BREATHING LOOP AND VALVE CONTROL MODEL
in the
DEEP LIFE REBREATHER VIRTUAL TEST BENCH

DOCUMENT NUMBER:	BL&VCM_report051023.doc
[Filename]	Folder: ..\Re-Br\PPO2_control\Valve_SC_051023
ORIGINATOR:	Alex Deas, Bob Davidov
DEPARTMENT:	**Engineering**
DATE ORIGINATED:	**30th October 2005**
REVISION:	**B**

<table>
<tr><th colspan="2">APPROVALS</th></tr>
<tr><td>_____
Hardware Architect</td><td>_____
Date</td></tr>
<tr><td>_____
Software Architect</td><td>_____
_____
Date</td></tr>
<tr><td>_____
Project Manager</td><td>_____
_____
Date</td></tr>
<tr><td>VK_____
Quality Officer</td><td>_____
_____
Date</td></tr>
</table>

Controlled Document ⊠   Classified Document ☐
DO NOT COPY.

Revision History

Revision	Date	Description
A	10th O 2005	Basic RB model stripped down for publication as part of Open Revolution submission
		Revision history continued overleaf

Revision History: Continued		
B	25 October 2005	Draft with material under NDA removed. For review. Items to be clarified highlighted in Yellow.

Table of Contents

List of Figures

1 PURPOSE AND SCOPE

This document describes the rebreather model used in the Deep Life Rebreather Virtual Test Bench (VTB). The VTB is a Win X program that provides:

1. A GUI so the use can control the tool and visualise results

2. An accurate model of the rebreather and its environment, so when an actuator is fired or an event happens, sensor valves change by the same amount they would in the real rebreather.

3. The rebreather code. This is exactly the same code that is run in the base unit and handset, but simply with the hardware API driving the RVT instead of the actual hardware. Rebreather code can be any number of C++, C or HDL blocks: the tool supports co-simulation of Verilog and VHDL with either Synopsis Chronologic or Mentor Model-Sim. This is to reflect that a rebreather comprises a micro controller running in parallel with an FPGA to provide the desired system integrity.

4. Verification tools, including the ability to run batch profiles, Monte Carlo runs and pseudo random profiles to check the code behaves within acceptable limits, with GUI test tools to ensure any software change released to a user does not have any unexpected side effects.

5. An accurate model of a human, to flag when any profile creates a hazardous condition or one that is not life sustaining.

The purpose of the VTB is:

- To enable accurate identification of model parameters.

- For test and optimisation of control algorithms.

- Testing of the controller code used on the actual rebreather.

- Testing of the rebreather system under any fault condition.

- Presentation of the operation of the rebreather and its subsystems.

- Diver Training.

The rebreather model is an intrinsic and critical part of the model, and rebreather design verification. It is subject therefore to verification in its own right. This document covers several example profiles, to enable the reviewer to understand the results produced by full verification of the model is part of the VTB. To that end, several examples are given of different PPO2 controllers:

- Bang Bang controllers

- PCM Controllers

- Injector Led Predictive Controllers

2 INTERFACES BETWEEN THE MODEL AND TEST BENCH

The model of the valve solenoid control is a part of the code within the test bench: that is the code that is subject to the testing.

Each input signal to the rebreather controller represents the output of real sensor: within the test bench, that output is replaced by a model for the sensor that has been developed in MatLAB. The sensors, whether virtual or real, are connected to the controller via the API interface.

Within the test bench, which signals must be connected is defined via the GUI. The same interface provides connection of the MatLAB simulink to the controller outputs or to the final control elements. This flow also provides control of each actuator under the control of the rebreather control code that is under test. The overall connectivity is shown in Figure 1.
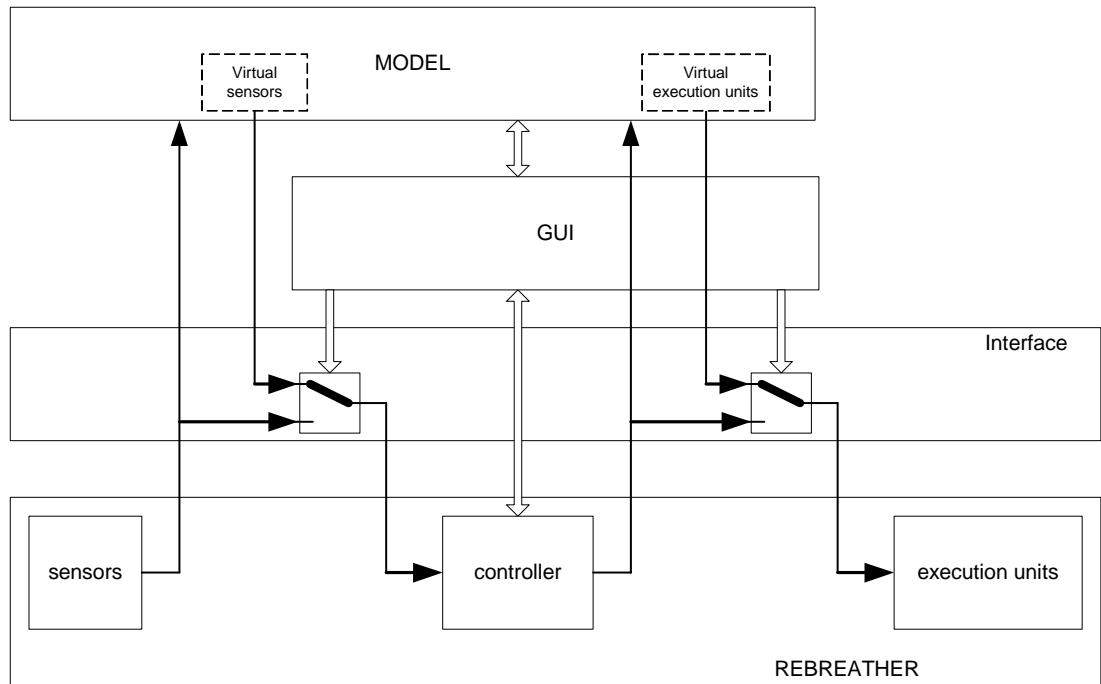
Figure 1. Interface Layers within the Rebreather Test Bench.

The main rebreather model has been developed using Simulink. It is the intention to compile the Simulink code into C, for release to the Open Revolution (OR) Initiative to support Deep Life´s OR submissions, as most lay reviewers will not have access to Simulink, or MatLAB.

The GUI passes the user defined parameters to the Simulink model and runs them. Each display in the GUI shows a state of the virtual rebreather, because the rebreather model generates the sensor readings, these are acted upon by the actual rebreather control code under test, then output to the GUI because the GUI API replaces the Hardware API.   For example, the Text LCD driver is replaced at a very low level with an emulator that puts the characters on the screen with a "skin" with the appearance of the particular rebreather system the code relates to.

The test bench provides accelerated and real-time simulation, for automated testing and for manual operation respectively.

The accelerated simulation needs to calculate the initial parameters for the real-time mode. For example, lets assume that the current depth is 3m, a user wishes to simulate the rebreather work at the depth of 90m. To get it in real-time it needs at least 180 seconds to descend to that level. To save time the system calculates the descent in the accelerate mode (user can see the plots of the parameters). Having reached the 90m the system can then start work in the real-time mode.

To support the dynamic mode, Simulink must send to GUI the current data and GUI must show them on its display in each clock. The normal display update rate is one second.

3 VALVE SOLENOID CONTROL MODEL

3.1 Overall structure

The model of the valve control system in MatLAB Simulink is shown in Figure 2. The system provides O2 and CO2 control in the breathing loop of the rebreather. The control loop includes the digital and analogue parts. The model of the digital controller can be defined in Simulink or in Modelsim. The ModelSim controller must be described in Verilog HDL or VHDL. The analogue part built in Simulink models the environment of the controller.

A Bang-Bang PPO2 controller is used to describe the model, simply because a Bang-bang controller is the simplest (crudest?). The Bang-bang controller is used to check the model and to obtain the base characteristics for comparison with the oher control algorithms. The main advantage of the Bang-bang controller here, is that it is completely predictable for a user, and produces errors the magnitude of which can be checked easily against the real rebreather to ensure that all other parameters used to control the model have been set up correctly.

3.2 Simulink PPO2 Loop Model

The Simulink model of the PPO2 control loop is shown in Figure 2, and includes the following subsystems:
1. Depth Data S-Function
2. ADC Model for checking the sampling system accurately.
3. Controller: the actual rebreather code or a Bang-bang controller
4. O2 Valve Model (solenoid, needle valve, pinch valve or variable orifice valve)
5. Human Model
6. Breathing Loop Model
7. PPO2 Sensor Model
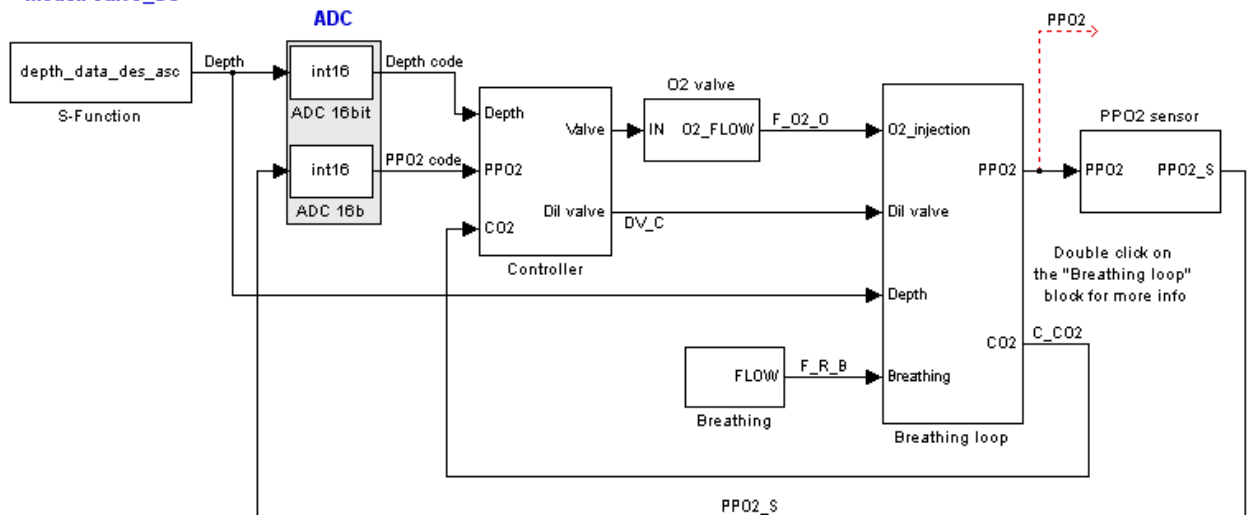
Figure 2. Simulink model: PPO2 Control Loop

NOTE: To switch off the CO2 control, define the life-time of the absorber to be more than simulation run time, or set the DV_C line between the "controller" and the "breathing loop" subsystems to 1.

3.3 Co-simulation of C code and Verilog HDL

In addition to the model of the controller, -Simulink works with the controller sections described in Verilog HDL or VHDL. To do this, it uses the co-simulation block from the Link for ModelSim Library shown in Figure 3, and a licensed copy of Mentor Graphics ModelSim or Synopsis Chronologic to be available. The Simulink model of the combined system acts as a server that runs the ModelSim model and exchanges data for each sample period.
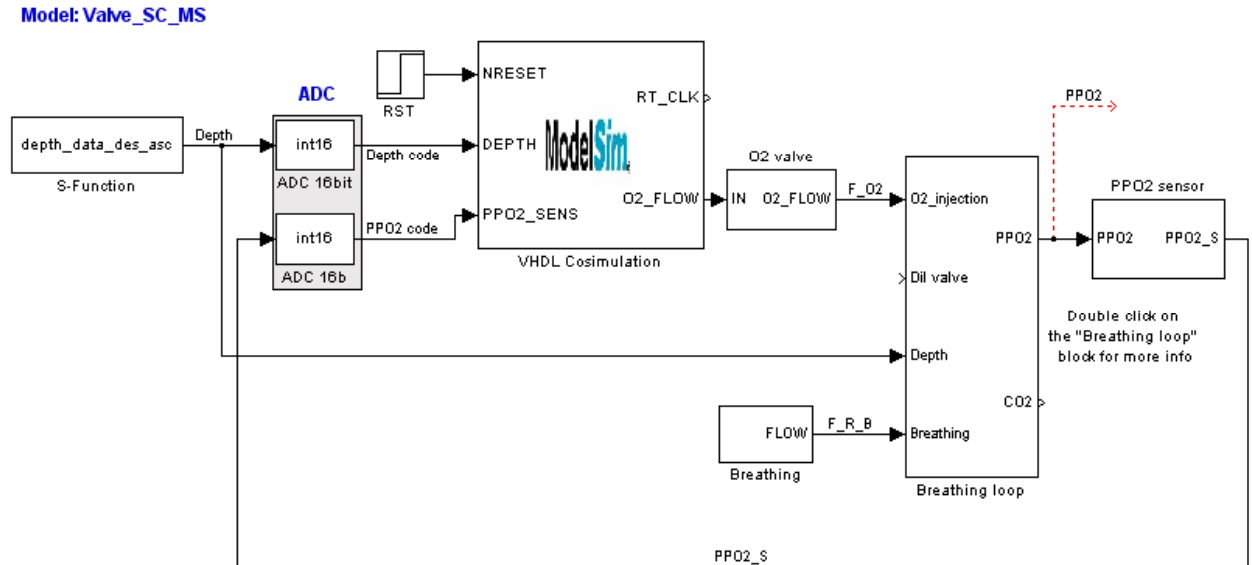
Figure 3. PPO2 Control Loop showing HDL co-simulation

3.4 Specifying Dive Profiles

Depth changes are one of the parameters that drive the model. Depth can be defined by means of a column in a table forming a dive profile, or from a C code breakout to a user function. An example of a depth profile extracted from a dive profile table is show in Figure 4.
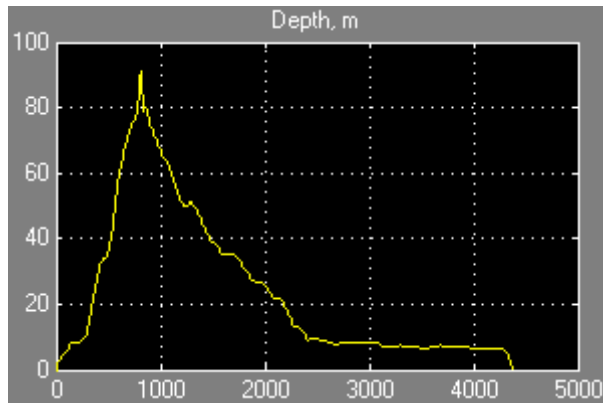
Figure 4. Dive profile to a depth of 91m for 810s

The MatLAB script in the example "depth_data_91_810.m" uses an M-file S-function providing depth table data and generate depth according to the current simulation time. The block structure of the depth simulation is shown in Figure 5.
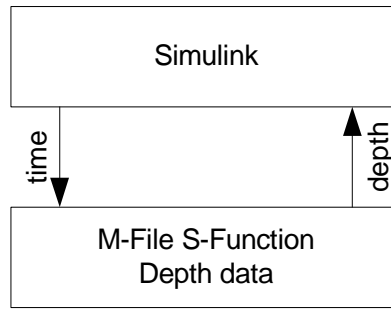
Figure 5. Depth profile data generator using an M-file S-function

The depth data passes through ADC before reaching the controller input. The results of simulation depends on the ADC resolution. The data from the file is plotted in Figure 6 (file: depth_data_91_810.m) and shows

- a). the depth profile,
- b). zoom on the data crossing 10 m depth during descent,
- c). zoom on the bottom of the depth profile,
- d). zoom on crossing 10 m depth during ascent.

a)

b)

c)

d)
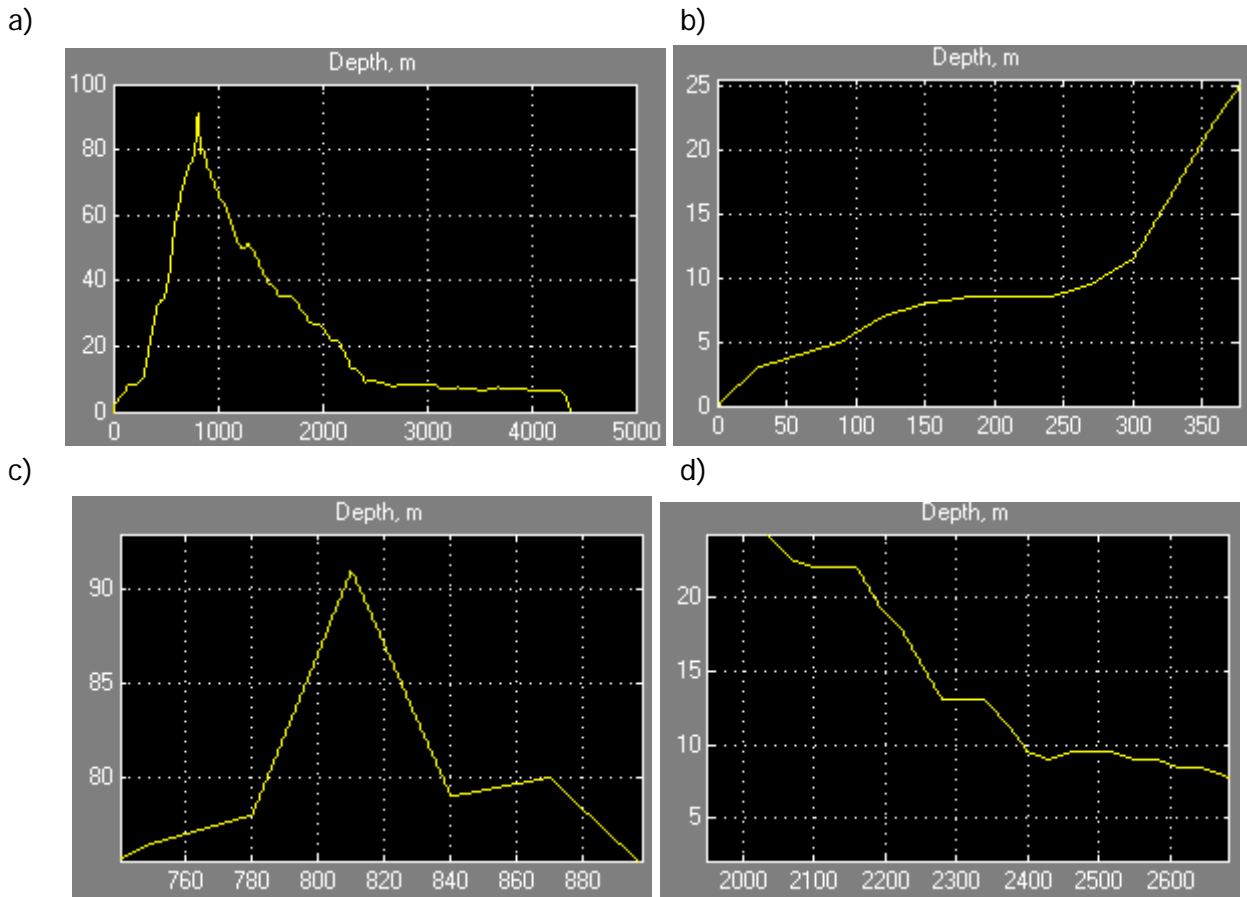
Figure 6. Depth data of 91m/810s: a) the profile for 4380 seconds; b) at the beginning of the dive; c) at the beginning of the ascent d) crossing 10 m during ascent

The faster descent to 91 m depth (in 180 seconds) excluding local ascent during descent and the local descent while ascent is shown in Figure 7. This profile is in the file: depth_data_91_180.m.

a)   b)
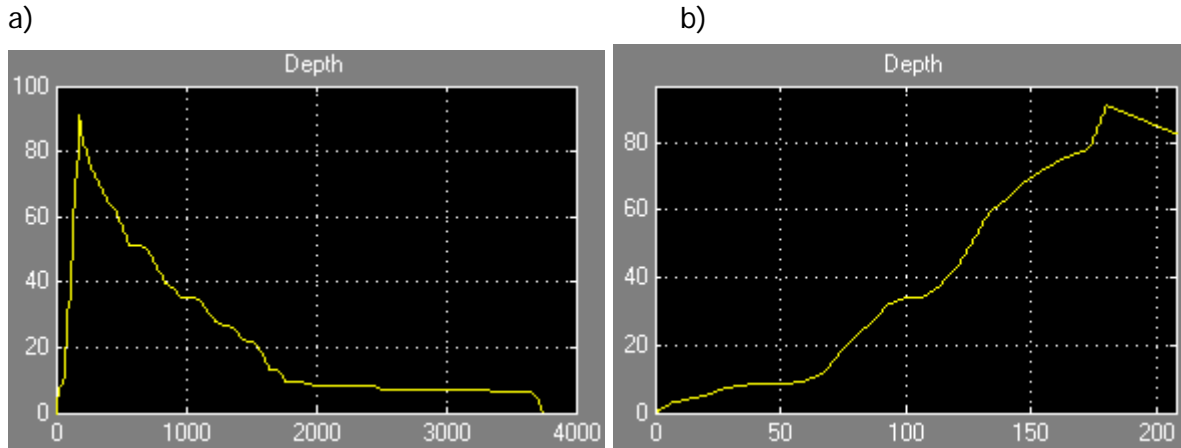
Figure 7. Depth data of 91m/180s: a) the profile for 3750 seconds; b) descent in 180 s

The MatLAB simulink S-Function of the depth_data_91_180.m file follows.

```matlab
%-----------------------------------------------------------------------------
%   Design unit : VALVE_CONTROL
%   Filename: depth_data_91_180.m
%
%   Description: Diving table data for the breathing loop simulink model.
%                Profile of the depth.
%
%    Limitations:
%
%   Author: Bob Davidov
%           Deep Life Ltd.
%
%   Simulator: MatLAB 7.01 R14 SP1
%
%   Version 1.1 by 20 October 2005, original release
%-----------------------------------------------------------------------------


%
% profile of the depth


function [sys,x0,str,ts]=depth_data_91_180(t,x,u,flag)
%LIMINTM Limited integrator implementation.
%   M-file S-function implementing a continuous changing of the depth
%
%   See sfuntmpl.m for a general S-function template.
%   Copyright 1990-2002 The MathWorks, Inc.
%   $Revision: 1.17 $

switch flag

  %%%%%%%%%%%%%%%%%%
  % Initialization %
  %%%%%%%%%%%%%%%%%%
  case 0
    [sys,x0,str,ts] = mdlInitializeSizes;

    param.version = '1.0 Beta A';                    % For clarity, confirm RB_MC
version
    disp(['Running main. RB_MC version ' param.version]);

  %%%%%%%%%%%%%%%
```

```matlab
  % Derivatives %
  %%%%%%%%%%%%%%%
  case 1
    sys = mdlDerivatives(t,x,u);

  %%%%%%%%%%%%%%%%%%%%%%%%%
  % Update and Terminate %
  %%%%%%%%%%%%%%%%%%%%%%%%%
  case {2,9}
    sys = []; % do nothing

  %%%%%%%%%%
  % Output %
  %%%%%%%%%%
  case 3
    sys = mdlOutputs(t,x,u);

  otherwise
    error(['unhandled flag = ',num2str(flag)]);
end

% end limintm


%
%===============================================================================
% mdlInitializeSizes
% Return the sizes, initial conditions, and sample times for the S-function.
%===============================================================================
%
function [sys,x0,str,ts] = mdlInitializeSizes(lb,ub,xi)

sizes = simsizes;
sizes.NumContStates  = 0;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 1;
sizes.NumInputs      = 0;
sizes.DirFeedthrough = 0;
sizes.NumSampleTimes = 1;

sys = simsizes(sizes);
str = [];
x0  = [];
ts  = [0 0];   % sample time: [period, offset]

% end mdlInitializeSizes


%
%===============================================================================
% mdlDerivatives
% Compute derivatives for continuous states.
%===============================================================================
%
function sys = mdlDerivatives(t,x,u)

sys = [];

% end mdlDerivatives


%
```

```matlab
%=========================================================================
% mdlOutputs
% Return the output vector for the S-function
%=========================================================================
%
function sys = mdlOutputs(t,x,u)

data_in = [0    0;
7    3;
13   4;
20   5;
27   7;
33   8;
40   8.5;
47   8.5;
53   8.5;
60   9.5;     % 10m at 61.75s
67   11.5;
73   17;
80   22.5;
87   27;
93   32;
100 34;
107 34;
113 37;
120 42.5;
127 51;
133 59;
140 63;
147 68;
153 71;
160 74;
167 76.5;
173 78;
180 91;      % the depth is 91m at 180sec.
210 82;
240 80;
270 75;
300 72.5;
330 70.5;
360 67.7;
390 65.5;
420 64;
450 62.5;
480 60;
510 57;
540 54;
570 51;
600 51;
630 51;
660 51;
690 50.5;
720 49;
750 46.7;
780 44;
810 42.5;
840 40;
870 39;
900 38.5;
930 37;
960 35.5;
```

```
990 35.5;
1020    35;
1050    35;
1080    35;
1110    34.5;
1140    32.5;
1170    31;
1200    30;
1230    28;
1260    27.5;
1290    27;
1320    26.5;
1350    26.5;
1380    25.5;
1410    24;
1440    22.5;
1470    22;
1500    22;
1530    22;
1560    19.5;
1590    18;
1620    15.5;
1650    13;
1680    13;
1710    13;
1740    11.5;
1770    9.5;
1800    9.5;
1830    9.5;
1860    9.5;
1890    9.5;
1920    9;
1950    9;
1980    8.5;
2010    8.5;
2040    8;
2070    8;
2100    8;
2130    8;
2160    8;
2190    8;
2220    8;
2250    8;
2280    8;
2310    8;
2340    8;
2370    8;
2400    8;
2430    8;
2460    8;
2490    7.5;
2520    7;
2550    7;
2580    7;
2610    7;
2640    7;
2670    7;
2700    7;
2730    7;
2760    7;
2790    7;
```

```
2820    7;
2850    7;
2880    7;
2910    7;
2940    7;
2970    7;
3000    7;
3030    7;
3060    7;
3090    7;
3120    7;
3150    7;
3180    7;
3210    7;
3240    7;
3270    7;
3300    7;
3330    7;
3360    6.5;
3390    6.5;
3420    6.5;
3450    6.5;
3480    6.5;
3510    6.5;
3540    6.5;
3570    6.5;
3600    6.5;
3630    6.5;
3660    6.5;
3690    5.7;
3720    3;
3750    0];

%data_in(:,1)=data_in(:,1)./100;


sys = interp1q(data_in(:,1), data_in(:,2), t);
%t


% end mdlOutputs
```

3.5 The Human Model

The diver is modelled a breathing unit that generates flow from a specific respiratory rate and tidal volume, consumes O2, generates CO2. The metabolic rate, respiratory rate and tidal volume are top line parameters set by the user: these are literally on the top line of the GUI.

For the examples in this document, the metabolic rate (F_{R_B}) was set to 1.5 l/min of O2 or 0.025 l/s. The respiratory rate is 60/2.1: this is a high rate used to test the mode, and tidal volume is 3 litres. The on-off time ratio is 0.33/0.66. The resulting breathing profile shown in Figure 8 is not smoothed to apply the worst case pulsation to the PPO2 control loop. This volume used for testing the model is about the worst case a person could generate: it is a hyperventilation condition.
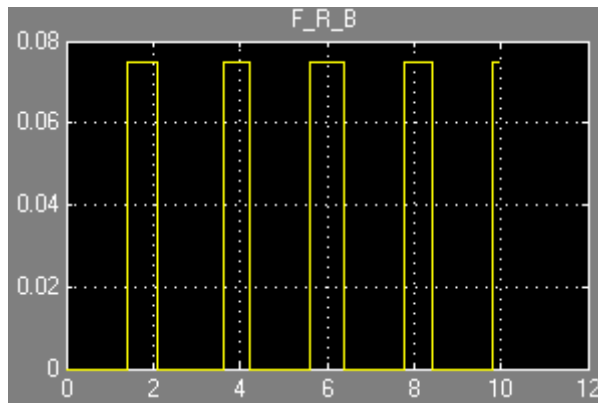
Figure 8. Flow from breathing using the example parameters

3.6 Breathing loop model

The simulation of the breathing loop models the gas flows from the oxygen and diluent cylinders through the breathing loop to the ambient environment, including O2-CO2 transformation and CO2 absorption. The model is used to

- build the rebreather model;
- test and optimise the O2 injector control algorithm;
- calculate PPO2 and other gas parameters;
- provide diving when the PPO2 sensor is out;
- provide diving control without scrubber;
- test and optimise the rebreather;
- provide work of rebreather in atmospheric condition close to deep diving condition, using the simulated data, for example, the above depth profiles.

The breathing loop includes the hose, counterlungs and scrubber shown in Figure 9.

Breathing changes the O2 level in the loop. The gas from the oxygen cylinder is brought into the loop by the PPO2 controller, or when the model is being tested, by a Bang-bang controller.

The diluent gas is injected into the breathing loop either manually or automatically by an ADV. The ADV fires to increase the internal pressure when the internal pressure is less than the difference between the ambient pressure and the diluent valve operating pressure (which can be set by the user). The gas is exhausted from the breathing loop through the exhaust valve when the internal pressure is more than the sum of the ambient and the exhaust valve operating pressures: this occurs on ascent.
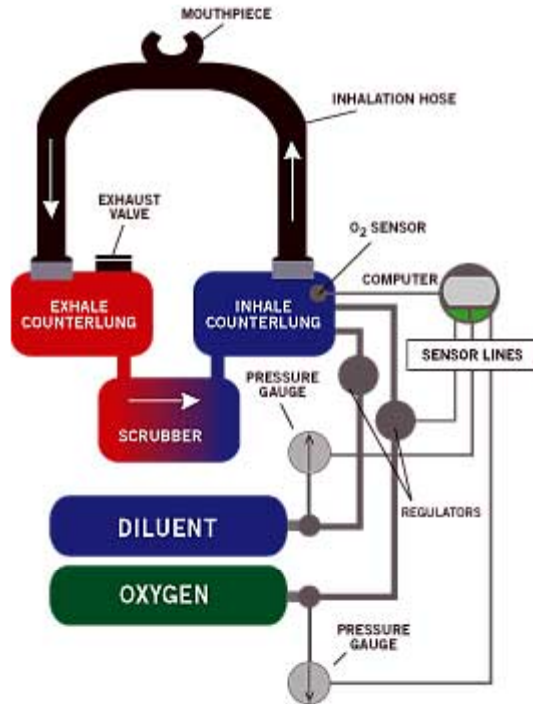
Figure 9. Mains structure in the breathing loop model

The lungs – breathing loop interaction is shown in Figure 10. The operation will be explained with the aid of some example parameter settings. When O2 consumption in the breathing is 0.05 l/s the diver converts a portion of the O2 to CO2 and this is absorbed by the scrubber, which has the effect of decreasing the gas volume in 0.1 litres each 2s breathing period. The decrease in PPO2 depends on the rebreather internal space. If the space is 4 litres the decrease of PPO2 is about 12% when the setpoint is 0.209; 3.6% for 0.7 and 2.1% when the setpoint is 1.2.
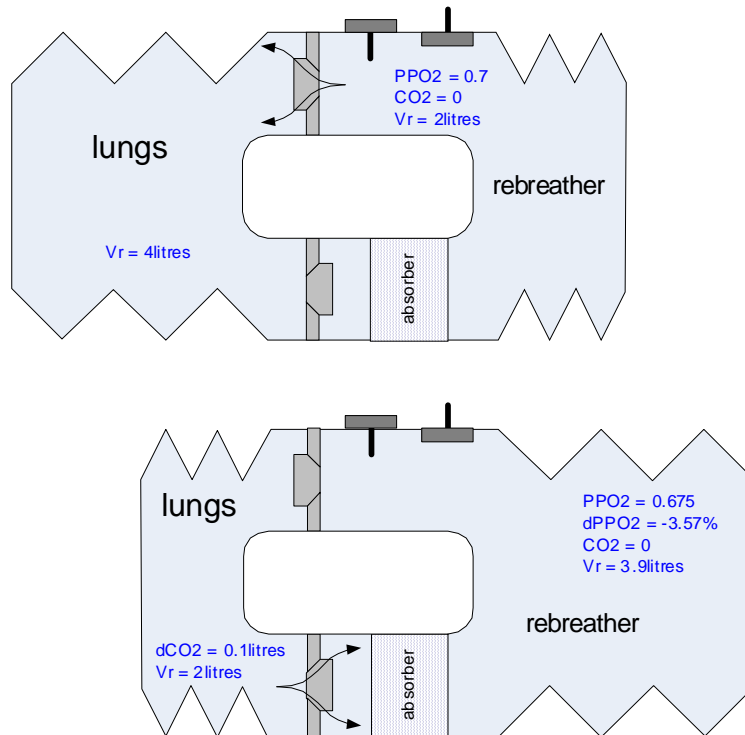
Figure 10. Lungs – breathing loop interaction.

The breathing loop model calculates the current gas volumes being inside the breathing loop as differences between the injected and exhaust gases. Respiration transforms O2 in CO2. The absorbent transforms O2 into a liquid phase and so decreases the gas volume. When the absorbent is exhausted, an alarm mode, the CO2 from breathing is not transformed into the liquid and remains in the breathing loop so adding O2 will increase breathing loop pressure relative to ambient: this causes some gas to be exhausted into the environment. Calculation of the gas volume that must be inside the breathing loop under the ambient pressure includes provision for the overpressure inside the breathing loop.

An example of the breathing loop parameters is shown in Figure 11.
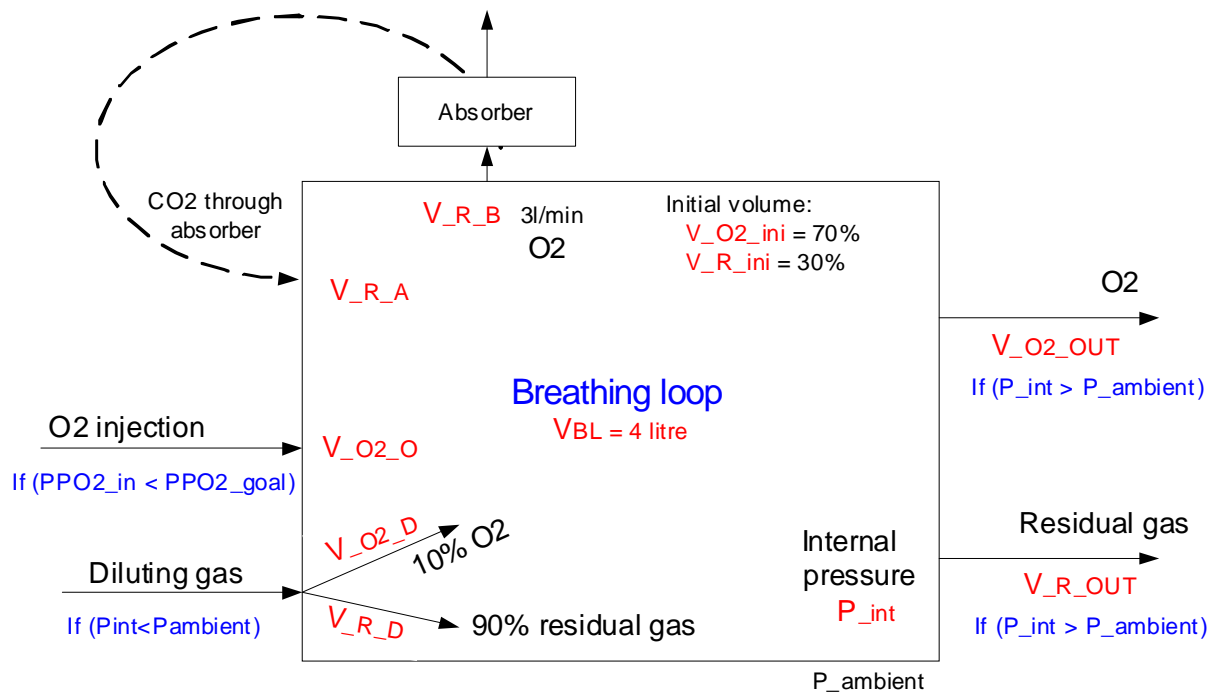
Figure 11. Example of gas flow through the breathing loop.

In the example in Figure 11, the initial O2 volume in breathing loop V_{O2_ini} = 0.7 x 4 litres

The initial residual gas in the breathing loop is V_{R_ini} = 0.3 x 4 litres

3.6.1 "Valve_SC/Breathing loop" subsystem

The Simulink model of the breathing loop is shown in Figure 12 and includes the following subsystems
- O2 balance   (Figure 13)
- CO2 balance   (Figure 16)
- Residual gas balance (Figure 17)
- Overpressure   (Figure 18)
- Diluent injection   (Figure 19)
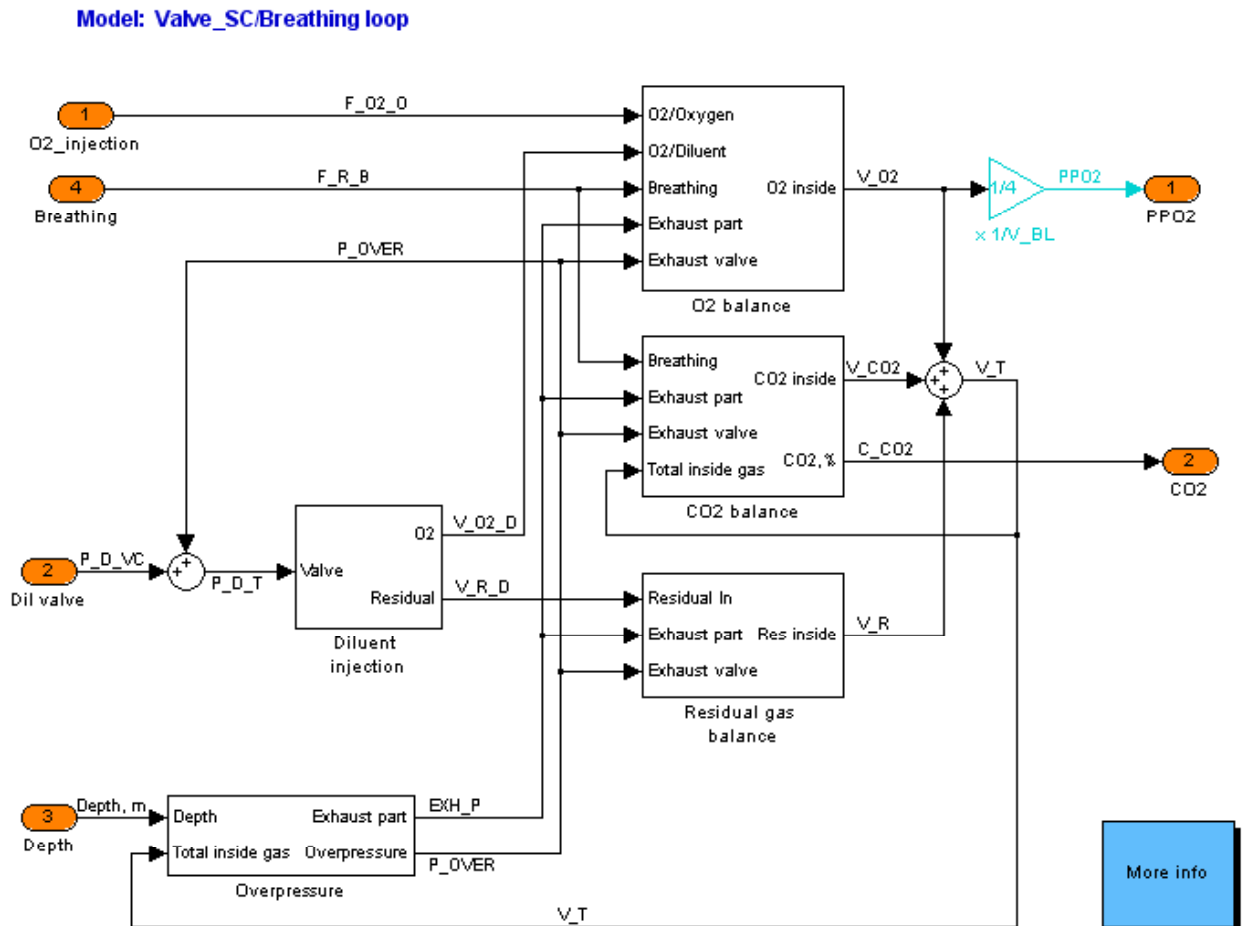
Figure 12. "Valve SC/Breathing loop" model.

- P_{D_T} Intermediate pressure to the diluent valve

- P_{D_VC} Diluent valve control pressure

- P_{OVER} Over pressure inside the breathing loop relative to the ambient pressure

- $PPO2$ O2 partial pressure inside the breathing loop

- V_{BL} ==Mean== breathing loop volume

- V_{CO2} CO2 volume inside the breathing loop

- V_{O2} O2 volume inside the breathing loop

- V_R Residual gas volume inside the breathing loop

- V_T Total gas volume inside the breathing loop

These linked by the following relationships:

$$P_{D\_T} = P_{D\_VC} + P_{OVER}$$

$$V_T = V_{O2} + V_R + V_{CO2}$$

$$PPO2 = \frac{V_{O2}}{V_{BL}}$$

3.6.2 "Valve_SC/Breathing loop/O2 balance" subsystem
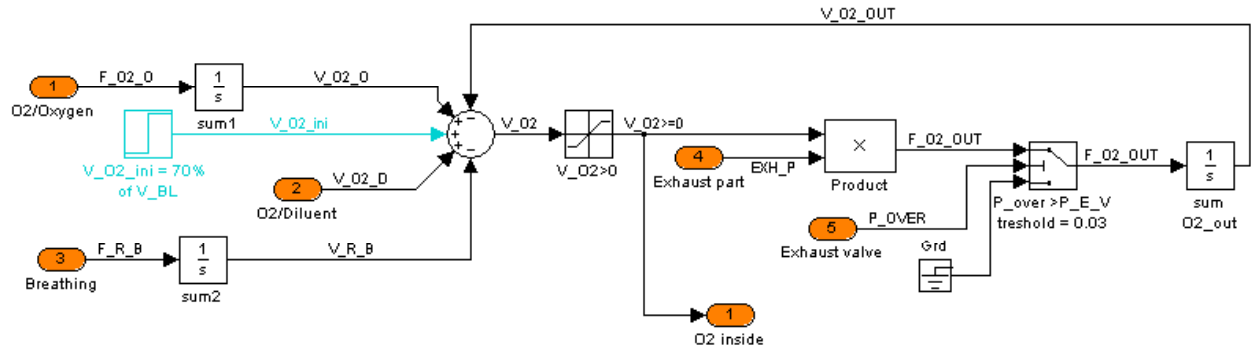
Figure 13. "Valve_SC/Breathing loop/O2 balance" model.

The parameters in the O2 balance model are as follows:

- F_{O2_O} Injected O2 flow from the oxygen cylinder

- F_{O2_OUT} Flow of O2 exhaust

- F_{R_B} Flow of O2 consumption in breathing (metabolic rate)

- P_{E_V} Operating pressure of the exhaust valve

- $PPO2_{INI}$ Initial PPO2 in the breathing loop

- P_{OVER} Over pressure inside the breathing loop relative to the ambient pressure

- V_N Gas volume inside the breathing loop under the ambient pressure

- V_{O2} O2 volume inside the breathing loop

- V_{O2_D} O2 volume from the diluent cylinder

- V_{O2_ini} Initial volume of O2 in the breathing loop

- V_{O2_O} O2 consumption from the oxygen cylinder

- V_{O2_OUT} Volume of O2 exhaust

- V_{R_B} Volume of O2 consumption in breathing

- V_T Total gas volume inside the breathing loop

The parameters are linked by the following relationships:

$$V_{O2\_O} = \int F_{O2\_O}$$

$$V_{R\_B} = \int F_{R\_B}$$

$$V_{O2\_ini} = PPO2_{INI} * V_{BR}$$

$$V_{O2} = V_{O2\_ini} + V_{O2\_O} + V_{O2\_D} - V_{R\_B} - V_{O2\_OUT}$$

$$V_{O2} \geq 0$$

$$F_{\text{O2\_OUT}} = V_{O2} * EXH_P \qquad \text{when } P_{0\text{VER}} > P_{\text{E\_V}}$$

$$F_{\text{O2\_OUT}} = 0 \quad \text{when } P_{0\text{VER}} \leq P_{\text{E\_V}}$$

$$V_{O2\_OUT} = \int F_{O2\_OUT}$$

3.6.3 "Valve_SC/Breathing loop/CO2 balance" subsystem

Example of Data used to calculate absorber parameters:

Parallel flow through two 4 inch diameter 198mm ExtendAir scrubbers, with flow rates in the ratio 5:4 rates. The absorbent volume is: $Va = 2\dfrac{\pi(2.54 \times 4)^2}{4} 19.8 = 3210$ cc

Absorbent density: $Da = 0.64$ g/cc

Absorbent performance: $Pa \cong 0.75$ lb CO2/lb LiOH

Example of calculation of the absorbent life:

Weight of absorbent (LiOH): $Ma = Da \times Va = 0.64 \times 3210 = 2054$ g

Absorbent CO2 consumption: $M_{CO2} = Pa \times Ma = 0.75 \times 2054 = 1540$ g

CO2 molar mass: 44 g/mol, where the mol is 22.4 litres

Weight of 1 litre: 1.964 g/l (approx 2g/l)

Average CO2 flow: $F_{R\_B} = 0.05$ l/s or 0.1 g/s

Time Life of absorbent: $T_A = \dfrac{M_{CO2}}{F_{R\_B}} = \dfrac{1540}{0.1} = 15400$ seconds or 4.28 hours

Three examples of absorber experimental data is shown in Figure 14, where Ac_07_06_05, C0303XXX-1 and C0303XXX-2 are different scrubber cartridges under standardised test conditions.
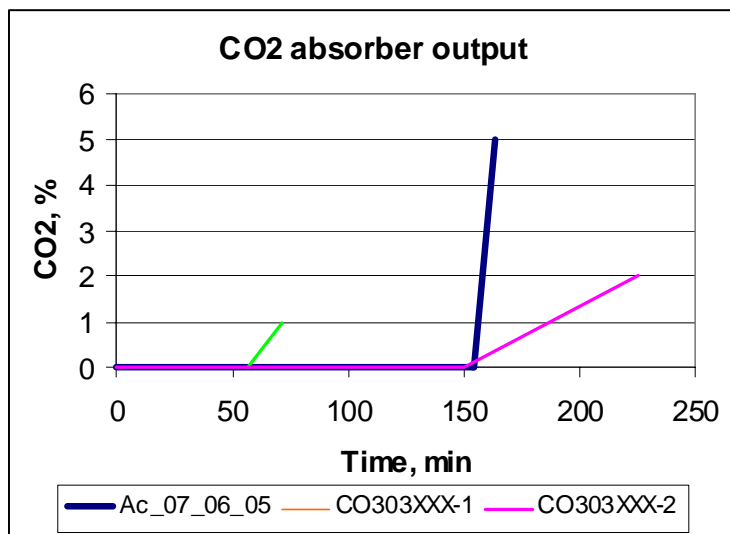
Figure 14. Absorber output CO2 level in standarised life tests

The CO2 growth of the 2x cartridge in the same space is approx. twice less than the growth of 1x cartridge as 4%/130min to 4%/60 min. The CO2 growth of Ac_07_06_05 corresponds to the worse case. The time of the CO2 growth (agony time) up to 4% of the –1 cartridge is 6min.

The rebreather model represents the scrubber life as shown in Figure 15. This is similar to actual test data. The T_A_G period can be set by the user.
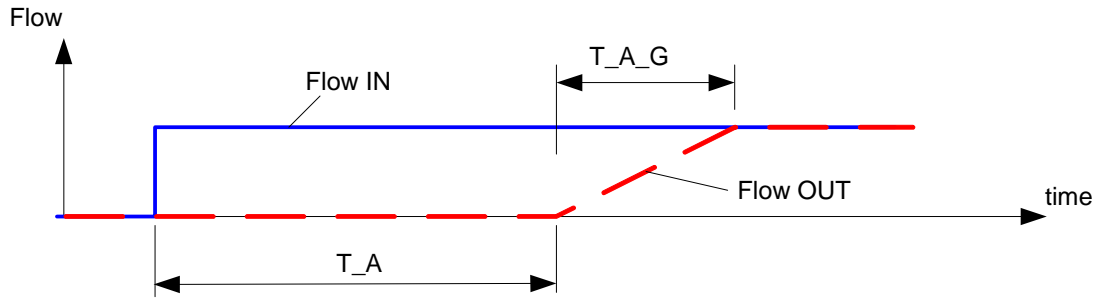
Figure 15. Absorber modeling parameters.
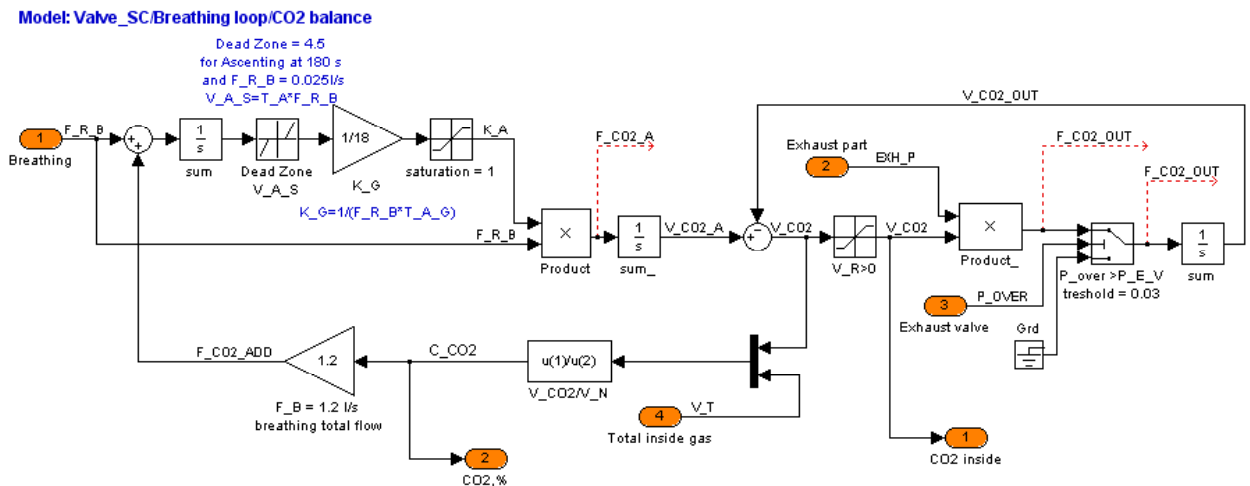
Figure 16. "Valve_SC/Breathing loop/CO2 balance" model.

The parameters used in the CO2 balance model are:

- C_{CO2} CO2 concentration in the breathing loop

- E_{CO2} Error of CO2 concentration (overlevel)

- F_B Breathing total flow

- F_{CO2_A} Part of the breathing CO2 flow passed though the absorber

- F_{CO2_ADD} CO2 flow growth in the loop proportional to the absorber CO2 output

- F_{R_B} Flow of O2 consumption in breathing (metabolic rate)

- K_A Absorbers CO2 attenuation

- K_G Absorbers CO2 output growth speed

- P_{E_V} Operating pressure of the exhaust valve
- P_{OVER} Over pressure inside the breathing loop relative to the ambient pressure
- T_A Time of the absorption when the CO2 output growth starts
- T_{A_G} Time of the CO2 growth in the absorber output (absorber agony)
- V_{A_S} Absorbed CO2 at the beginning of the CO2 output growth
- V_{CO2} CO2 volume of the breathing loop
- V_{CO2_A} Volume of the absorber CO2 output
- V_{CO2_OUT} CO2 exhaust
- V_N Gas volume inside the breathing loop under the ambient pressure
- V_T Total gas volume inside the breathing loop

These parameters are linked by the following equations:

$$V_{A\_S} = F_{R\_B} \times T_A$$

$$K_G = \frac{1}{F_{R\_B} \times T_{A\_G}}$$

$$K_A = 0, \qquad \text{when } \int F_{R\_B} < V_{A\_S}$$

$$K_A = K_G \left(F_{R\_B} + F_{CO2\_ADD} \right) \left(\int F_{R\_B} - V_{A\_S} \right), \text{ when } \int F_{R\_B} \le V_{A\_S}$$

$$\text{and } K_G \left(F_{R\_B} - F_{CO2\_ADD} \right) \left(\int F_{R\_B} - V_{A\_S} \right) < 1$$

$$K_A = 1, \qquad \text{when } K_G \left(F_{R\_B} + F_{CO2\_ADD} \right) \left(\int F_{R\_B} - V_{A\_S} \right) \ge 1$$

$$F_{CO2\_A} = K_A F_{R\_B}$$

$$V_{CO2\_A} = \int F_{CO2\_A}$$

$$V_{CO2} = V_{CO2\_A} - V_{CO2\_OUT}$$

$$V_{CO2} = \ge 0$$

$$V_{CO2\_OUT} = V_{CO2} * EXH_P \qquad \text{if } (P_{0VER} > P_{E\_V}) \qquad \text{(see Human breathing model)}$$

$$V_{CO2\_OUT} = 0 \qquad \text{if } (P_{0VER} \le P_{E\_V}) \qquad \text{(see Human breathing model)}$$

$$C_{CO2} = \frac{V_{CO2}}{V_T}$$

$$F_{CO2\_ADD} = C_{CO2} \times F_B$$

3.6.4 "Valve_SC/Breathing loop/Residual gas balance" subsystem

Model: Valve_SC/Breathing loop/Residual gas balance
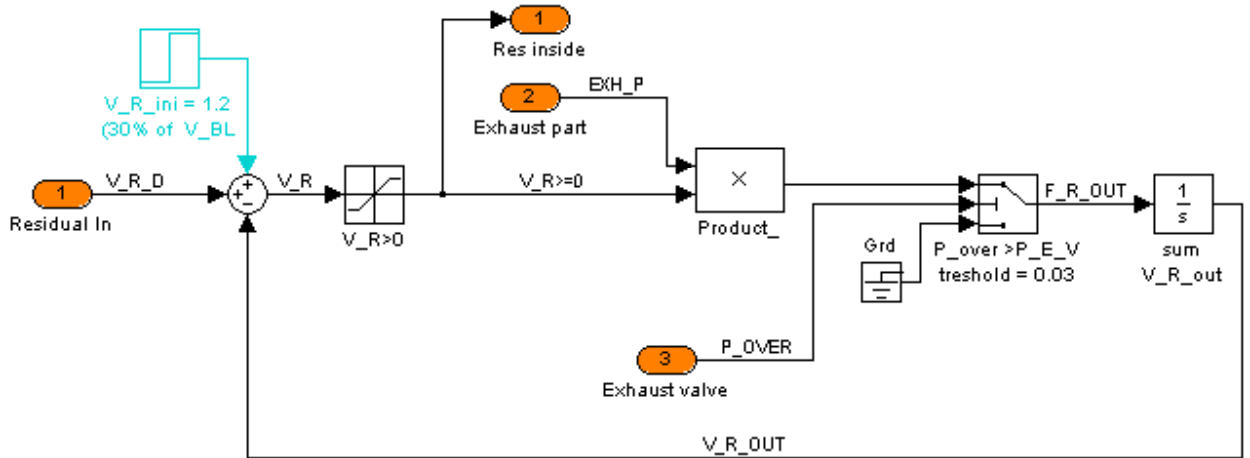
Figure 17. "Valve_SC/Breathing loop/Residual gas balance" model

The parameters used by this subsystem of the breathing loop model are:

- F_{O2_OUT} Flow of residual gas exhaust

- P_{E_V} Operating pressure of the exhaust valve

- P_{OVER} Over pressure inside the breathing loop relative to the ambient pressure

- $PPO2_{INI}$ Initial PPO2 in the breathing loop

- V_N Gas volume inside the breathing loop under the ambient pressure

- V_R Residual gas volume inside the breathing loop

- V_{R_ini} Volume of residual gas in the breathing loop *

- V_{R_D} Residual gas volume from the diluent cylinder

- V_{R_OUT} Exhaust of the residual gas

- V_T Total gas volume inside the breathing loop

The parameters are linked by the following relationships:

$$V_{R\_ini} = (1 - PPO2_{INI})V_{BR}$$

$$V_R = V_{R\_ini} + V_{R\_D} - V_{R\_OUT}$$

$$V_R \geq 0$$

$$F_{R\_OUT} = V_R * EXH_P \qquad \text{when } P_{0VER} > P_{E\_V}$$

$$F_{R\_OUT} = 0 \qquad \text{when } P_{0VER} \leq P_{E\_V}$$

$$V_{R\_OUT} = \int F_{R\_OUT}$$

3.6.4.1 "Valve_SC/Breathing loop/Overpressure" subsystem
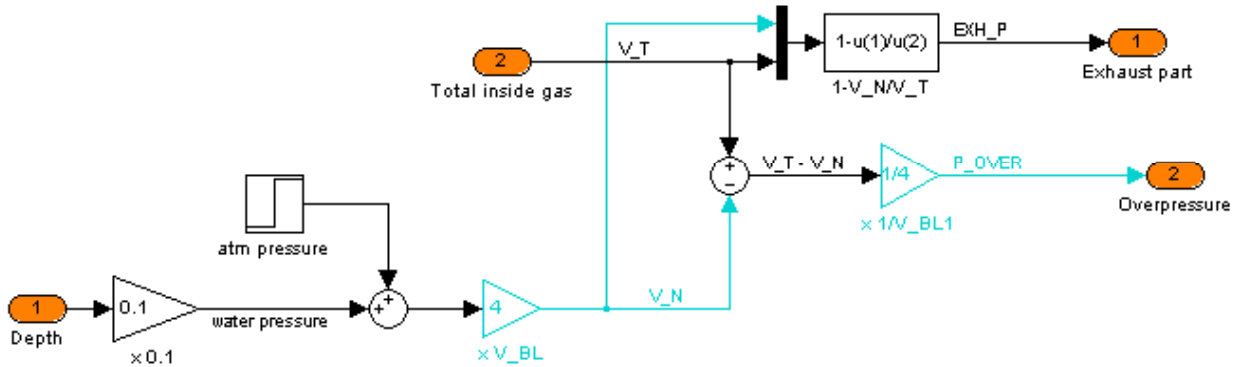
Figure 18. "Valve_SC/Breathing loop/Overpressure" model

The overpressure subsystem model uses the following parameters:
- D Depth of diving
- EXH_P Exhaust part of the total gas
- P_{int} Pressure inside the breathing loop
- XXX   Pressure at which the over-pressure valve lifts
- P_{OVER} Over pressure inside the breathing loop relative to the ambient pressure
- V_{BL} Breathing loop space
- V_N Gas volume inside the breathing loop under the ambient pressure
- V_T Total gas volume inside the breathing loop

$$V_N = V_{BL}(1 + \frac{D}{10})$$

$$EXH_P = 1 - \frac{V_N}{V_T}$$

$$P_{int} = \frac{V_T}{V_{BL}}$$

$$P_{OVER} = \frac{V_T - V_N}{V_{BL}}$$

3.6.5 "Valve_SC/Breathing loop/Diluent injection" subsystem

Model: Valve_SC/Breathing loop/Diluent injection
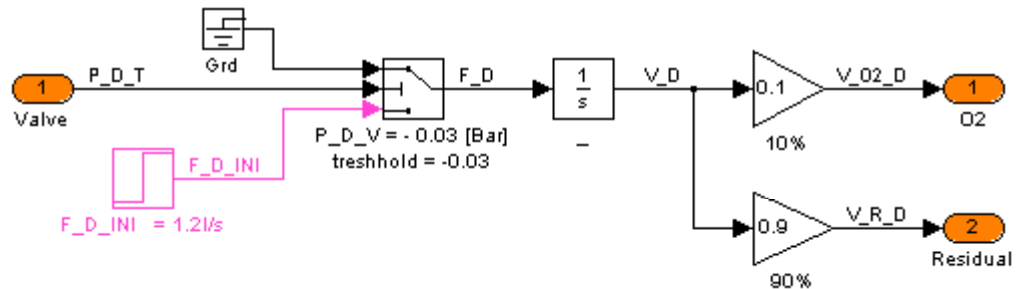
Figure 19. "Valve_SC/Breathing loop/ Diluent injection" model

The Dilutent Injection subsystem of the breathing loop model uses the following parameters:

- F_D Flow of the diluent gas
- F_{D_INI} Initial flow of the diluent when the valve is open
- P_{D_V} Operating pressure of the diluent gas valve
- P_{OVER} Over pressure inside the breathing loop relative to the ambient pressure
- V_{O2_D} O2 volume from the diluent cylinder
- V_{R_D} Residual gas volume from the diluent cylinder

These parameters are linked by the following relationships:

$$F_D = F_{D\_INI} \quad \text{when } P_{0VER} < -P_{D\_V}$$

$$F_D = 0 \quad \text{when } P_{0VER} \geq -P_{D\_V}$$

$$V_D = \int F_D$$

$$V_{O2\_D} = C_{O2\_D} V_D$$

$$V_{R\_D} = \left(1 - C_{O2\_D}\right) * V_D$$

3.7 O2 injector Model

3.7.1 PCM and Injector Led Models

PCM and Injector Led PPO2 controllers are part of the rebreather code. The simulation environment allows those control algorithms to be tested, but their characteristics are too complex for them to act as a reference model. The reference model should be simple to understand, demonstrate the dynamics of the rebreather and be able to be tested on the real hardware to ensure all the model parameters have been defined properly: for this a Bang-bang controller was chosen.

3.7.2 Reference Bang-Bang O2 injector

The bang-bang injector is a two level valve: it is controlled by a solenoid. The maximum O2 flow is set to default value of 6 l/min (0.1 l/s) for the mode without CO2 control. The 6l/min figure is the EN14143 requirement.

The valve time constant of the valve (less than 0.1s) is significantly less than the sensor time constant (min time constant of the control loop) that a solenoid valve is described as an ideal valve. The input of the injector input is a control bit: 0 or 1.
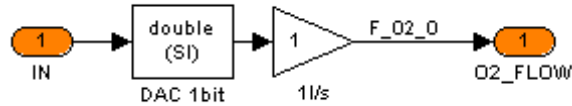
Figure 20. Model of ideal solenoid injector.

3.7.3 Analogue O2 injector

Analogue O2 injectors are needle valves, pinch valves and variable orifice valves. Models exist for each type of injector.

The analogue injector must supply O2 in the breathing loop to compensate for the O2 lost in breathing (metabolic loss) and after increasing the PPO2 setpoint. The default metabolic rate is 1.5 l/min (0.025 l/s). The range of the DL variable orifice O2 injector is from 0 to 6 l/min (in 640 steps). The corresponding diameter of the orifice is from 0 to 0.25 mm in a log scale.   This is modelled.

The Drager Dolphin circuit with the non-compensated pressure reduction valve (PRV) is used for constant mass flow applications up to the depth when the ambient pressure approximately twice less than the injector inlet pressure set by the PRV. The Ray circuit uses a compensated PRV to maintain the difference between the inlet and the ambient pressures constant. The mass flow of the injector supplied through the compensated valve depends on the depth and the "spring pressure" of the valve, as shown in Figure 21.
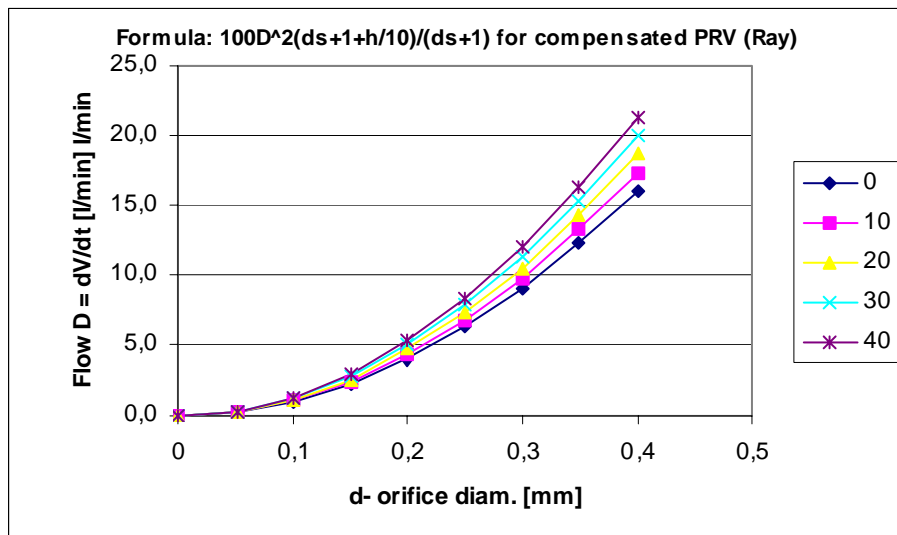
Figure 21. Ray injector.

3.8 O2 sensor model

The O2 sensor is modelled as a low frequency filter with a delay, plus various failure modes. Note the main delay in the rebreather model is the gas mass: when O2 is injected it must travel into the gas in the loop and be conveyed to the sensor. That delay is handled separately from the O2 sensor model. The O2 sensor model ignores multiple sensors: there is one sensor reading no matter how many sensors are fitted. This means that when a rebreather is modelled that uses, for example 4 sensors, all read the same apart from each will have its own calibrated voltage for the level in air (0.209% O2), and one or more may be subject to fault modes imposed by the simulator.

The frequency response of the O2 sensor alone is:

$$W_S = \frac{K_S}{T_S S + 1} e^{-T_d S}$$

where K_S is the gain; T_S is the sensor time constant T_d is the delay; $S = j\omega$ or $S = j2\pi f$

where f is frequency.
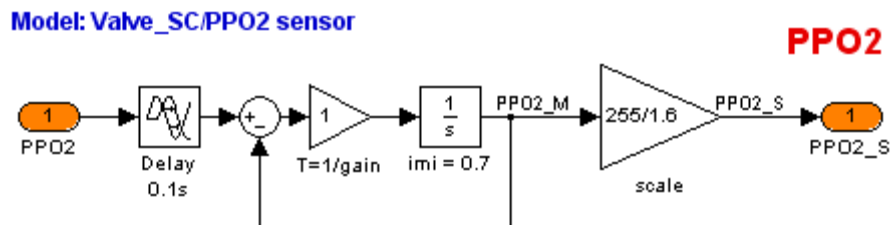
Figure 22. "Valve_SC/PPO2 sensor" model.

The parameters in the O2 sensor model are:

- K_{S_PPO2} PPO2 sensor gain

- T_{S_PPO2} PPO2 sensor time constant

- T_{D_PPO2} PPO2 sensor time delay

- ADC_{PPO2} ADC of PPO2 sensor (for the calibration scaling factor)

3.9 Verilog of the Bang-Bang reference controller with VHDL wrapper

Verilog HDL
```
//----------------------------------------------------------------------------------------
//   Design unit : VALVE_CONTROL
//   Filename: VLV_SC.v
//
//   Description: valve solenoid control
//          Limitations: none, except it is a Bang-bang controller
//   Author: Bob Davidov
//          Deep Life Ltd.
//
```

```
//   Simulator: ModelSim 6.0c
//
//   Version 1.1 Original release
//------------------------------------------------------------------------------------------

//`timescale 100ms / 10ms

module VLV_SC( SYS_CLK, DEPTH, NRESET, PPO2_SENS, SYNC_CLK, O2_FLOW);

input SYS_CLK, NRESET;
input [7:0] DEPTH;              // 8-bit ADC. Depth resolution = 1m, DL use a 15 bit ADC,
input [7:0] PPO2_SENS;         // 8-bit ADC of PPO2 sensor: DL use a 24 bit ADC

output SYNC_CLK, O2_FLOW;

reg O2_FLOW;                   // solenoid valve injector control
reg [7:0] PPO2_GOAL;           // 8-bit PPO2 set point (code 255 means PPO2 is 1.6)
                               // code 112 means PPO2 is 0.7
                               // code 191 means PPO2 is 1.2

assign SYNC_CLK = SYS_CLK; // control of the MatLAB clock

// Real time clock is 1 s
always @SYS_CLK                // Simulink lowest clock period is 2 s
  if ( !NRESET )
    begin
      if (DEPTH < 10)          // Depth in m
        PPO2_GOAL = 112;       // 0.7 of PPO2
      else
        PPO2_GOAL = 191;       // 1.2 of PPO2

      if (PPO2_SENS <  PPO2_GOAL)
        O2_FLOW = 1;
       else
         O2_FLOW = 0;
    end
  else
      O2_FLOW = 0;             //solenoid is off
endmodule
```

3.9.1 VHDL wrapper

The Simulink connects the verilog compiled data via the VHDL wrapper.

The ModelSim command: wrapverilog **verilog_file_name**.v runs generation of the VHDL file which name is **verilog_file_name**_wrap.vhd

The listing of the used VHDL file is as follows:

```
-- Module vlv_sc VHDL Wrapper
--
-- Generated by The MathWorks wrapverilog tcl command
-- Generated on: 2005-10-03 09:34:16
--
----------------------------------------------------------------
LIBRARY IEEE;
  USE IEEE.std_logic_1164.all;

ENTITY vlv_sc_wrap IS
    port(
        SYS_CLK        : in    std_logic;
        DEPTH          : in    std_logic_vector(15 downto 0);
        NRESET         : in    std_logic;
        PPO2_SENS      : in    std_logic_vector(7 downto 0);
        SYNC_CLK       : out   std_logic;
        O2_FLOW        : out   std_logic
    );

END vlv_sc_wrap;

ARCHITECTURE rtl OF vlv_sc_wrap IS

component VLV_SC
    port(
        SYS_CLK        : in    std_logic;
        DEPTH          : in    std_logic_vector(15 downto 0);
        NRESET         : in    std_logic;
        PPO2_SENS      : in    std_logic_vector(7 downto 0);
        SYNC_CLK       : out   std_logic;
        O2_FLOW        : out   std_logic
    );
end component;

FOR ALL : vlv_sc
  USE ENTITY work.vlv_sc(ignored);

BEGIN
  u_vlv_sc: vlv_sc
    PORT MAP (
      SYS_CLK  => SYS_CLK ,
      DEPTH  => DEPTH ,
      NRESET  => NRESET ,
      PPO2_SENS  => PPO2_SENS ,
```

```
  SYNC_CLK  => SYNC_CLK ,
  O2_FLOW  => O2_FLOW
);
```

END rtl;

4 EXAMPLE OF MODEL TEST RESULTS

4.1 Simulation parameters

During the run which forms the example considered here, the real time of the dive is 4380 seconds. It corresponds to the time in the dive profile.

The clock of the controller is set to a real time value of 1s in order to exaggerate effects such as the PPO2 control. The time step of the Simulink is 0.01 s. The step is less than the smallest time constant of the model. These parameters are set up in the Simulink control window as shown below.

4.2 Bang-Bang O2 control, 10Hz RT, Dive to Depth of 91m, for 810s

Decompression obligations are ignored: the purpose of this run is to test the rebreather models not the diver. Decompression algorithms used by DL are subject to separate verification.

The run uses the reference model, i.e. a Bang-bang control system, with 10Hz time constant, and total CO2 absorption. The descent to 91 m is in 810 seconds. The metabolic rate is 0.05 l/s.

Initial parameters

- ADC_{DEPTH} ADC of depth sensor: 16 bit

- ADC_{PPO2} ADC of PPO2 sensor: 16 bit

- C_{O2_D} O2 concentration in the diluent cylinder: 10%

- D_{FILE} Name of depth profile data file

  - file: depth_data_91_810.m (91 m at 810s, max descent rate of 0.3 m/s)

- F_{D_INI} Flow of the diluent when the valve is open: 1.2 l/s

- F_{O2_O} Injected O2 flow from the oxygen cylinder: 0.1 l/s

- F_{R_B} Flow of O2 consumption in breathing (metabolic rate): 0.05 l/s

- P_{D_V} Operating pressure of the diluent gas valve: -0.03 Bar

- P_{E_V} Operating pressure of the exhaust valve: 0.03 Bar

- RT_{CLK} Clock of the Real-Time: 0.1

- SP_{CO2} CO2 setpoint: 0.03   (3%)

- SP_{PPO2} PPO2 setpoint: 0.7/1.2 (the depth of switching is 10 m)

- T_A Start of the absorber CO2 output growth

           > 3750 (to switch off the absorber alarm mode)

- T_{D_PPO2} PPO2 sensor time delay: 0.1 s

- T_{S_PPO2} PPO2 sensor time constant: 1s (a sol-gel sensor: galvanic sensors are 6 - 10s)

- T_{SIM} Simulation time: 3750 s

- T_{90m} Descent time: 180 s

- V_{BL} Breathing loop space: 4 litres

- V_{O2_ini} Initial O2 volume in the breathing loop: 2.8 litres (70% of 4 litres)

- V_{R_ini} Volume of residual gas in the breathing loop: 1.2 litres (30% of 4 litres)
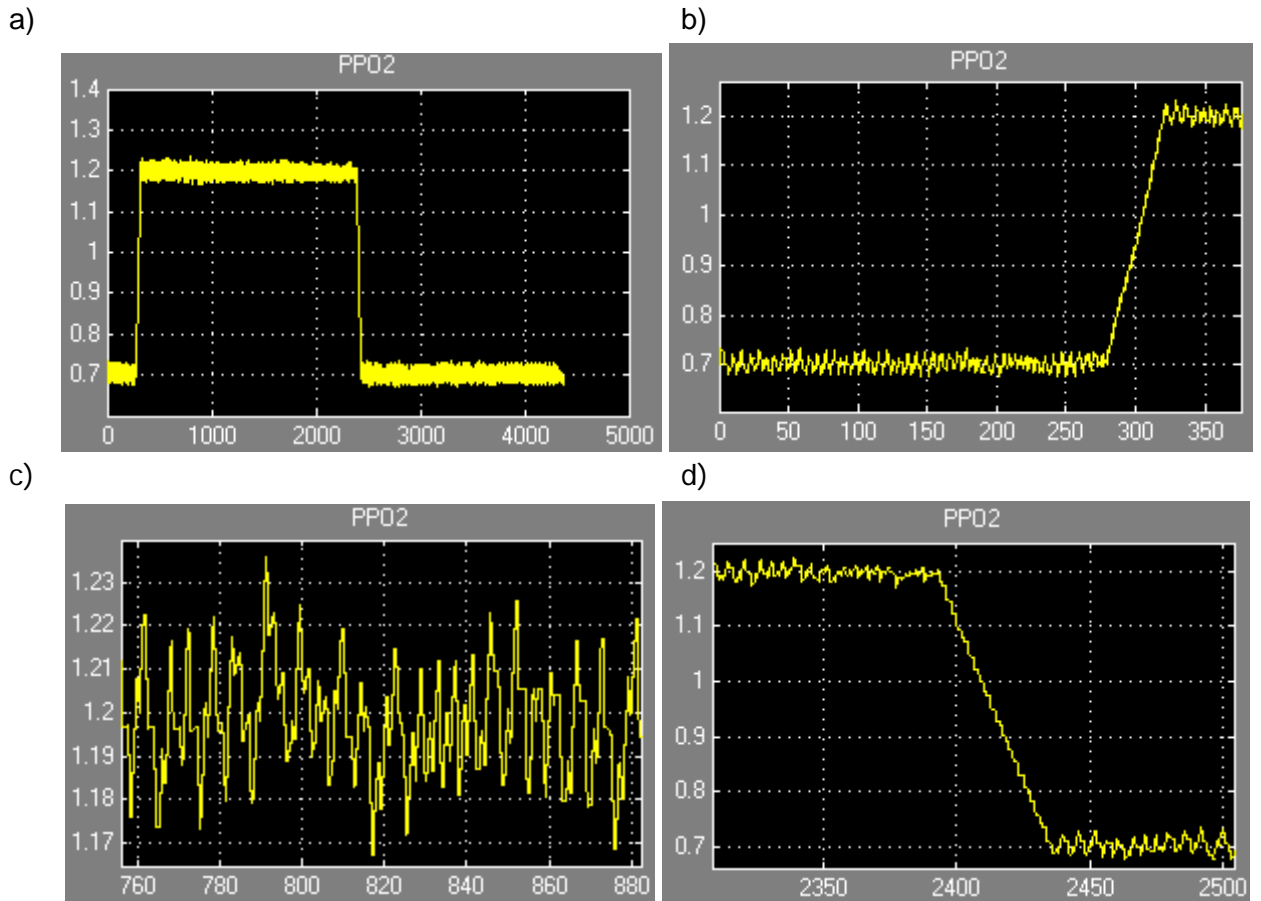
a)

b)

c)

d)

Figure 23. PPO2 sensor input: a) the profile for 4380 seconds; b) at the beginning of the dive; c) at the beginning of the ascent d) crossing of 10 m during ascent.
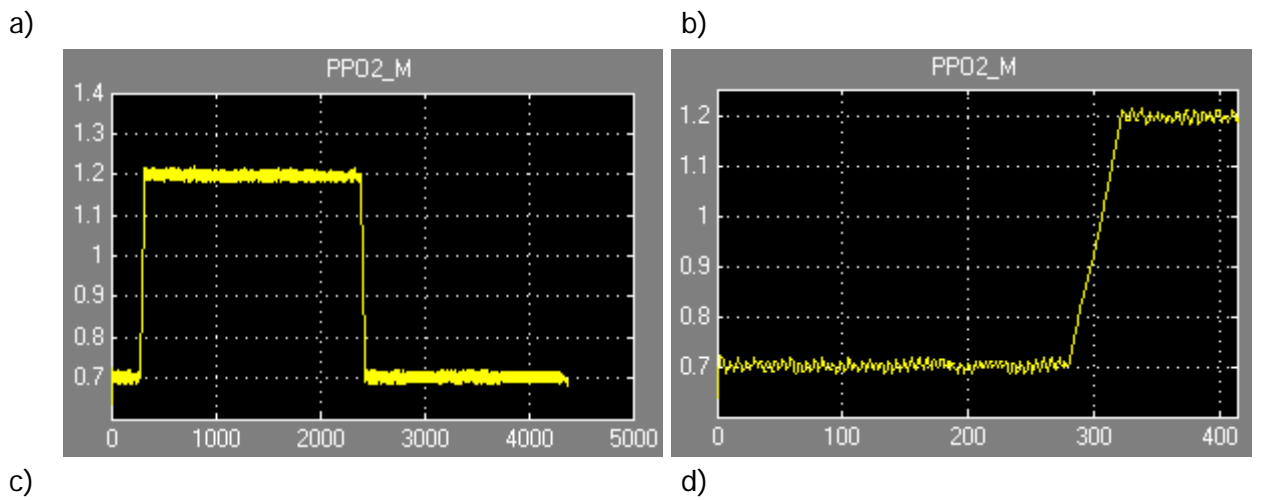
a)

b)

c)

d)
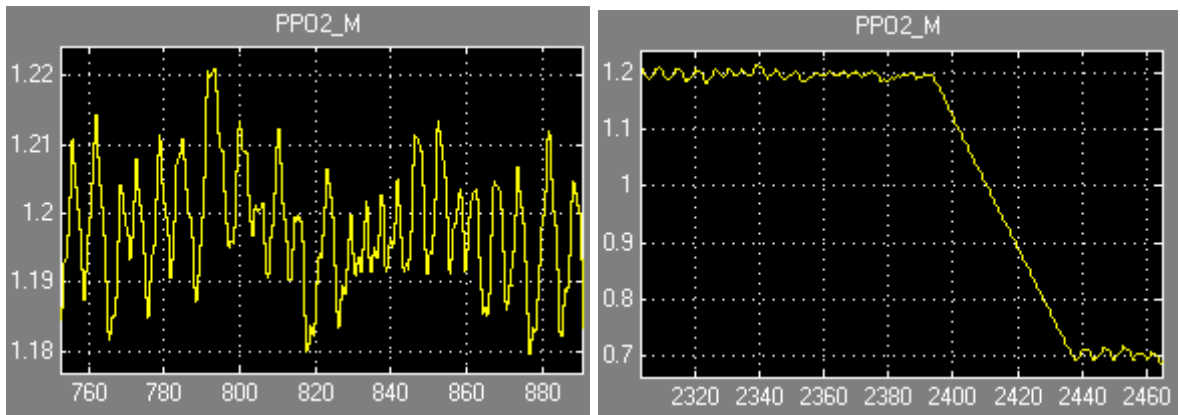
Figure 24. PPO2 sensor output: a) the profile for 4380 seconds; b) at the beginning of the dive; c) at the beginning of the ascent d) crossing of 10 m during ascent.
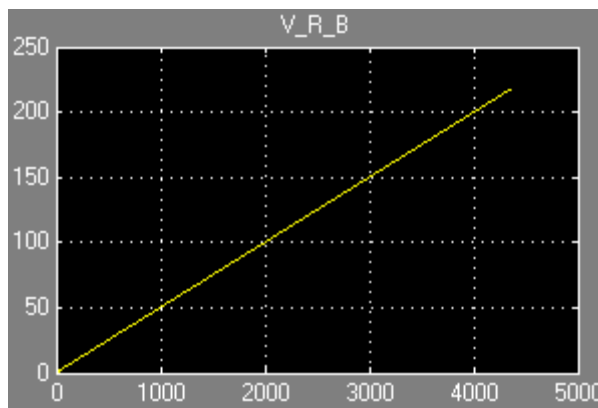
Figure 25. Volume of O2 loss by breathing (219 litres).
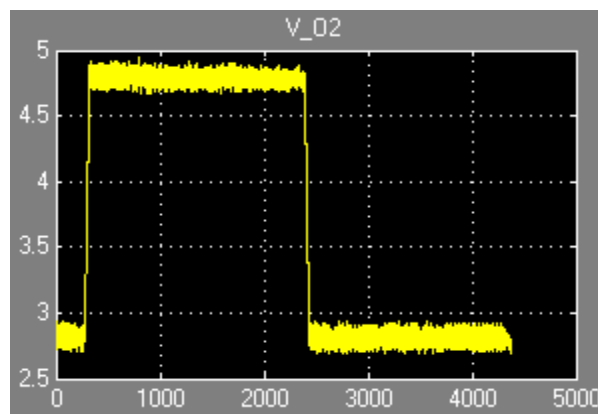
Figure 26. O2 volume inside the breathing loop.

Figure 27. O2 exhaust (15.9 litres).

Figure 28. O2 input from the oxygen cylinder (230.5 litres).
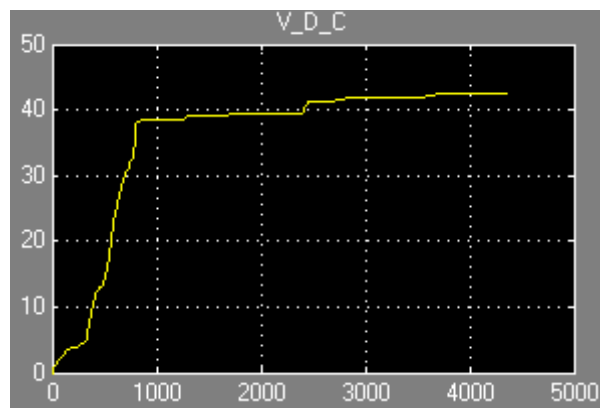
Figure 29. Diluent gas input (total is 42 litres, O2 is 10%).
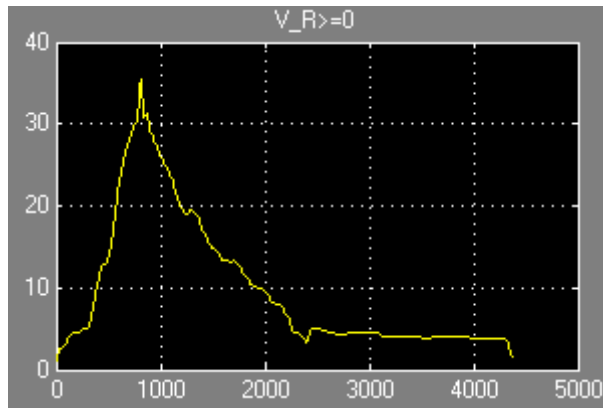
Figure 30. Residual gas volume inside the breathing loop.
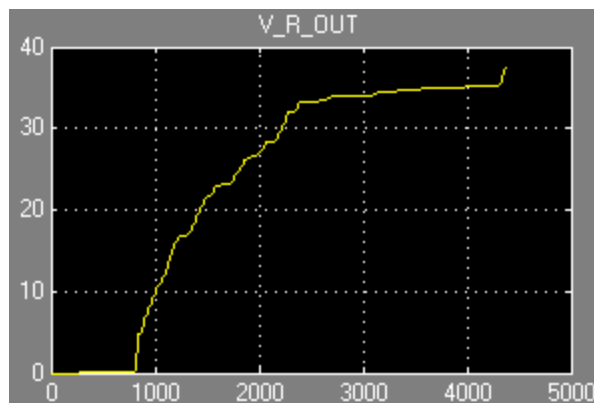
Figure 31. Exhaust of the residual gas (37.6 litres).
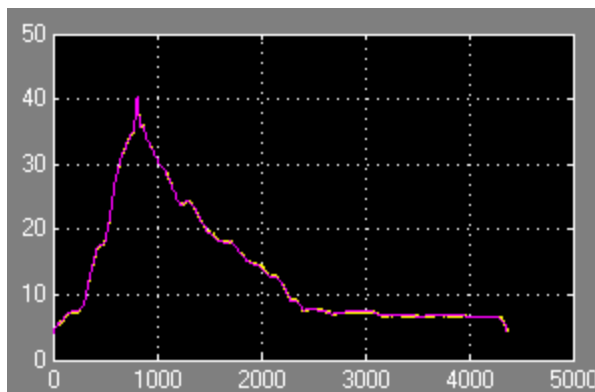
Figure 32. Gas volume inside the breathing loop under the ambient and inside pressure (Purple is ambient)
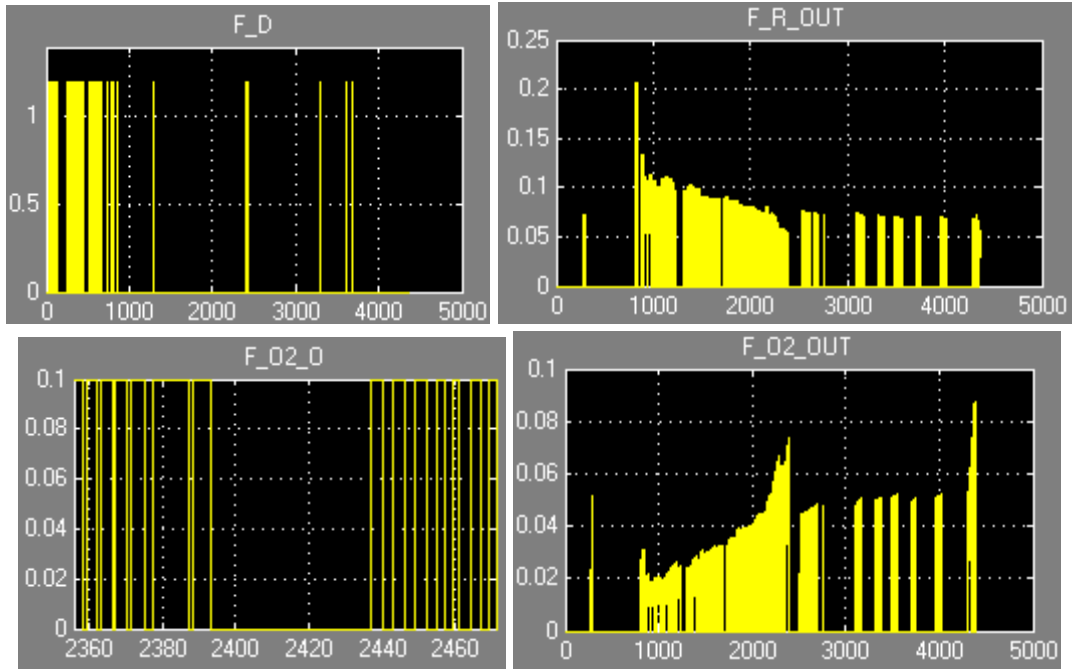
Figure 33. IN and OUT gas flows.

4.3 Model quality

The following parameters were read from the model during the test profile shown and checked manually:

O2 balance

    O2 in:

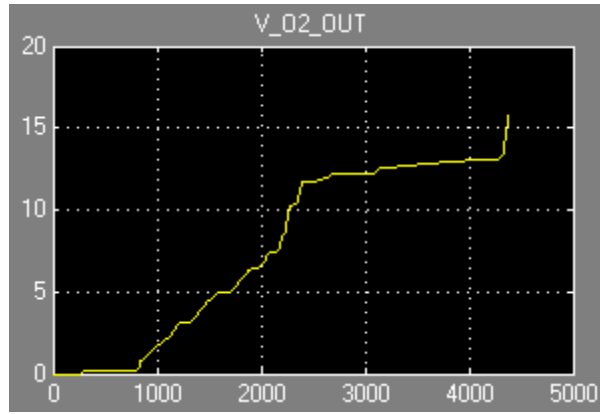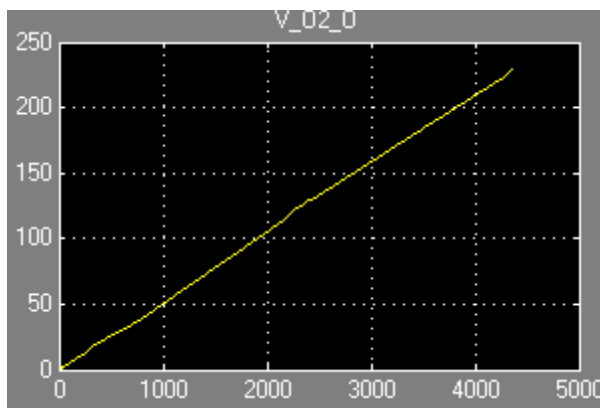- O2 from the oxygen cylinder   230.5 litres
- O2 from the diluent cylinder   4.2 litres

                            Total: 234.7 litres

    O2 consumption:

- breathing loss   219 litres
- O2 exhaust   15.9 litres

                            Total: 234.9 litres

The formula is

$$V_{O2\_O} + V_{O2\_D} = V_{R\_B} + V_{O2\_OUT}$$

Gas balance

    Gas in:

- O2 from the oxygen cylinder   230.5 litres
- Gas from the diluent cylinder   42 litres

                            Total: 272.5 litres

    Gas out:

- O2 exhaust   15.9 litres
- Residual gas exhaust   37.6 litres
- Breathing loss   219 litres
                                   Total: 272.5 litres

The formula is

$$V_{O2\_O} + V_{R\_D} = V_{O2\_OUT} + V_{R\_OUT} + V_{R\_B}$$

4.3.1 Variation of O2 in diluent gas

This test detects how O2 concentration in the diluent gas effects the PPO2 overhead. The system used for the check is the Bang-bang controller, 10 Hz real time, total CO2 absorption. The metabolic rate is 0.025 l/s.

Initial parameters

- ADC_{DEPTH} ADC of depth sensor: 16 bit

- ADC_{PPO2} ADC of PPO2 sensor: 16 bit

- C_{O2_D} O2 concentration in the diluent cylinder: 0%, 10%, 30%

- D_{FILE} Name of depth profile data file
    - file: depth_data_91_180.m (91 m at 180s, max descent of 1.3 m/s)
    - file: depth_data_91_810.m (91 m at 810s, max descent of 0.3 m/s)

- F_{D_INI} Flow of the diluent when the valve is open: 1.2 l/s

- F_{O2_O} Injected O2 flow from the oxygen cylinder: 0.1 l/s

- F_{R_B} Flow of O2 consumption in breathing (metabolic rate): 0.025 l/s

- P_{D_V} Operating pressure of the diluent gas valve: -0.03 Bar

- P_{E_V} Operating pressure of the exhaust valve: 0.03 Bar

- RT_{CLK} Clock of the Real-Time: 0.1

- SP_{CO2} CO2 setpoint: 0.03   (3%)

- SP_{PPO2} PPO2 setpoint: 0.7/1.2 (the depth of switching is 10 m)

- T_A Start of the absorber CO2 output growth

    > 3750 (absorber life time is more than the test duration)

- T_{D_PPO2} PPO2 sensor time delay: 0.1 s

- T_{S_PPO2} PPO2 sensor time constant: 1 s

- T_{SIM} Time of simulation: 3750 s

- T_{90m} Time of 90m descent: 180 s

- V_{BL} Breathing loop space: 4 litres

- V_{O2_ini} Initial O2 volume in the breathing loop: 2.8 litres (70% of 4 litres)

- V_{R_ini} Volume of residual gas in the breathing loop: 1.2 litres (30% of 4 litres)

Plotted parameters:

- $PPO2$ O2 partial pressure inside the breathing loop
- F_{O2} Injected O2 flow from the oxygen cylinder
- V_{O2_D} O2 volume from the diluent cylinder
- V_{R_B} volume of O2 consumpted in breathing
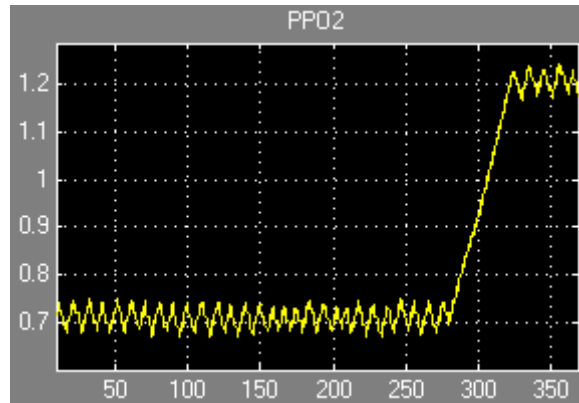

Figure 34. PPO2 levels with a metabolic rate of 3 l/min (0.05 l/s), Depth 91 m at 810s, 10% diluent O2.
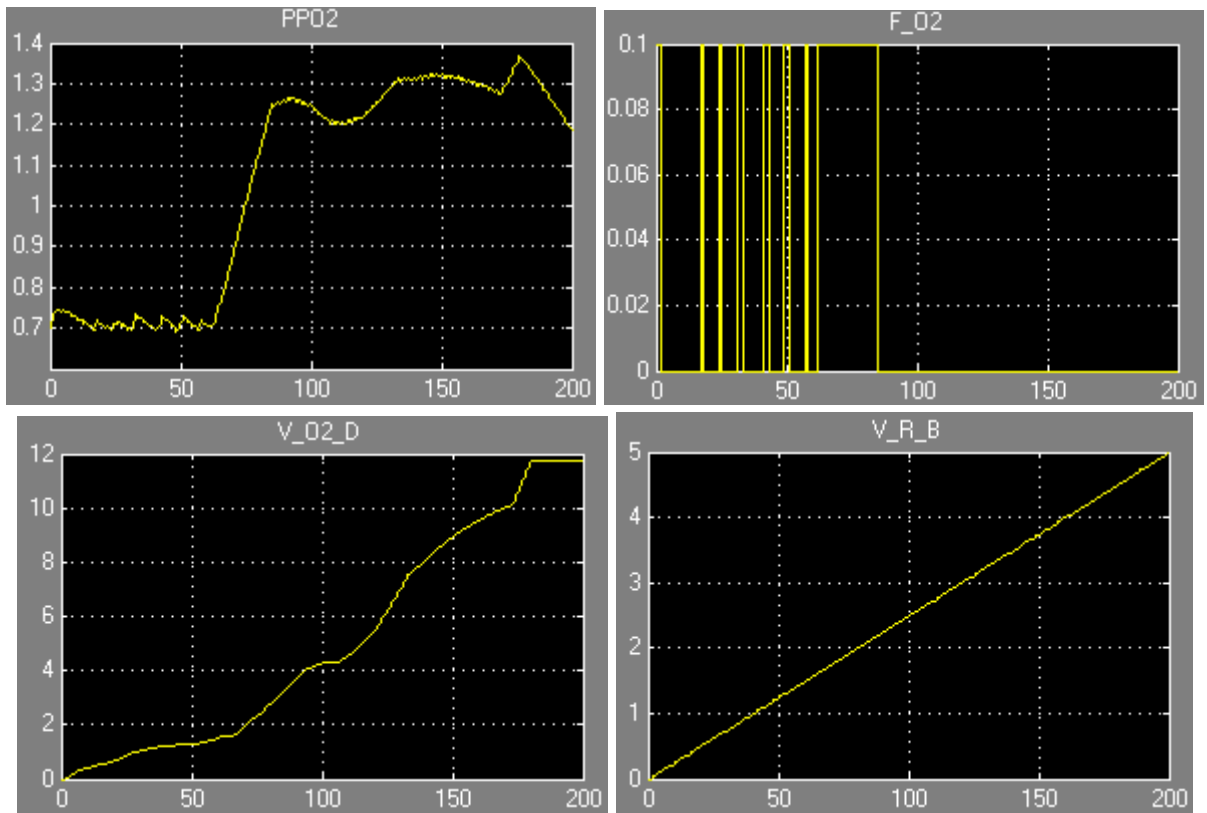

Figure 35. PPO2 levels with a metabolic rate of 1.5 l/min (0.025 l/s), Depth 91 m at 180s, 10% diluent O2. (LABEL ALL GRAPHS, a, b, c, d)
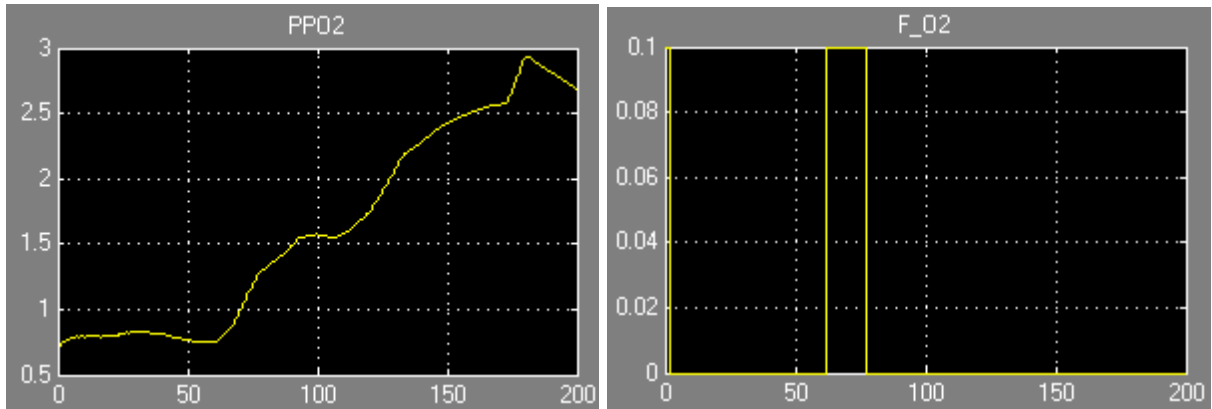
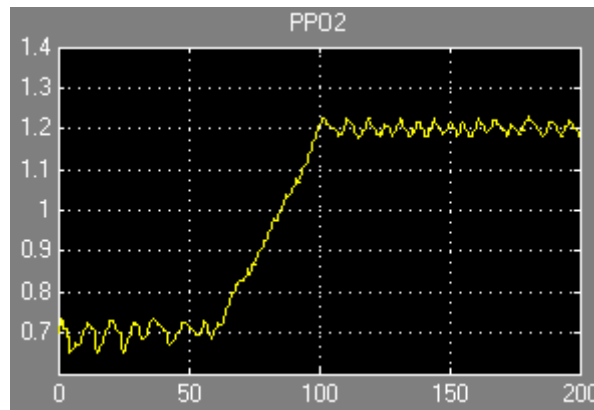**Figure 36. PPO2 with a metabolic rate of 1.5 l/min (0.025 l/s), Depth 91 m at 180s, 30% diluent O2.**

Figure 37. PPO2 with a metabolic rate of 1.5 l/min (0.025 l/s), Depth 91 m at 180s, 0% diluent O2.

Conclusion

1. During descent the overhead of PPO2 is proportional to the O2 incoming from the diluent cylinder that is not consumed by breathing. To decrease the PPO2 overhead it is necessary to decrease the velocity of descent or/and the O2 concentration in the diluent gas. Increased transformation of O2 into CO2 decreases the overhead.

2. The more the concentration of O2 in the diluent gas than less the max depth of diving must be.

3. PPO2 limits the max depth of the dive with some diluents. With 30% diluent O2 and 0.7/1.2 PPO2 setpoint at 10 m, for example, the alarm PPO2 of 1.6 is reached on the 40 m depth at 100 s (see PPO2 of Figure 36). Lets assumed that depth is 10m, the PPO2 is 1.2 and the concentration of O2 in diluent gas is 30%. The maximum increasing of PPO2 is 0.4 (as 1.6-1.2). The depth where the PPO2 gets the max level of 1.6 (without breathing or with the breathing that is much less than the diluent O2 income) is 23 m. It is 10 m + (0.4/0.3)*10m.

4. The simulator has provision for gas switching. For deep dive profiling this is necessary to keep PPO2 within limits, otherwise the act of a flush or simply a fast descent causes the PPO2 to exceed the set limits – especially if the metabolic rate is low.

Suggestion (to RB controller team)

1. Indicate "Stop descent" when the PPO2 is too high

4.4 Bang-Bang O2 control, 10Hz RT, depth 91m/180s

The test shows how the faster descent (relative to descent in 810s) and the lower metabolic rate (twice less than 0.05 l/s) effects the PPO2.

The time of descent on the 91m profile is 810 seconds. The model is rerun, again using the Bang-bang control system with 10 Hz real time and a total CO2 absorption, but with the following initial parameters.

Initial parameters

- ADC_{DEPTH} ADC of depth sensor: 16 bit
- ADC_{PPO2} ADC of PPO2 sensor: 16 bit
- C_{O2_D} O2 concentration in the diluent cylinder: 10%
- D_{FILE} Name of depth profile data file
  - file: depth_data_91_810.m (91 m at 180s, max descent of 1.3 m/s)
- F_{D_INI} Flow of the diluent when the valve is open: 1.2 l/s
- F_{O2_O} Injected O2 flow from the oxygen cylinder: 0.1 l/s
- F_{R_B} Flow of O2 consumption in breathing (metabolic rate): 0.025 l/s
- P_{D_V} Operating pressure of the diluent gas valve: -0.03 Bar
- P_{E_V} Operating pressure of the exhaust valve: 0.03 Bar
- RT_{CLK} Clock of the Real-Time: 0.1
- SP_{CO2} CO2 setpoint: 0.03   (3%)
- SP_{PPO2} PPO2 setpoint: 0.7/1.2 (the depth of switching is 10 m)
- T_A Start of the absorber CO2 output growth
  - > 3750 (to switch off the absorber alarm mode)
- T_{D_PPO2} PPO2 sensor time delay: 0.1 s
- T_{S_PPO2} PPO2 sensor time constant: 1 s
- T_{SIM} Time of simulation: 3750 s
- T_{90m} Time of 90m descent: 180 s
- V_{BL} Breathing loop space: 4 litres
- V_{O2_ini} Initial O2 volume in the breathing loop: 2.8 litres (70% of 4 litres)
- V_{R_ini} Volume of residual gas in the breathing loop: 1.2 litres (30% of 4 litres)

a)

b)

c)

d)

Figure 38. PPO2 sensor input: a) the profile for 4380 seconds; b) at the beginning of the dive; c) at the beginning of the ascent d) crossing of 10 m during ascent. Note the excess PPO2 due to the low metabolism and fast descent.

Figure 39. Volume of O2 loss by breathing (93.7 litres).

Figure 40. O2 volume inside the breathing loop.
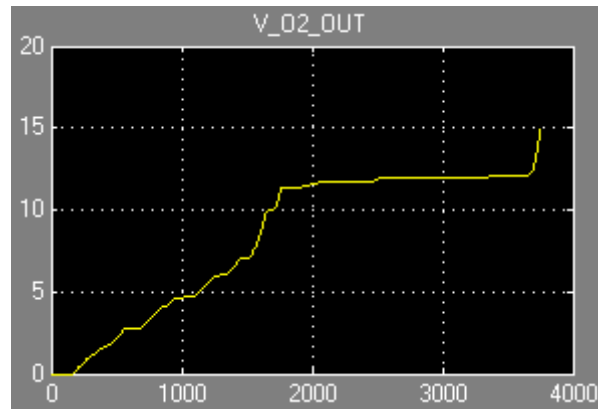
Figure 41. O2 exhaust (15 litres).
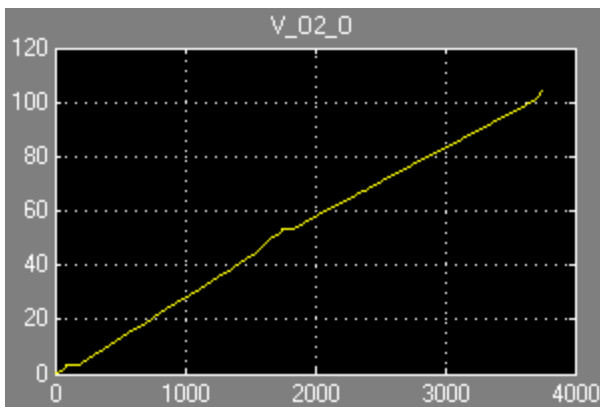
Figure 42. O2 input from the oxygen cylinder (104.8 litres).
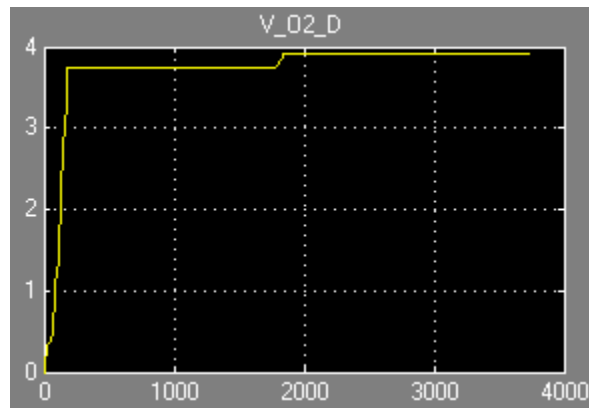
Figure 43. 10% O2 input from the diluent cylinder (3.92 litres).
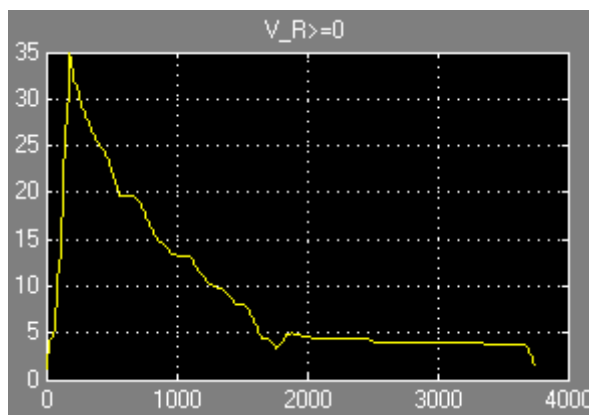
Figure 44. Residual gas volume inside the breathing loop.
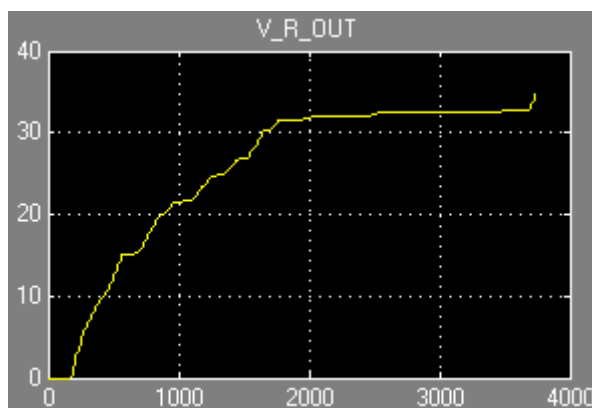
Figure 45. Exhaust of the residual gas (35.1 litres).
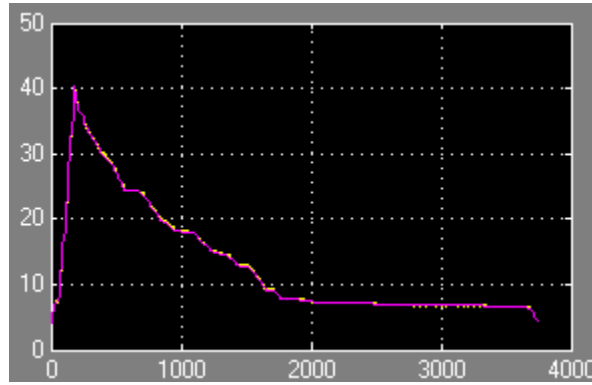
Figure 46. Gas volume inside the breathing loop under the ambient and inside pressure.
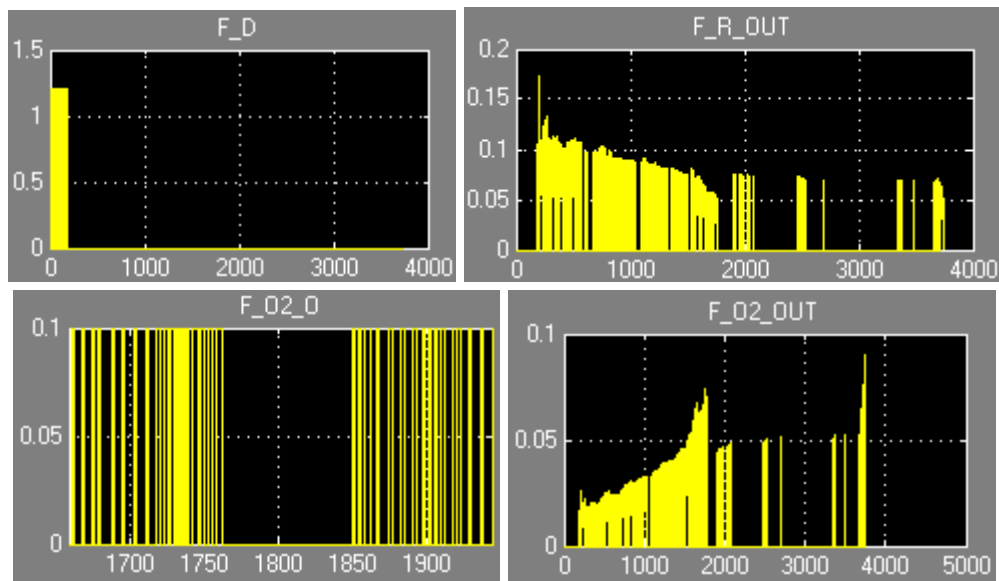
Figure 47. IN and OUT gas flows.
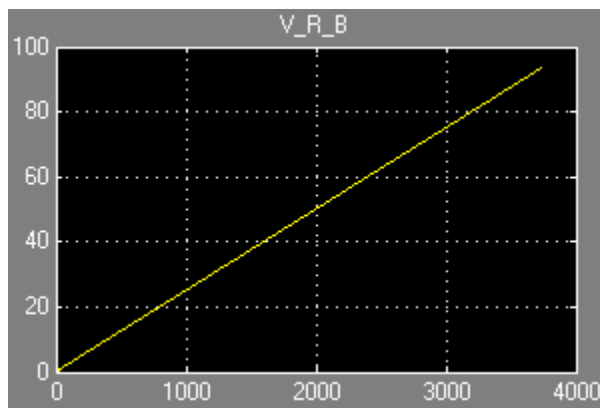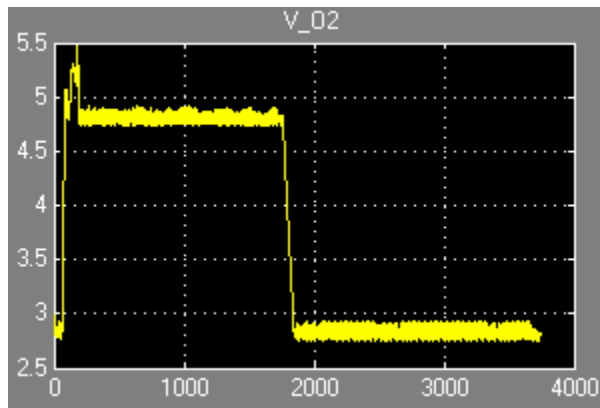
The checks of model quality from this run give the following results:

O2 balance

O2 in:
- O2 from the oxygen cylinder   104.8 litres
- O2 from the diluent cylinder   3.92 litres

                                    Total: 108.7 litres

O2 consumption:
- breathing loss   93.7 litres
- O2 exhaust   15 litres

                                    Total: 108.7 litres

The formula is

$$V_{O2\_O} + V_{O2\_D} = V_{R\_B} + V_{O2\_OUT}$$

Gas balance

Gas in:
- O2 from the oxygen cylinder   104.8 litres
- Gas from the diluent cylinder   39.2 litres
                           Total: 144 litres

Gas out:
- O2 exhaust   15 litres
- Residual gas exhaust   35.1 litres
- Breathing loss   93.7 litres
                           Total: 143.8 litres

The formula is

$$V_{O2\_O} + V_{R\_D} = V_{O2\_OUT} + V_{R\_OUT} + V_{R\_B}$$

Conclusion

1. The above equation shows that the model is correct.

2. At crossing the 10 m depth (during ascent) the setpoint of PPO2 is decreased from 1.2 to 0.7. The arise PPO2 overhead is decreased by breathing during 90 seconds (in the range from 1760 to 1850 seconds). The loss of gas inside the breathing loop is the absorbed CO2. It is compensated by input gas from the diluent cylinder.

4.5 CO2 concentration, CO2 control = OFF

The test of the model without CO2 control that switches off the absorption at the max depth of 91m at 180 sec. Assumption: the metabolic rate is constant in the breathing loop having any CO2 concentration.

Initial parameters

- ADC_{DEPTH} ADC of depth sensor: 16 bit

- ADC_{PPO2} ADC of PPO2 sensor: 16 bit

- C_{O2_D} O2 concentration in the diluent cylinder: 10%

- D_{FILE} Name of depth profile data file
    - file: depth_data_91_810.m (91 m at 180s, max descent of 1.3 m/s)

- F_{D_INI} Flow of the diluent when the valve is open: 1.2 l/s

- F_{O2_O} Injected O2 flow from the oxygen cylinder: 0.1 l/s

- F_{R_B} Flow of O2 consumption in breathing (metabolic rate): 0.025 l/s

- P_{D_V} Operating pressure of the diluent gas valve: -0.03 Bar

- P_{E_V} Operating pressure of the exhaust valve: 0.03 Bar

- RT_{CLK} Clock of the Real-Time: 0.1

- SP_{CO2} CO2 setpoint: 0.03   (3%)

- SP_{PPO2} PPO2 setpoint: 0.7/1.2 (the depth of switching is 10 m)

- T_A Start of the absorber CO2 output growth
          180 s (at the max depth)

- T_{A_G} Time of the CO2 growth in the absorber output (absorber agony): 360 s

- T_{D_PPO2} PPO2 sensor time delay: 0.1 s

- T_{S_PPO2} PPO2 sensor time constant: 1 s

- T_{SIM} Time of simulation: 3750 s

- T_{90m} Time of 90m descent: 180 s

- V_{BL} Breathing loop space: 4 litres

- V_{O2_ini} Initial O2 volume in the breathing loop: 2.8 litres (70% of 4 litres)

- V_{R_ini} Volume of residual gas in the breathing loop: 1.2 litres (30% of 4 litres)
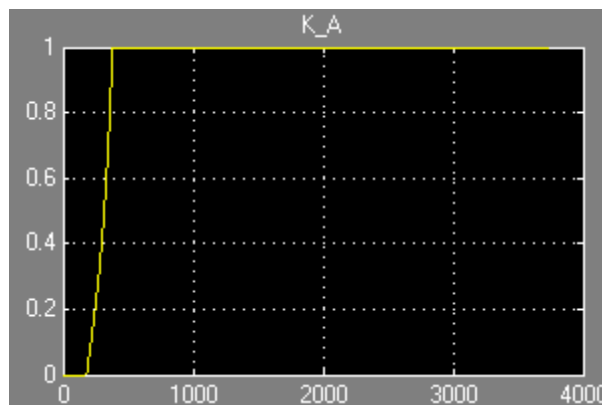
Figure 48. Absorber attenuation changing in the range [810-1012.6] sec.
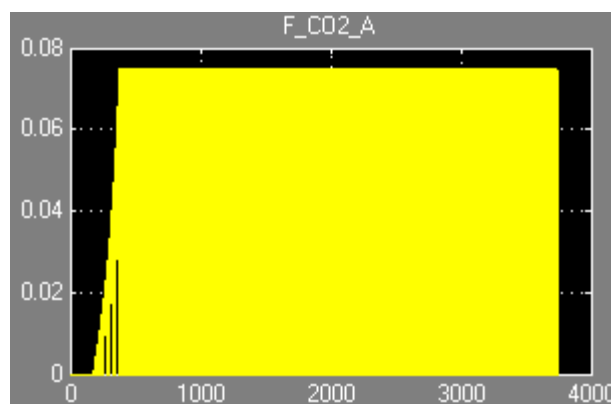
Figure 49. CO2 absorber output flow.
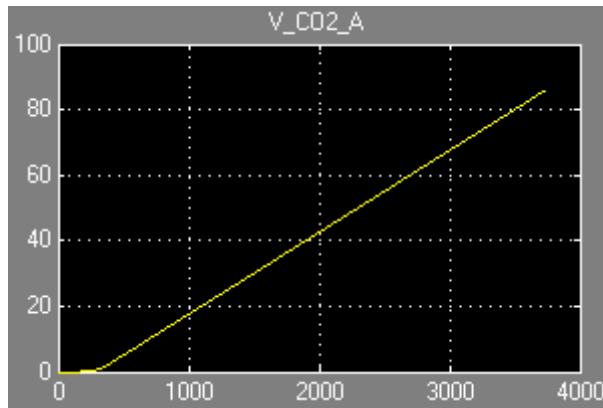
Figure 50. Volume of CO2 passed through the absorber (86.1 litres).
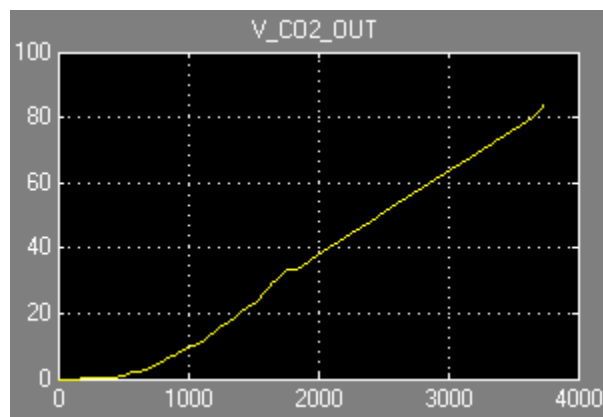
Figure 51. Exhaust of CO2 (84.6 litres).
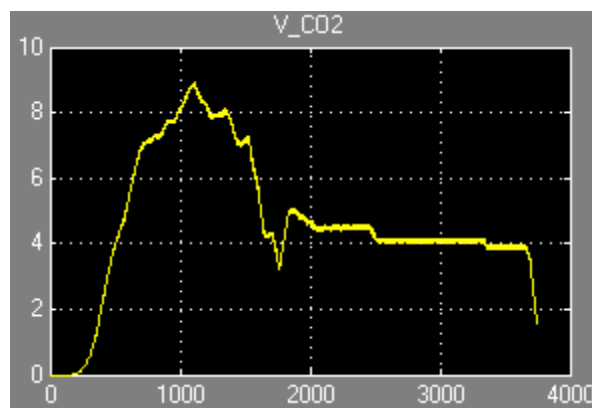
Figure 52. CO2 in the breathing loop (1.52 litre).
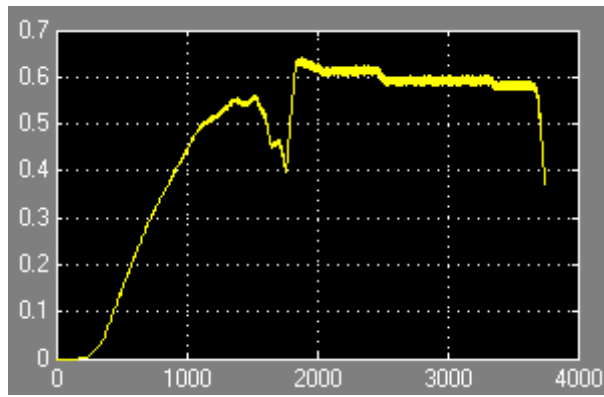
Figure 53. CO2 concentration.
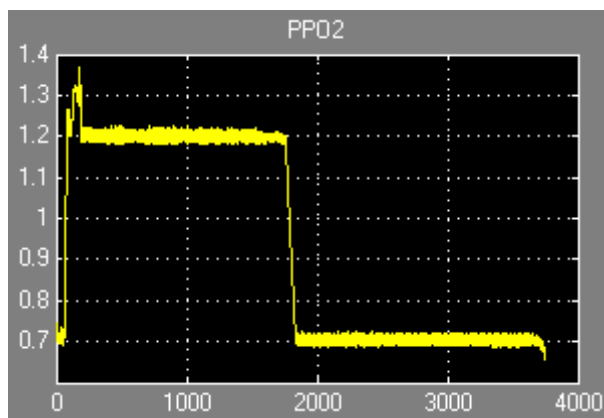
Figure 54. PPO2
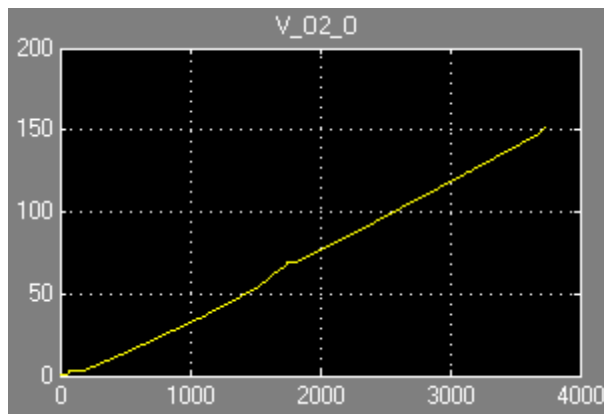
Figure 55. O2 injection from the oxygen cylinder (153.2 litres).
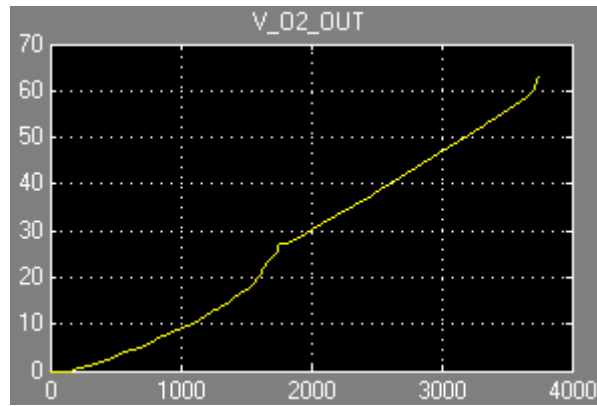
Figure 56. O2 exhaust (63.4 litres).
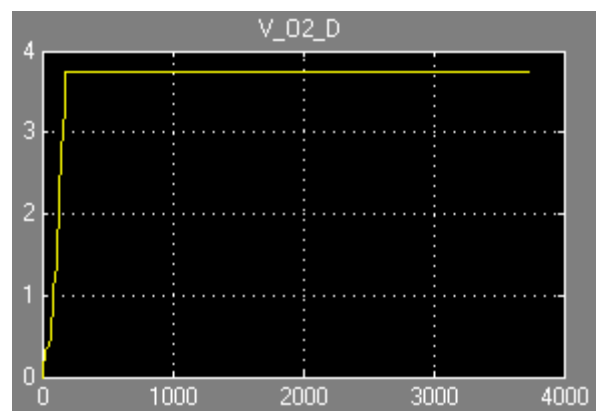
Figure 57. 10% O2 diluent gas injection from the diluent cylinder (3.73 litres).
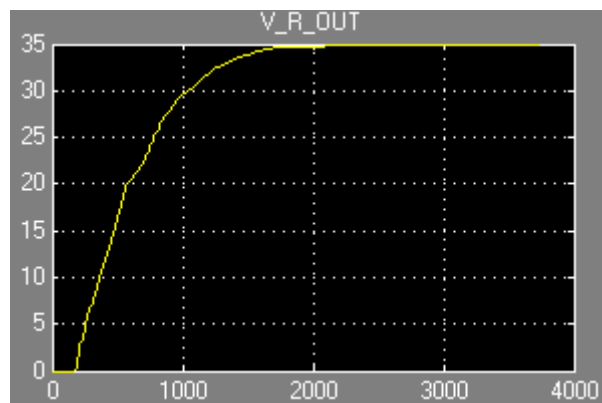
Figure 58. Residual gas (without O2 and CO2) exhaust (34.8 litres).

O2 balance

O2 in:
- O2 from the oxygen cylinder   153.2 litres

- O2 from the diluent cylinder   3.73 litres

               Total: 156.93 litres

O2 consumption:
- breathing loss   93.7 litres
- O2 exhaust   63.4 litres

               Total: 157.1 litres

Gas balance

Gas in:
- O2 from the oxygen cylinder   153.2 litres
- Gas from the diluent cylinder   37.3 litres

               Total: 190.5 litres

Gas out:
- O2 exhaust   63.4 litres
- Residual gas exhaust   34.8 litres
- CO2 exhaust   84.6 litres
- Absorbered CO2   7.6 = (93.7 – 86.1) litres

               Total: 190.4

Conclusion

1. The above gas balance shows that the model is correct.

2. The diluent gas does not injected during ascent and decreasing of the PPO2 setpoint, because the decreasing of the gas by absorption is off.

3. Maximum concentration of CO2 in the breathing loop is reached: 60% (see Figure 53)

5 SANITY CHECKS ON MODEL RESULTS

The checks below are not validation of the model, simply checks that it is doing sensible things. Full validation is part of the software validation subproject.

1. During the example dive (see BL&VCM_report051010.doc: the depth profile in Figure 4 and the test initial parameters on the page 55) the O2 consumption from the oxygen cylinder is 337 litres, the diluent gas consumption is 16.4 litres.

1.1. Including transmission of all exhaled CO2 into the liquid phase, the O2 input from the oxygen cylinder is 260.5 litres the diluent gas input is 109.2 litres (see Figure 14 and 15 of BL&VCM_report051014.doc)..

1.2. Increasing the exhaust valve and diluent valve operating pressure from 0 to 0.03 Bar and decreasing the time constant of the PPO2 sensor from 1 to 0.1 sec., the O2 input from the oxygen cylinder is 230.7 litres the diluent gas input is 42.4 litres (see Figure 14 and 15 of BL&VCM_report051020.doc). The diluent gas (about 39 litres) is spent mainly during descent.

2. Attempt to increase the Real Time frequency of the Simulink-ModelSim model is unsuccessful. MatLAB R14SP1 has a bug in the "Link for ModelSim" block that does not support the shared-memory mode. It also described as a bug on Internet forums. The Simulink-ModelSim co-simulation via the TCP/IP socket is limited to 1 Hz.

2.1. An adequate model of the bang-bang controller was built in MatLAB Simulink (see Figure 5 of BL&VCM_report051022.doc). The O2 input from the oxygen cylinder is 230.3 litres(it was 230.7 litres with ModelSin) the diluent gas input is 41.8 litres (it was 42.4 litres in ModelSim)

3. The point of 1.2 PPO2 was set in the 10 m depth at 275 s however the 1.2 PPO2 level inside the breathing loop was reached in the depth of 18 m at 340 s. The reason of this delay is the limited O2 injection of 6l/min.

4. PPO2 error near the 0.7 level from a Bang-bang controller with the settings shown is about 11% (PPO2 sensor time constant is 1 s, Depth and PPO2 ADC is 8 bit), 11% (PPO2 sensor time constant is 0.1 s, Depth and PPO2 ADC is 8 bit) 7.1% == 100x(0.73-0.68)/0.7 (PPO2 sensor time constant is 0.1 s, Depth and PPO2 ADC is 16 bit).

5. The process of breathing changes the PPO2 at the respiratory rate. When the O2 consumption in breathing (metabolic rate) is 0.025 l/s the absorption decreases the gas volume at 0.1 litres each 2 seconds. If the PPO2 level is 0.7 in the 4 litres space the above absorption decreases PPO2 at 1.1%

5.1 PPO2 depends on
  o metabolic parameters
  o diluent input (10% O2)
  o gas exhaust
  o O2 control system (Bang-bang reference controller or actual PCM or Injector led control).

5.2 The PPO2 level for a Bang-bang controller when the PPO2 setpoint and the depth are constant (the diluent input and gas exhausting are zero) is shown in the table.

PPO2 = 0.7; Depth = 2m;

PPO2 ADC, bit		PPO2 sensor time constant, s		Real – Time lock, s		PPO2 inside BR		PPO2, %
8	16	0.1	1	0.1	1	min	max	
X			X		X	0.67	0.73	8.6
X			X	X		0.685	0.724	5.6
X		X			X	0.685	0.722	5.3
X		X		X		0.697	0.71	1.9
	X		X		X	0.667	0.732	6.1
	X		X	X		0.686	0.724	5.4
	X	X			X	0.684	0.723	5.6
	X	X		X		0.697	0.71	1.9

5.3 The PPO2 sensor with the time constant of 1s works as a low pass filter of the PPO2 variation with the pole of 1/2pi Hz. It smoothes the real PPO2 and limits the accuracy the valve control. To increase accuracy of the PPO2 control up to 0.1% it needs the sensors passing the breathing fluctuation higher than 1Hz, or a PCM or Injector Led controller.

6. Having the existing model of the breathing loop and measuring O2 loss via CO2 consumption it is possible to define PPO2 level when the PPO2 sensor is off.

7. Required maximum flow of the diluent gas must be at least $F_D = \dfrac{V_{BL}}{10/D_{\max}}$, where V_{BL} is the volume of the breathing loop in litres; $D_{\max}$ is the maximum dive depth speed.

6 FUTURE DEVELOPMENT FOR THE RB MODEL

1. Model optimisation

   - Simulation parameters (time, method, resolution, ets) controlled via the M file as well as via the GUI

   - Reading simulated data in the real-time mode

   - GUI with the structure similar to that developed by the DL team for the StatEye GUI (www.stateye.org) for the model development, as distinct from the present GUI which is for the whole simulator

   - CO2 sensor parameters

   - Valve parameters

   - Dolphin and Ray modes

   - Dependences of cylinder gas flow on the temperature

   - Flow dynamics parameterised by the physical characteristics of the O2 injector orifice area

2. Decreasing the real-time clock in Simulink-ModelSim co-simulation by at least 10 times to 0.1 sec.

3. Connection of Simulink to rebreather via the PC DAC/ADC interface.

The example of the **"Simulink – Rebreather – GUI"** connection is shown in Figure 59.

Simulink applies the simulated depth signal to the rebreather electronic instead of the depth sensor output. It provides testing of the rebreather functionality in atmospheric condition close to diving condition.
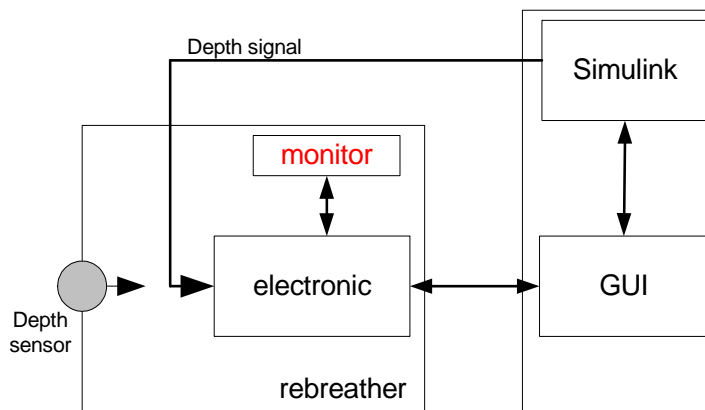

Figure 59. Simulink – Rebreather - GUI block structure.

APPENDIX A: INDEX OF ALL PARAMETERS

Model Parameters

- ADC_{PPO2} ADC of PPO2 sensor
- ADC_{DEPTH} ADC of depth sensor
- C_{CO2} CO2 concentration in the breathing loop
- C_{O2_D} O2 concentration in the diluent cylinder
- D Depth of diving
- D_{FILE} Name of depth profile data file
- E_{CO2} Error of CO2 concentration (overlevel)
- EXH_P Exhausting part of the total gas
- F_B Breathing total flow
- F_D Flow of the diluent gas
- F_{D_INI} Flow of the diluent when the valve is open
- F_{CO2_A} Part of the breathing CO2 flow passed though the absorber
- F_{CO2_ADD} CO2 flow growth in the loop proportional to the absorber CO2 output
- F_{O2_O} Injected O2 flow from the oxygen cylinder
- F_{O2_OUT} Flow of O2 exhaust
- F_{R_B} Flow of O2 consumption in breathing (metabolic rate)
- F_{O2_OUT} Flow of residual gas exhaust
- K_A Absorbers CO2 attenuation
- K_G Absorbers CO2 output growth speed during absorber agony
- K_{S_PPO2} PPO2 sensor gain
- P_A Ambient pressure
- P_{D_T} Total pressure applied to the diluent valve
- P_{D_V} Operating pressure of the diluent gas valve
- P_{D_VC} Diluent valve control pressure
- P_{E_V} Operating pressure of the exhaust valve
- P_{int} Pressure inside the breathing loop
- P_{OVER} Over pressure inside the breathing loop relative to the ambient pressure
- $PPO2$ O2 partial pressure inside the breathing loop
- $PPO2_{INI}$ Initial PPO2 in the breathing loop

- $PPO2_M$ Measured O2 partial pressure inside the breathing loop
- $PPO2_S$ Scaled PPO2
- RT_{CLK} Clock of the Real-Time
- SP_{CO2} CO2 setpoint
- SP_{PPO2} PPO2 setpoint
- T_A Start of the absorber CO2 output growth
- T_{A_G} Time of the CO2 growth in the absorber output (absorber agony)
- T_{D_PPO2} PPO2 sensor time delay
- T_{S_PPO2} PPO2 sensor time constant
- T_{SIM} Time of simulation
- T_{90m} Time of 90m descent
- V_{A_S} Absorbed CO2 at the beginning of the CO2 output growth
- V_{BL} Breathing loop space
- V_{CO2} CO2 volume of the breathing loop
- V_{CO2_A} Volume of the absorber CO2 output
- V_{CO2_OUT} CO2 exhaust
- V_D Diluent gas consumption
- V_N Gas volume inside the breathing loop under the ambient pressure
- V_{O2} O2 volume inside the breathing loop
- V_{O2_D} O2 volume from the diluent cylinder
- V_{O2_O} O2 consumption from the oxygen cylinder
- V_{O2_OUT} Volume of O2 exhaust
- V_R Residual gas volume inside the breathing loop
- V_{R_B} volume of O2 consumption in breathing
- V_{R_D} Residual gas volume from the diluent cylinder
- V_{R_OUT} Exhaust of the residual gas
- V_T Total gas volume inside the breathing loop

NOTE: The residual gas is the total gases except of O2 and CO2.

Initial parameters of the valve control model

- ADC_{DEPTH} ADC of depth sensor: 16 bit

- ADC_{PPO2} ADC of PPO2 sensor: 16 bit

- C_{O2_D} O2 concentration in the diluent cylinder: 0%; 10%; 30%

- D_{FILE} Name of depth profile data file
    - file: depth_data_91_180.m (91 m at 180s, max descent of 1.3 m/s)
    - file: depth_data_91_810.m (91 m at 810s, max descent of 0.3 m/s)

- F_B Breathing total flow: 1.2 l/s

- F_{D_INI} Flow of the diluent when the valve is open: 1.2 l/s

- F_{O2_O} Injected O2 flow from the oxygen cylinder: 0.1; 1 l/s

- F_{R_B} Flow of O2 consumption in breathing (metabolic rate): 0.025 l/s

- K_G Absorbers CO2 output growth speed during absorber agony: 1/9

- K_{S_PPO2} PPO2 sensor gain: 255/1.6

- P_{D_V} Operating pressure of the diluent gas valve: -0.03 Bar

- P_{D_VC} Diluent valve control pressure: -1 Bar   (< -2*0.03 Bar)

- P_{E_V} Operating pressure of the exhaust valve: 0.03 Bar

- $PPO2_{INI}$ Initial PPO2 in the breathing loop: 0.7

- RT_{CLK} Clock of the Real-Time: 0.1

- SP_{CO2} CO2 setpoint: 0.03   (3%)

- SP_{PPO2} PPO2 setpoint: 0.7/1.2 (the depth of switching is 10 m)

- T_A Start of the absorber CO2 output growth
    - 0 s (for the worst case of the absorber alarm mode)
    - 180 s (at the max depth)
    - > 3750 (to switch off the absorber alarm mode)

- T_{A_G} Time of the CO2 growth in the absorber output (absorber agony): 360 s

- T_{D_PPO2} PPO2 sensor time delay: 0.1 s

- T_{S_PPO2} PPO2 sensor time constant: 1 s

- T_{SIM} Time of simulation: 3750 s

- T_{90m} Time of 90m descent: 180 s

- V_{BL} Breathing loop space: 4 litres