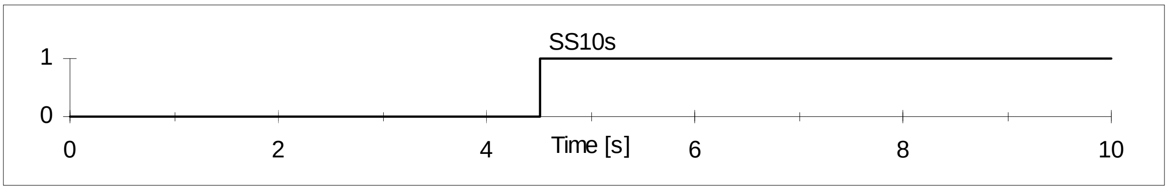
## APPENDIX B





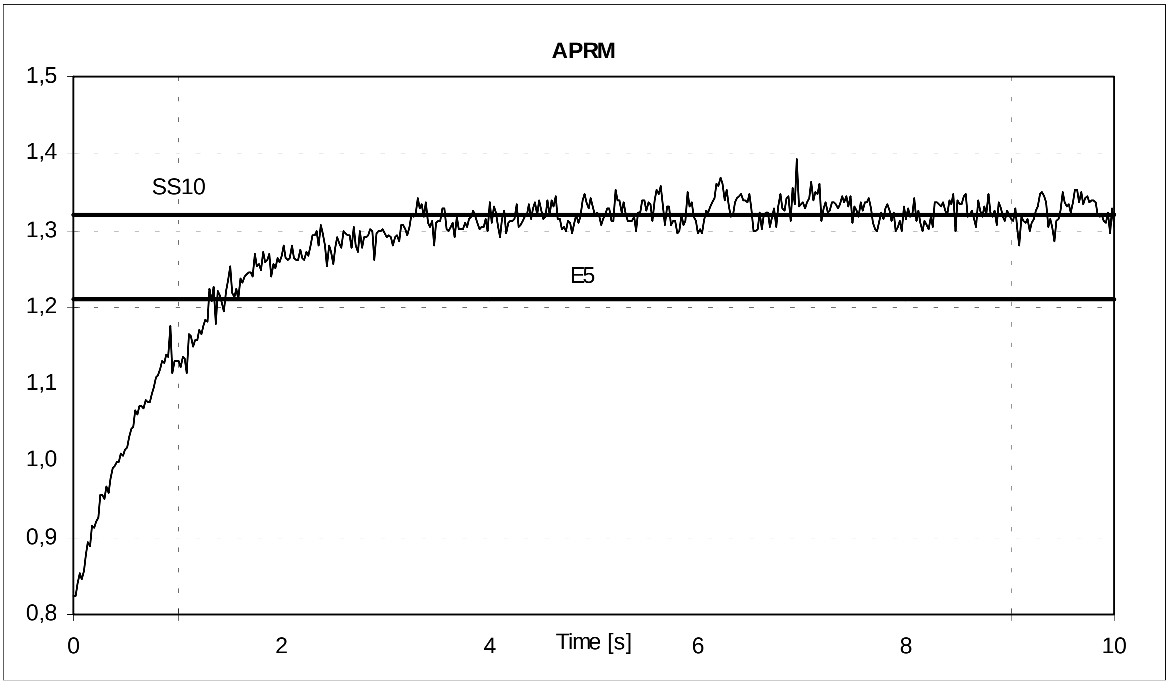


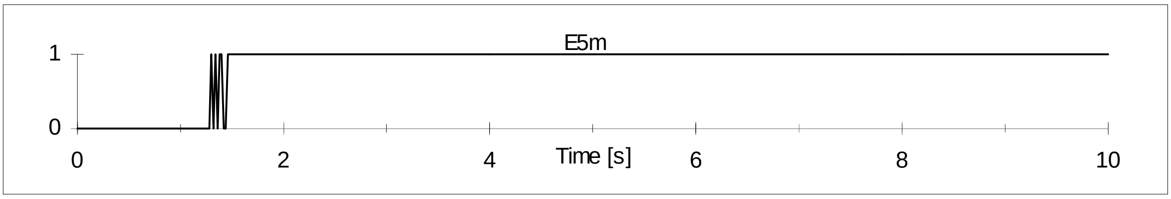


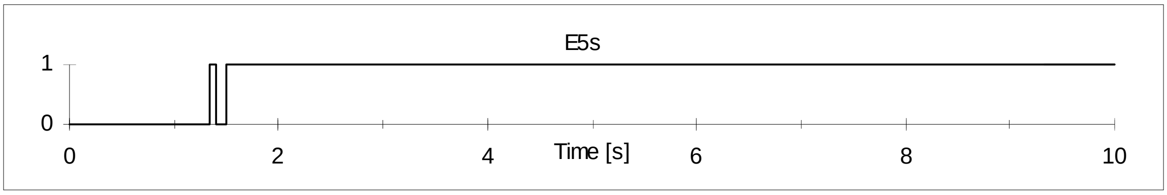


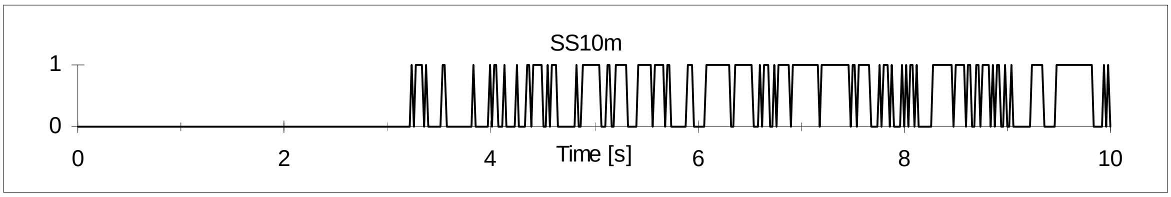
- m refers to the logical **m**odel
- s refers to the pilot **s**ystem

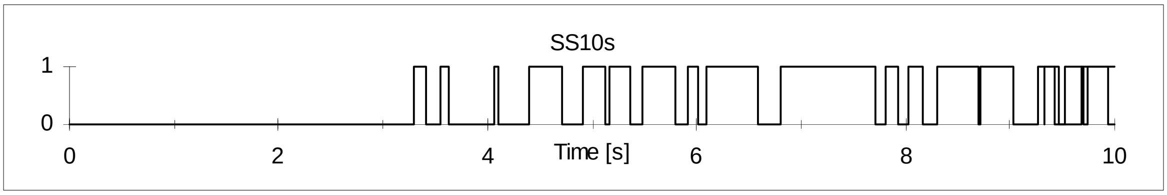
Core flow = 0.7





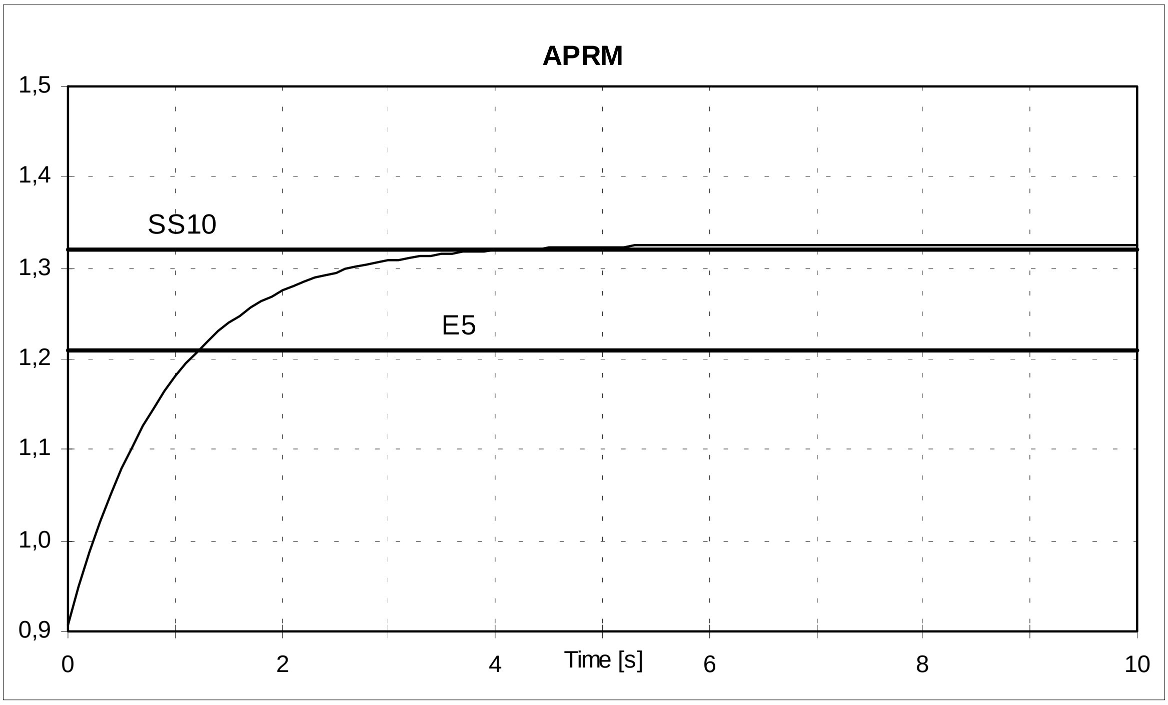


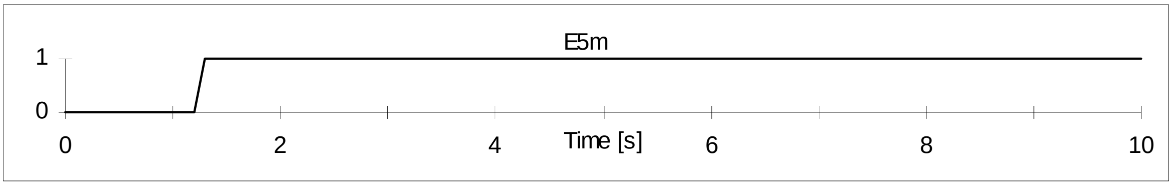


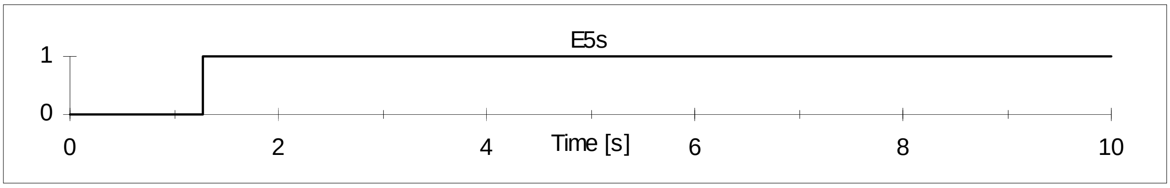


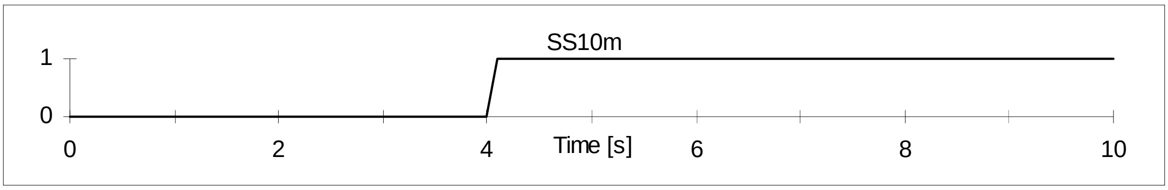
The graphs give the APRM time series calculated from the individual LPRM sensor signals feeded to the logical model and the pilot system and the responses of these.

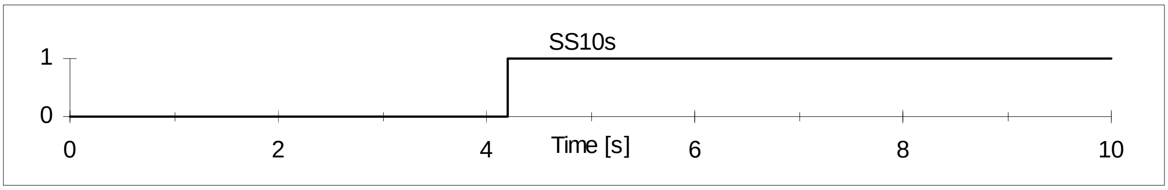
Core flow = 0.8







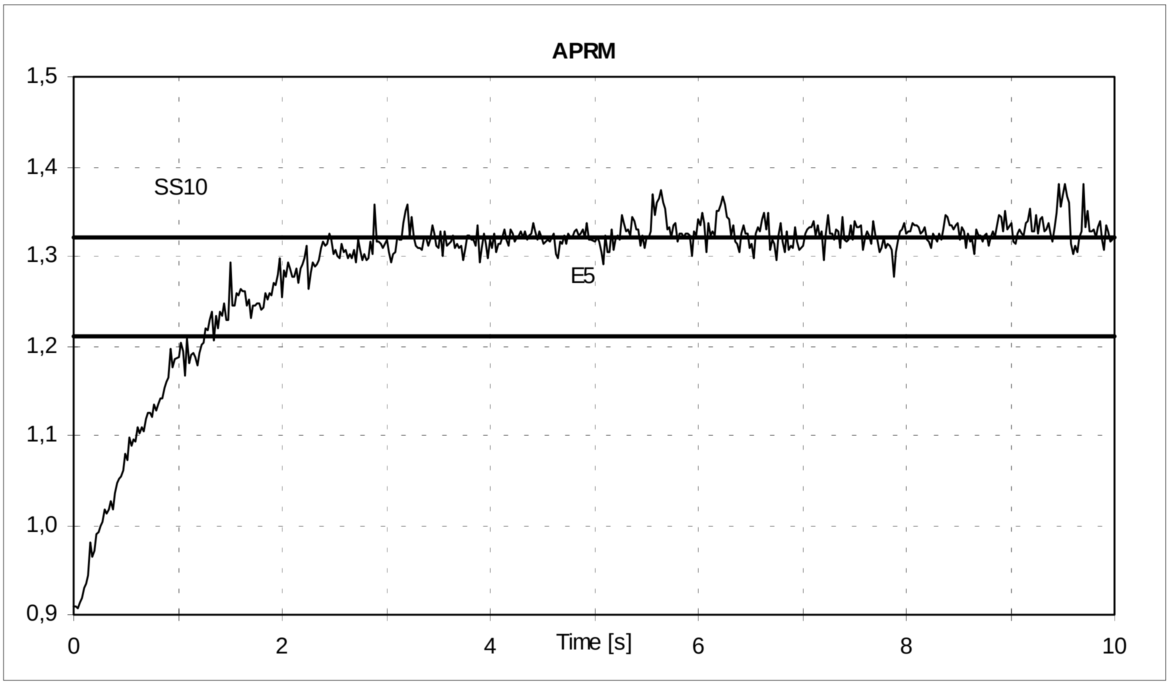


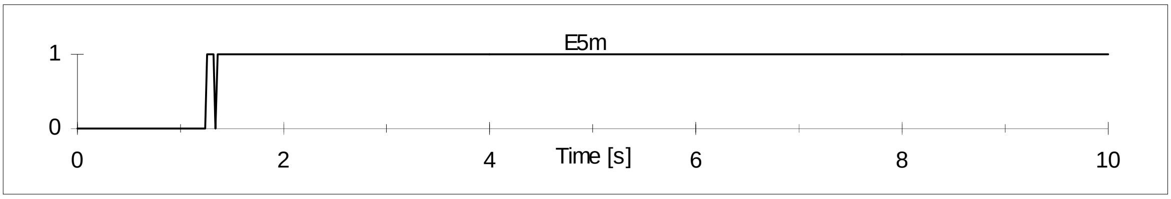


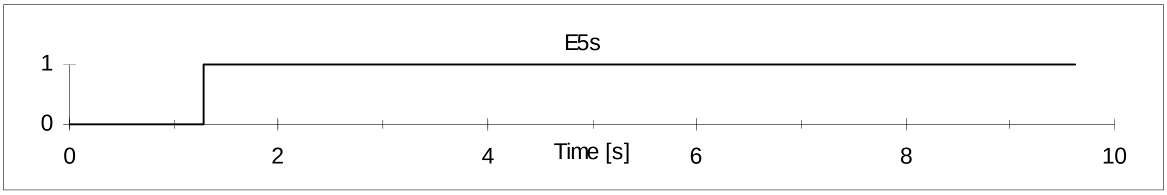
- m refers to the logical **m**odel
- s refers to the pilot **s**ystem

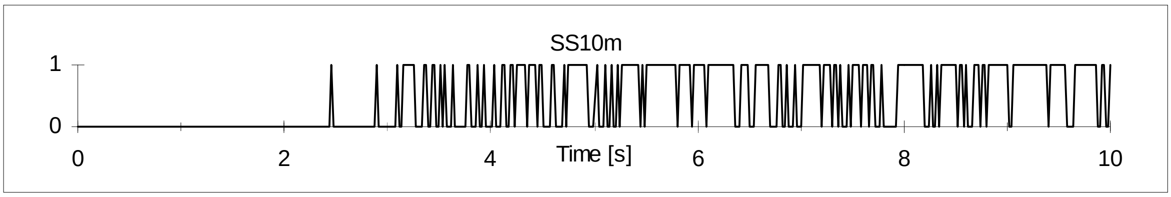
# APPENDIX B

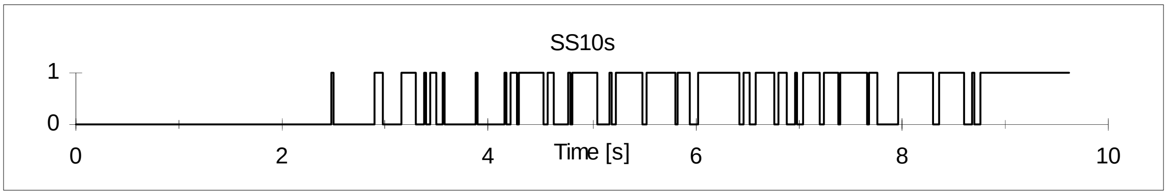
Core flow = 0.8





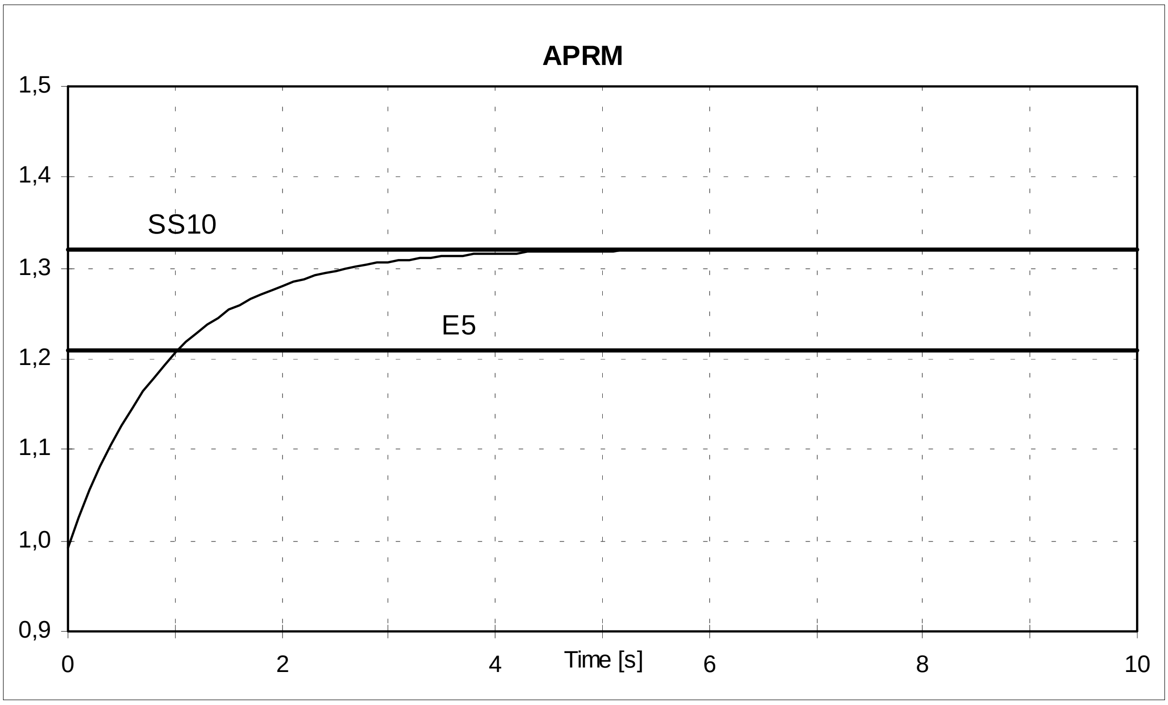


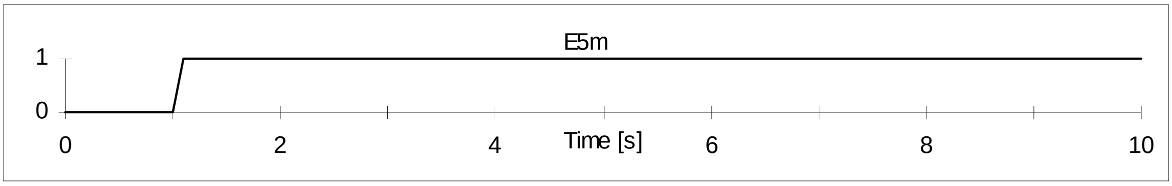


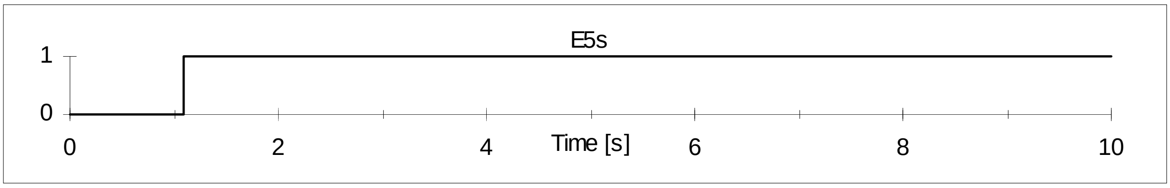


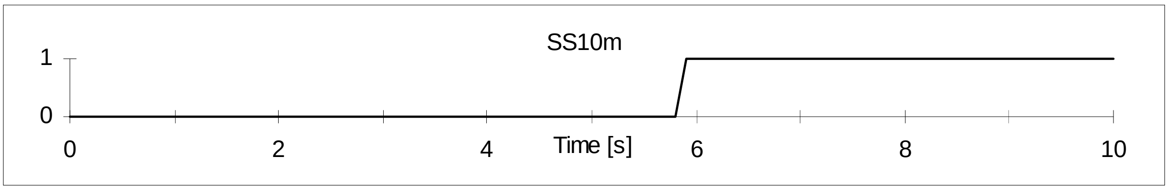
The graphs give the APRM time series calculated from the individual LPRM sensor signals feeded to the logical model and the pilot system and the responses of these.

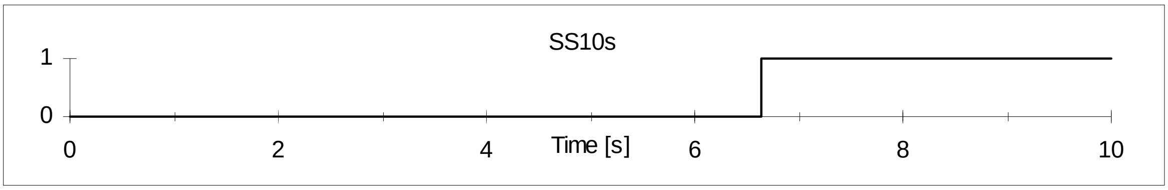
Core flow = 0.9





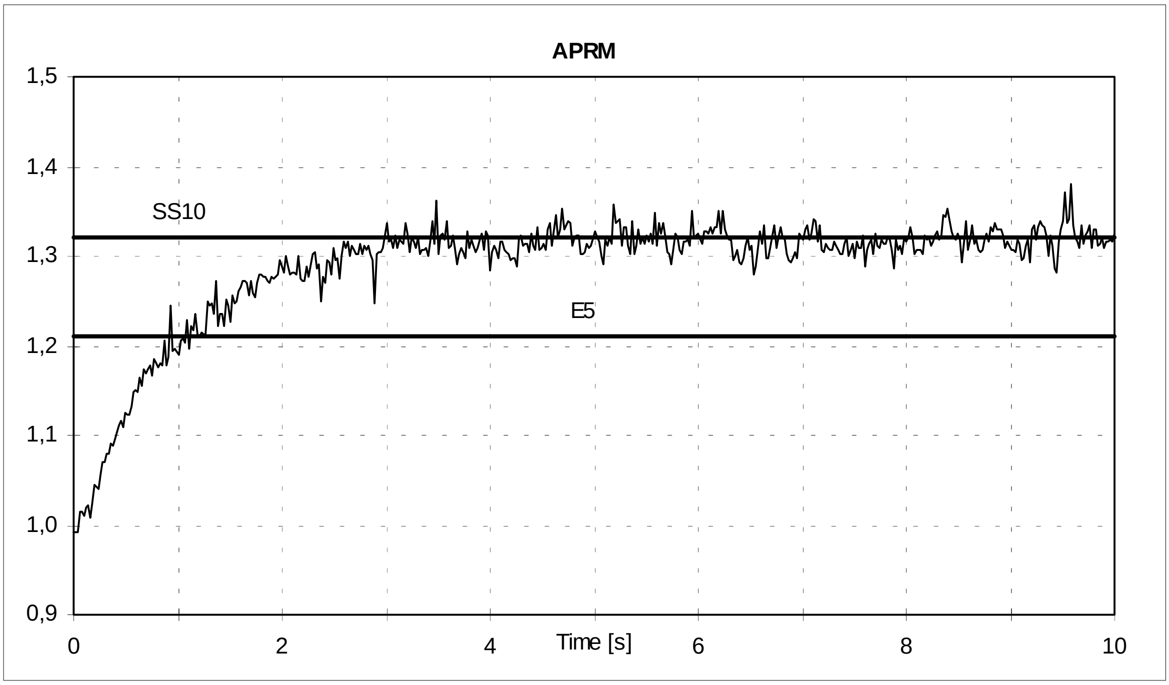


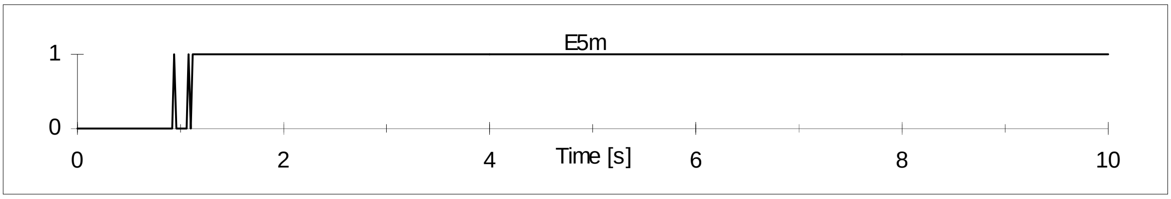


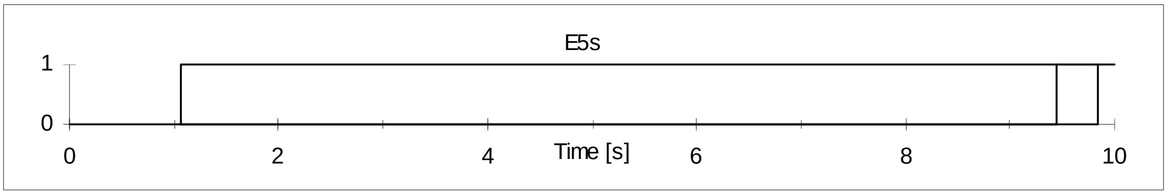


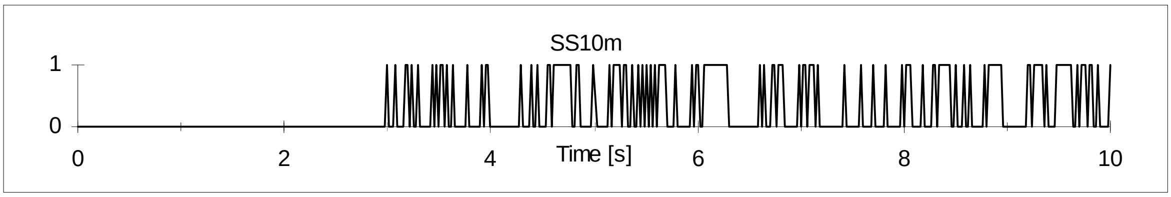
• m refers to the logical **m**odel • s refers to the pilot **s**ystem

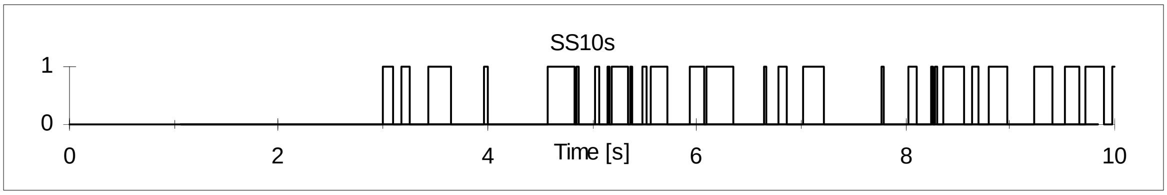
Core flow = 0.9





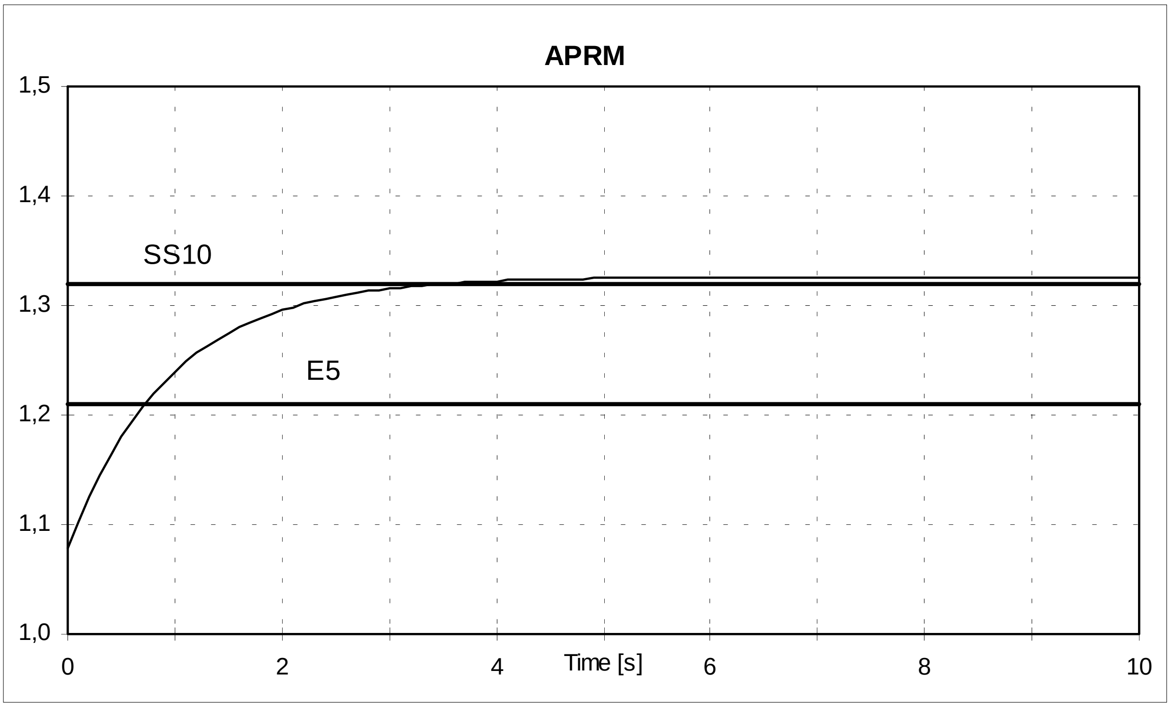


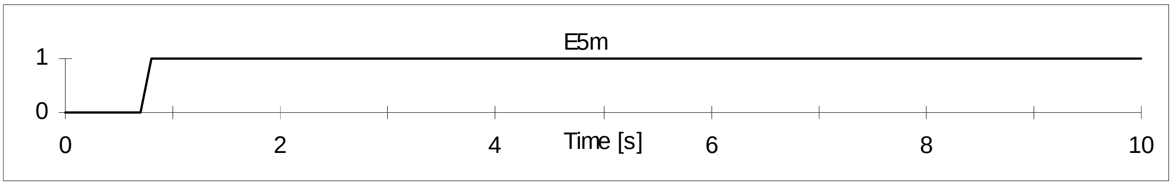


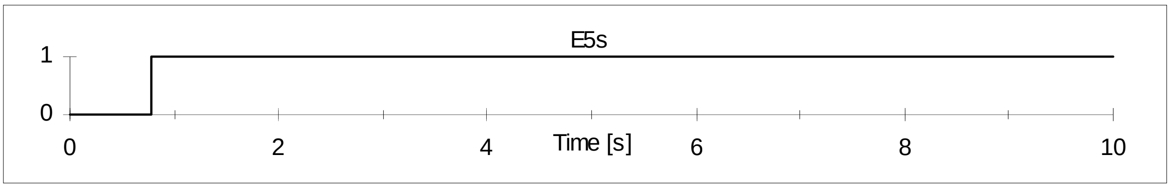


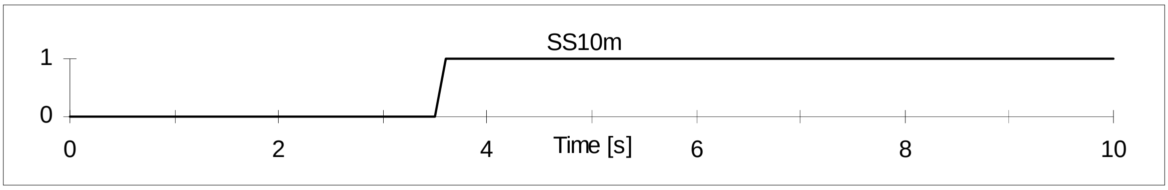
The graphs give the APRM time series calculated from the individual LPRM sensor signals feeded to the logical model and the pilot system and the responses of these.

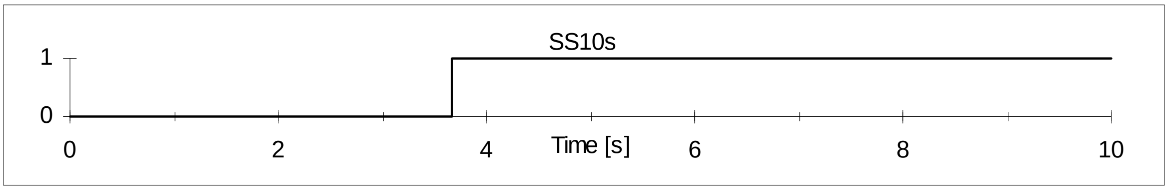
Core flow = 1.0





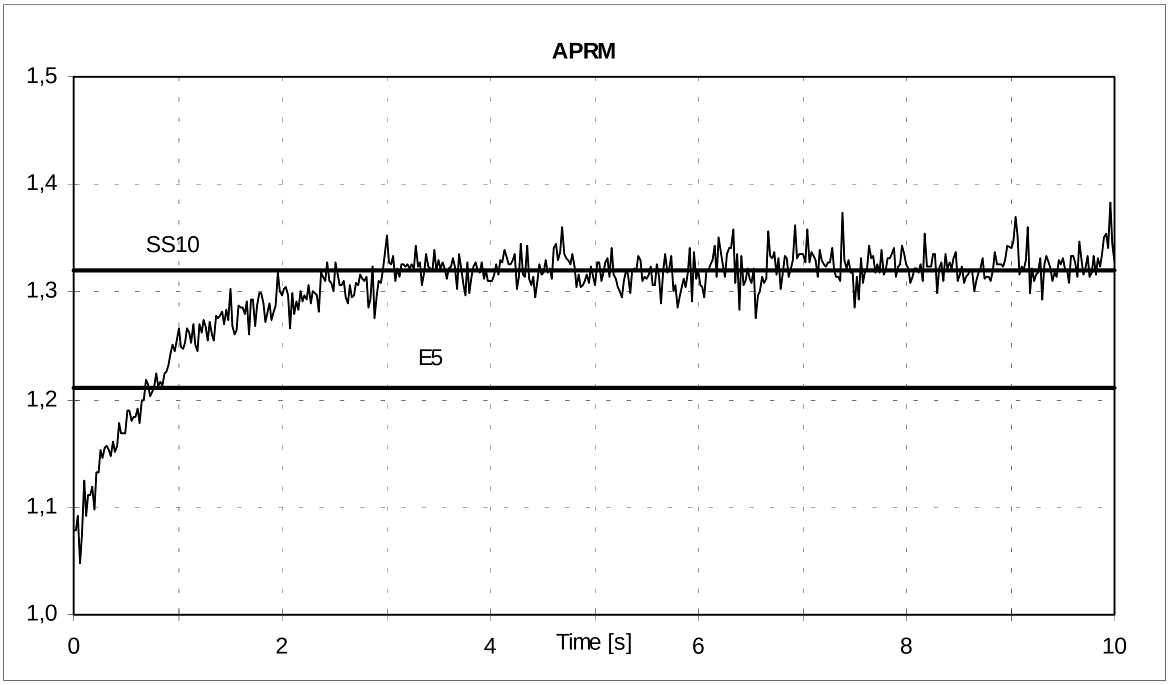


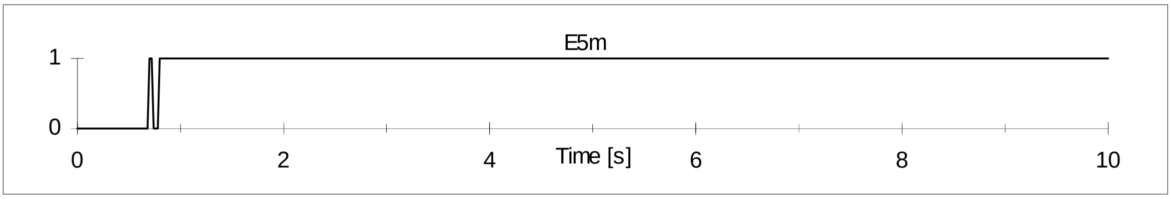


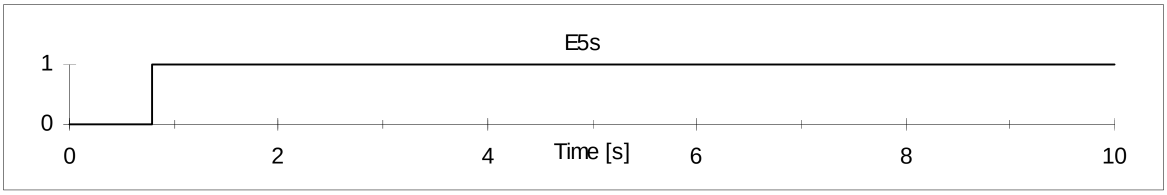


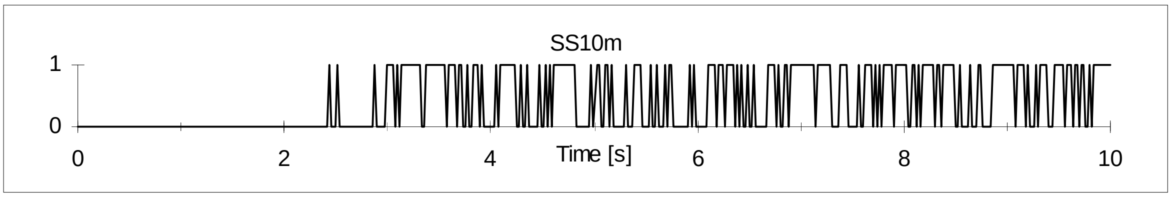
- m refers to the logical **m**odel
- s refers to the pilot **s**ystem

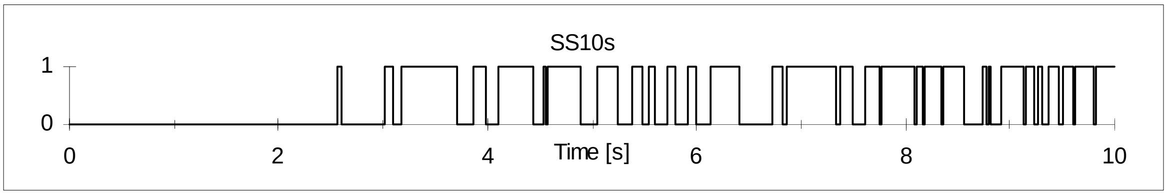
Core flow = 1.0





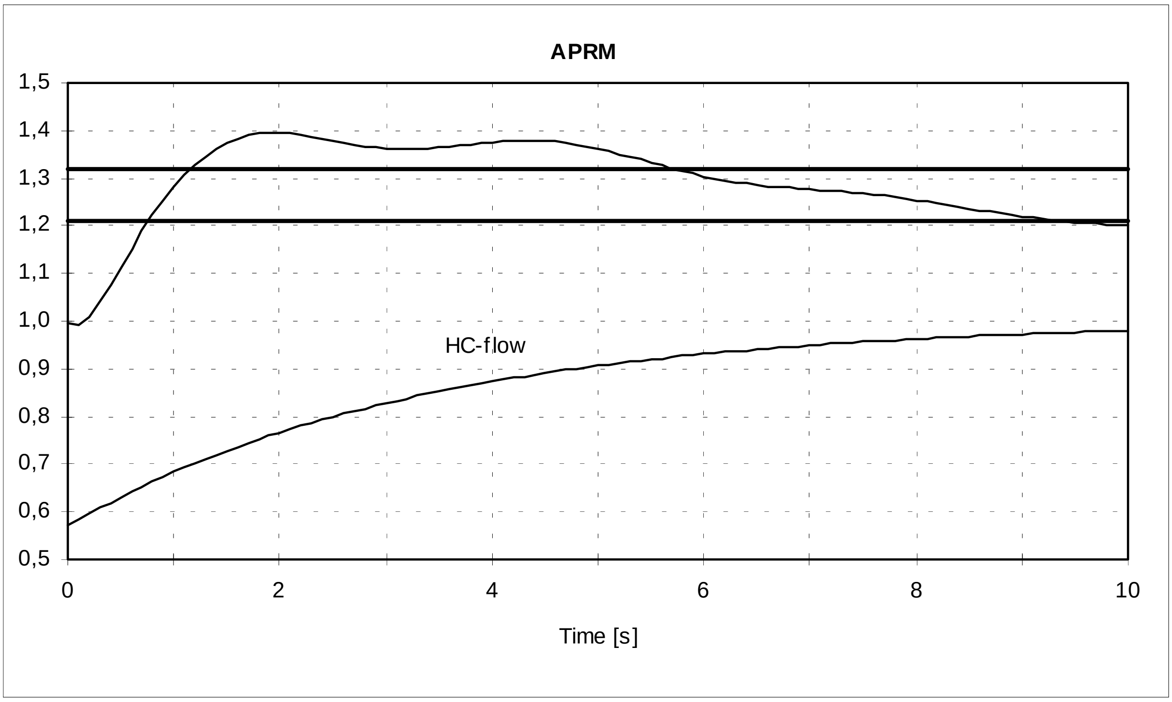


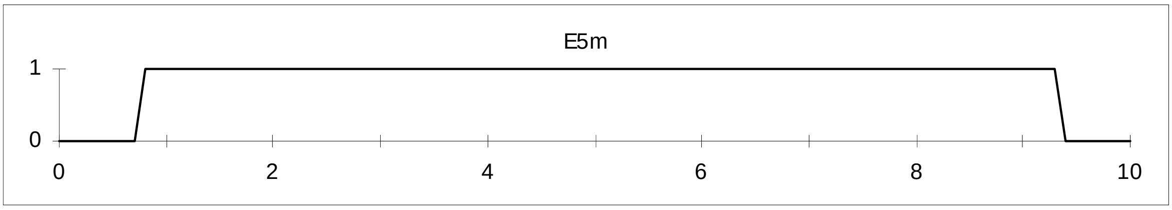


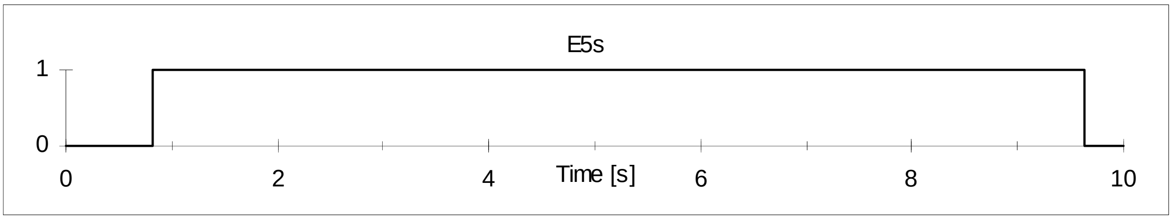


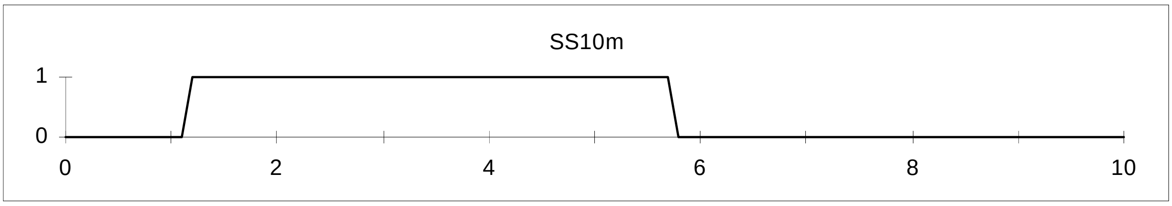
The graphs give the APRM time series calculated from the individual LPRM sensor signals feeded to the logical model and the pilot system and the responses of these.

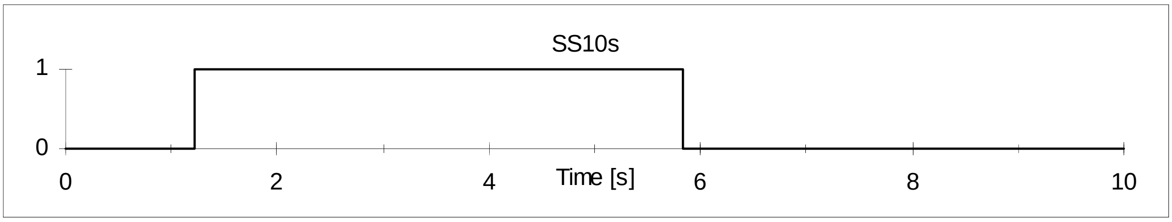
Flow transient





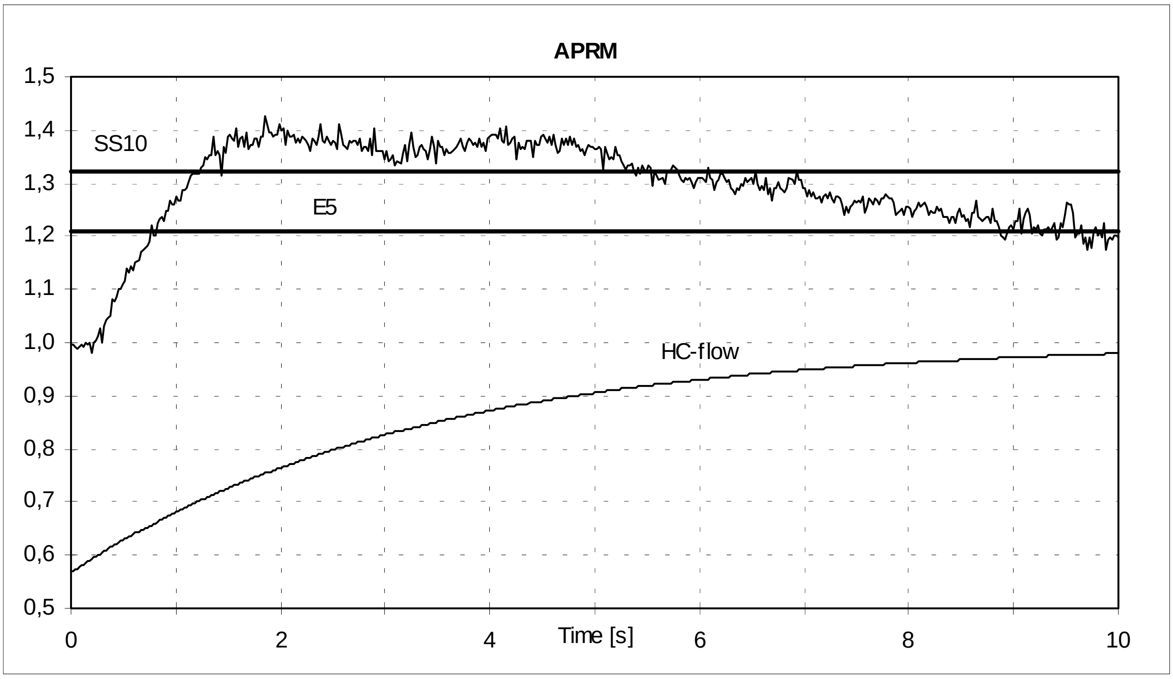


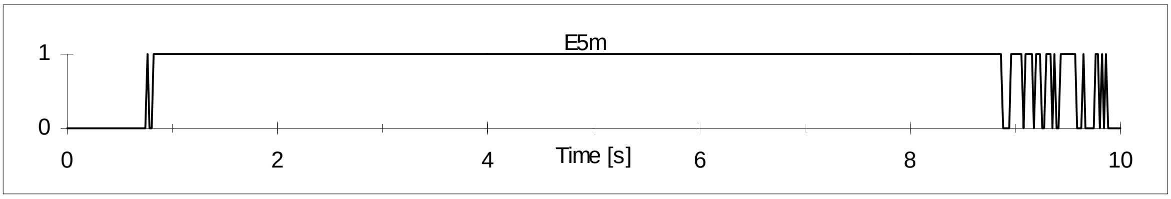


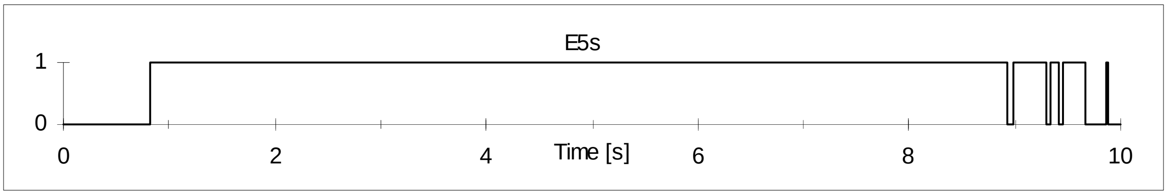


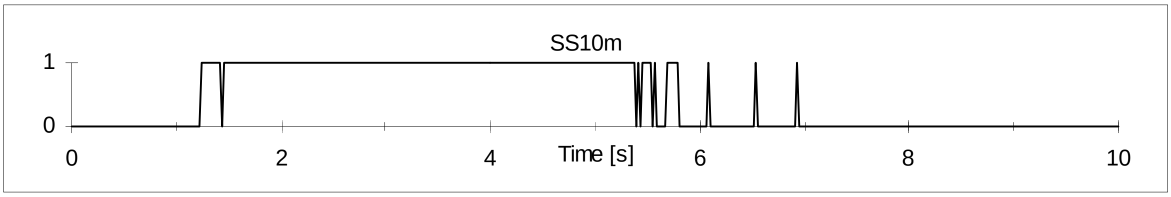
- m refers to the logical **m**odel
- s refers to the pilot **s**ystem

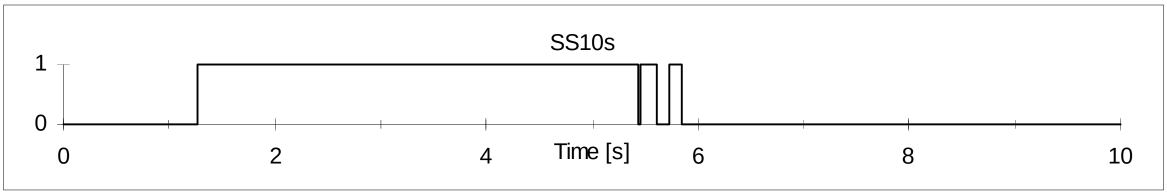
Flow transient











The graphs give the APRM time series calculated from the individual LPRM sensor signals feeded to the logical model and the pilot system and the responses of these.