

# Bubble model based decompression algorithm optimised for implementation on a low power microcontroller

Benjamin Kuch and Giorgio Buttazzo  
*Scuola Superiore Sant'Anna – RETIS Lab, Pisa, Italy*

Arne Sieber  
*Imego, Göteborg, Sweden*

## Abstract

The calculation of a decompression schedule, according to the Varying Permeability Model (VPM) with Boyle's Law compensation extension, requires many sophisticated arithmetic operations. Therefore if it is calculated with a limited arithmetic instruction set on a microcontroller, a decompression schedule cannot be calculated in an acceptable time frame. This paper describes the principles behind an optimisation in calculation speed of the VPM with the Boyle's Law compensation extension for the determination of decompression schedules on a low power microcontroller. It was accomplished in three independent steps: converging the cubic root equation of the Boyle's Law compensation algorithm; using a set of predictive models to calculate the adapted bubble radius without using a cubic root solver; and pre-calculating the exponential terms of the Haldane and Schreiner equations, in order to reduce processing time and dynamic adjustment of the step size within the iterative process of the decompression schedule calculation. The modified algorithm was tested on an Atmel ATmega644P running at 8MHz. Calculating decompression schedules with these enhancements were approximately five times faster than with the original algorithm.

**Keywords:** decompression theory, Varying Permeability Model (VPM), Reduced Gradient Bubble Model (RGBM), bubble model

## 1. Introduction

During diving, the pressure that the body is exposed to is increased (10m of depth is equivalent to a pressure increase of 1bar) and inert gas, such as nitrogen and in the case of mixed-gas diving, helium, is taken up by the body (Henry's Law). These inert gases are normally stored in the form of physical solution in the tissues and liquids of the body.

During ascent, the ambient pressure decreases and the excess inert gases come out of solution, called 'offgassing'. Normally most offgassing occurs during exhalation, but if inert gas is forced to come out of solution too quickly, bubbles are formed inside the body which may lead to decompression illness (DCI) (Ehm et al., 2003). To avoid DCI during ascent, decompression stops are added at various depths, and the maximum ascent speed is typically limited to a value less than 10msw/min (Bühlmann, 1984). Decompression algorithms are used to calculate the depth and duration of these decompression stops to predict safe decompression schedules.

To be able to predict a decompression schedule, the inert gas absorption in the bodily tissue has to be modelled. For constant ambient pressure, this is typically done by the Haldane equation. For linearly varying ambient pressure, the inert gas absorption is calculated by the Schreiner equation (Baker, 2001).

Presently a wide range of decompression algorithms exist which can basically be classified as either Haldane or bubble models. Haldane decompression models (e.g. Workman, 1965, or Bühlmann, 2002), are mostly based on empirical models, which are described by mathematical functions. In contrast to Haldanian decompression models, which try to predict decompression schedules, so that no DCI occurs, bubble driven models simulate the physical behaviour of bubbles during pressure changes. Based on that, decompression schedules are calculated in order to limit the total volume of gas that is allowed to be present in the form of bubbles. The most well-known bubble models are the Reduced Gradient Bubble Model (RGBM) (Wienke, 1990, 2003) and the Varying Permeability Model (VPM) (Yount et al., 1999).

Although bubble models are seemingly popular and a multitude of PC decompression software are

based on them, only a few diving computers take advantage of them. Some recreational diving computer manufacturers claim to use RGBM in their products, but they actually use a folded version of the RGBM, which is in principle a Haldane model with modified M-values (Wienke and O’Leary, 2002). The reason for this might be that bubble driven models require much more computational processing power than Haldanian models, which standard dive computers do not currently possess. For example, one important aspect of a dive computer is battery lifetime, as a computer should be as power efficient as possible. The consequence of this efficiency is that these standard dive computers hold very little computational power.

The present work describes a way to simplify the VPM in order to allow quasi-real-time calculation of decompression schedules on power efficient dive computers.

## 2. Functionality of the VPM

The development of the VPM, ranging from theoretical background, descriptions of the basic functionality, to all the various adjustments, can be found in literature. This section gives an overview of the major functionalities of VPM that are particularly important for the further understanding of this paper, while a summary of the whole model can be found elsewhere (e.g. Watts, 2007).

VPM uses the same dissolved gas tissues as those found in Bühlmann’s ZH-L16 decompression model (Bühlmann, 1984). Sixteen hypothetical tissue compartments simulate the dissolved inert gas pressures of inert gases absorbed by the body during a dive and expelled during ascent. In addition to that traditional approach, VPM adds one ‘simulated bubble’ of each inert gas to each tissue compartment.

The simulated bubble has specific properties, the most important of which are the following:

- The bubble is assumed to have a specific size indicated by its initial radius at the beginning of a dive.
- Ordinarily the bubble is gas-permeable. Dissolved gases from the tissue compartment can move across the bubble’s skin and into the bubble.
- If too much pressure is applied to the bubble (e.g. as can occur during deep dives), the molecules of the bubble’s skin are squeezed together and the gas is no longer able to diffuse. As a consequence, the bubble’s skin becomes impermeable and the gas is trapped inside the bubble. The onset of impermeability of the bubble is dive-profile dependent and has to be calculated by

solving a cubic equation (Yount and Strauss, 1976; Yount, 1979; Yount and Hoffmann, 1986; Baker, 2000; Yount and Yeung, 1981).

The gas pressure inside the bubble ( $p_{bub}$ ) is calculated by Equation 1:

$$p_{bub} = p_{amb} + S/r \quad (1)$$

where  $p_{bub}$  is gas pressure inside the bubble,  $p_{amb}$  is ambient pressure,  $S$  is the properties of the bubble’s skin (a constant) and  $r$  is the bubble radius.

The bubble is surrounded by liquid holding various levels of dissolved gases. Dissolved gases in each tissue compartment diffuse across the skin into the bubble, thus inflating it if the compartment’s tissue tension ( $p_t$ ) exceeds the bubble pressure ( $p_{bub}$ ). The VPM bubble does not grow if the dissolved gas tension in a compartment is not allowed to exceed the pressure inside the bubble ( $p_t < p_{amb} + S/r$ ). Therefore the term  $p_t - p_{amb}$  defines the super saturation of a tissue compartment, which means that the dissolved gas tension exceeds the ambient pressure.

As an ascent strategy, VPM tracks the dissolved gases and bubble radii. During the ascent, the bubble pressure ( $p_{bub}$ ) is calculated continuously. The diver is stopped whenever a compartment’s tissue tension ( $p_t$ ) exceeds the compartment’s bubble pressure ( $p_{bub}$ ).

The resulting decompression schedule is very conservative, and the overall decompression time is quite long (especially for short dives). The human body can handle a certain volume of free gases. This hypothesis is defined in the critical volume algorithm (CVA) and permits the gas phase to inflate during decompression, under the constraint that the total volume of free gas never exceeds some critical value. The CVA defuses the compartment’s tissue tension limit in an iterative way. The decompression schedule calculation is repeated with the relaxed allowed tissue tensions until the calculated volume of free gas matches the pre-defined critical volume parameter ( $\lambda$ ) (Yount and Hoffmann, 1983).

The computed decompression schedules of the VPM (especially for schedules of deep dives) have shown that the VPM in the aforementioned form produces very aggressive decompression schedules in comparison to schedules produced using RGBM or Bühlmann with gradient factors methods. In 2002, Baker (2002) expanded the VPM by adding a Boyle’s Law compensation algorithm (VPM-B). The bubble radius is increased during ascent according to Boyle’s Law. The Boyle’s Law compensation algorithm introduces longer decompression stops, thus resulting in more conservative decompression schedules (Baker, 2002).

Computations within the VPM-B are calculated in an iterative way. A final decompression schedule is the product of several previous calculated decompression schedules with adjusted input parameters. Within the algorithm, several cubic root equations and exponential functions have to be solved to compute the final bubble radii and to solve the Schreiner and Haldane equations. This is very time consuming on a microcontroller with low processing performance. Thus the main objectives of this paper are to: converge the results of the cubic root equations; pre-solve the exponential functions; and reduce the amount of iterations within the algorithm to reduce overall processing time.

### 3. Methods

The VPM-B algorithm used in this work was, in general, a straight translation of Baker's original Fortran code (Baker, 2003) to Atmel AVR 8-bit microcontroller compatible C. For an integrated development environment (IDE), the IAR Embedded Workbench (IAR Systems) was selected, which is a set of development tools for developing embedded applications. It allows easy debugging and simulation, and provides compiler and stack configuration capabilities, which are useful when optimising the VPM-B for a low power microcontroller.

#### 3.1. Analysis

Several fictive dive profiles were calculated with the original VPM-B algorithm and analysed to assess processing time and to identify possible bottlenecks

in the code. The whole decompression calculation procedure was simulated and analysed with the IAR profiling and performance analysis tools in the debugger of the IAR Embedded Workbench. For the microcontroller, an Atmel ATmega644P 8-bit RISC microprocessor (4kB SRAM; 2kB EEPROM; up to 20MIPS throughput at 20MHz; power consumption at 1MHz, 0.4mA in active mode) running at 8MHz, was chosen.

Fig 1 presents all the functions, the amount of calls during one loop, the corresponding total accumulated runtime in CPU cycles and the percentage of a 70msw/20min dive with TX20/40 and nominal conservatism (standard VPM-B, no additional conservatism). Table 1 shows the corresponding decompression schedule. The dive was planned without decompression gas on purpose to keep it simple and easy to compare.

The analysis (Fig 1) showed clearly that the Boyle's Law compensation algorithm, which includes a root finder method to calculate the new adjusted bubble radii, was one of the biggest bottlenecks. The second bottleneck was the decompression stop function, which computes the decompression time for one decompression step (which includes the Haldane and Schreiner equations).

For an in-depth analysis of the VPM-B algorithm, it was translated to MATLAB R2008a (MathWorks). MATLAB is a numerical computing environment and programming language, which allows easy implementation of algorithms, programming schedules, matrix manipulation and plotting of functions and data.

Function	Calls	Accumulated Time (cycles)	Accumulated Time (%)
boyles_law_compensation	52	91053935	
calc_ascent_ceiling	3	138274	
calc_crushing_pressure	1	117687	
calc_deco_ceiling	430	19683749	
calc_initial_allowed_gradient	1	80662	
calc_max_actual_gradient	14	238154	
calc_start_of_deco_zone	1	4669745	
calc_surface_phase_volume_time	3	432719	
ceil	53	20999	
critical_volume	2	688613	
decompression_stop	52	122321326	
exp	16242	96958703	
gas_loadings_dynamic	58	17576841	
gas_loadings_static	1	216571	
haldane_equation	13792	97601314	
initialize	1	106007	
log	68	321201	
nuclear_regeneration	1	426690	
radius_root_finder	1664	80221807	
schreiner_equation	2336	21242422	
sqrt	64	209823	

**Fig 1:** Analysis of the VPM-B implementation for a 70msw/20min with TX20/40 dive, showing a list of methods, which are executed within the VPM-B during the calculation of a decompression schedule, and the workload of each method

**Table 1:** Decompression schedule calculated by VPM-B of a 70msw/20min with TX20/40 dive at nominal conservatism (no additional conservatism), implemented on a 8-bit RISC microprocessor

Depth (msw)	Time (min)
39	1
36	1
33	1
30	2
27	2
24	2
21	3
18	4
15	6
12	7
9	11
6	18
3	34

### 3.2. Converging Boyle's Law compensation algorithm

The VPM tries to simulate the bubble growth dependent on a specific dive profile. To calculate a decompression schedule, the gas pressure inside the bubble is relevant. To get this, the bubble radius has to be computed by solving a cubic root equation.

The Boyle's Law compensation algorithm is one of the most time consuming functions in the VPM-B, because it solves a cubic root equation to determine the new radius of a gas bubble due to the reduction in pressure between each decompression step. A cubic root equation is solved by convergence of the result in an iterative fashion. This is basically done by either the Newton-Raphson method, or the bisection method for cases where the Newton-Raphson method takes the solution out of bounds or does not converge fast enough (Press et al., 1992).

Since the VPM-B calculates 16 tissues for nitrogen and helium (and a decompression dive may have many decompression steps and many iterations during the CVA), this procedure is very time consuming. For example, in the 70msw/20min with TX20/40 dive example, the Boyle's Law compensation algorithm was called 52 times (first iteration + 2 more iterations within the CVA until it converged + final iteration to compute the decompression schedule =  $4 \times 13$  decompression steps). During these 52 calls, the cubic root equation was solved for two gas components (He and N<sub>2</sub>) and 16 tissues each ( $2 \times 16 \times 52 = 1664$  calls in total).

Converging the result of the root finder subroutine using a form of linear predictive models would accelerate this computation, because linear equations can be solved in a much faster way. The cubic root equation, which has to be solved to get

the final bubble radius, has the following form (Equation 2):

$$Ar^3 - Br^2 - C = 0 \quad (2)$$

Equation 2 describes the relation between compression energy and the radius of gas nucleus (Baker, 2000). A closer look at the  $A$ ,  $B$  and  $C$  values shows that only  $C$  is a dive profile dependent variable.  $A$  is the ambient pressure of the next scheduled decompression step in Pa. Since decompression steps are typically a multiple of three (3m, 6m, 9m, etc.),  $A$  can be calculated prior to a dive.  $B$  is constant and equal to  $-2 \times \gamma$  (surface tension of the bubble).  $C$  is the only parameter, which varies. It is defined in Equation 3:

$$C = (p_{amb} + (r/2\gamma)) \times r^3 \quad (3)$$

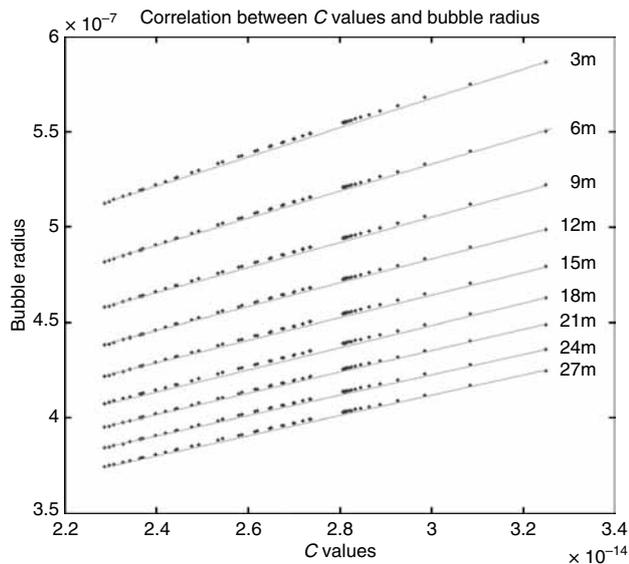
where  $p_{amb}$  is ambient pressure of the first decompression step in  $C$  (constant for each step),  $\gamma$  is surface tension of the bubble (constant) and  $r$  is the bubble radius at the first decompression step (dive profile dependant).

The final bubble radius is computed by a cubic root equation, where two of the three parameters are constant. Thus the first approach to improve the calculation speed of the Boyle's Law compensation algorithm was to determine the bubble radius by a predictive model, which described the relationship between the bubble radius and the  $C$  value for a predefined decompression step.

Plotting the  $C$  values with the corresponding bubble radii of a simulated dive in MATLAB, it can be seen that the values lie approximately on a slightly curved line for each decompression stop depth. In Fig 2, all  $C$  values and the corresponding bubble radii for nitrogen of a 50msw/30min/EAN21 dive were plotted using a scatter plot, which showed several lines. Each line corresponded to one decompression step. Since the first decompression step was at 27m for a 50msw/30min/EAN21 dive computed by the VPM-B at nominal conservatism, the graph showed nine different lines.

In a next step, all the  $C$  values and corresponding bubble radii of 1000 dive profiles were simulated and logged. Dive profiles were calculated between 30 and 100msw depth, 20 and 100min bottom time, different gas mixes and descent/ascent speeds. For the sake of simplicity, only one gas mixture per dive was used.

Since it purported that all  $C$  values and the corresponding bubble radii lay almost on a line, linear regression was used to fit a predictive model to the observed dataset of  $C$  and  $r$  values. Input parameter for the predictive model was  $C$  and the output value was  $r$ . Fitting was done using the least-squares method for a linear model provided in the curve



**Fig 2:** Correlation between C values and bubble radius at a 50msw/30min/EAN21 dive

fitting toolbox of MATLAB. (A detailed explanation of the least-squares method can be found in the MATLAB R2008a Product Help (Mathworks, 2008).

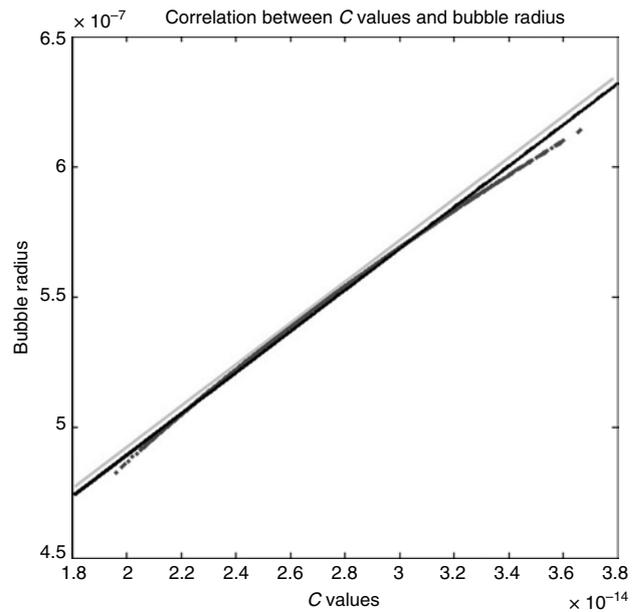
The correlation coefficient, which is a quantity that gives the quality of a least-squares fitting to the original data, was then calculated. For a perfect fit, the correlation coefficient is  $\pm 1$ . If there is no linear correlation or a weak linear correlation, the correlation coefficient is close to 0.

The fitted model was shifted in parallel to the maximum critical outlier of the dataset (the one which corresponds to the largest bubble radius). This guaranteed that this algorithm simplification was more conservative than the original Boyle's Law compensation algorithm, because it finally computed larger bubble radii and thus longer decompression times (Fig 3).

### 3.3. Pre-calculating exponential functions

The second bottleneck of the VPM-B was the decompression stop function, which computed the decompression time for one decompression step. As the decompression stop time cannot be calculated directly, an iterative approach was used to estimate the time of one stop and then the overall ascending time (time to surface). During this function the exponential function of the C library (<math.h>) was called several times (e.g. during a 70msw/20min with TX20/40 dive at nominal conservatism: time to surface (TTS)  $\times$  gases  $\times$  tissues  $\times$  iterations =  $\sim 16\ 242$  times) to solve the Haldane and Schreiner equations.

It calculated the tissue tension for the corresponding decompression step and the ascent between the decompression steps iteratively by



**Fig 3:** Correlation between C values and radii of 1000 simulated dives at the 3m decompression step (original data in dark grey, regression line in black and parallel shifted regression line to maximum critical outlier in light grey)

desaturating the tissues for  $n$  minutes, until a safe ascent to the next decompression step is possible. A typical iteration step is equal to 1min.

As an example, the Haldane equation is expressed in Equation 4:

$$p_t = p_0 + (p_i - p_0)(1 - e^{-kt}) \quad (4)$$

where  $p_t$  is compartment tissue tension (final);  $p_0$  is initial compartment inert gas pressure;  $p_i$  is inspired compartment inert gas pressure;  $t$  is time (of exposure or interval);  $k$  is time constant (in this case, half-time constant); and  $e$  is base of natural logarithms.

Solving an exponential function on a low power 8-bit microcontroller requires many CPU cycles, which is time consuming. As an example, the ATmega644P operating at 8MHz needed 5250 CPU cycles ( $\sim 0.67$ ms) to solve one single exponential function (IAR C/C++ Compiler for AVR 5.11B).

The second approach to reduce the processing time for calculating a VPM-B decompression schedule was to pre-calculate the exponential terms of the Haldane equation. This was possible because during the iterative calculation, the step size was constant and equal to 1min. Pre-calculated terms were stored in the flash memory of the microcontroller and thus did not need to be calculated during program execution. This technique is often referred to as dynamic programming (Bellman, 1957). The same pre-calculated terms could be used for solving the Schreiner equation within the decompression area if the ascent speed is constant.

### 3.4. Dynamic step size adjustment

A decompression schedule is calculated in several iterations. In the original Baker VPM-B code, the step size of the decompression time per iteration is freely selectable but constant. A high step size value leads to a low resolution, and conversely a small step size value leads to a high resolution of the approximated decompression schedule. The typical step size is 1min, but the iterative calculation takes a long time, especially in the case of long decompression schedules. The third approach used in this paper was to accelerate the calculation by dynamically increasing the step size (decompression time per iteration) in relation to the total decompression time. This approach allowed using a reduced amount of iterations for the calculation of a decompression schedule.

For a diver, the most important parameter is the ceiling depth that has to be observed, as this is the depth to which the diver can safely ascend according to the model. The iterations with variable step sizes are only used to predict the TTS. The variable step size approach leads to slightly longer TTS predictions (maximum of 10% longer). It is important to understand that this modification of the algorithm (with variable step sizes) does not influence the ceiling calculation, as this is independent of the step size. Thus, the third idea was to increase the step size dynamically according to the overall TTS. A typical value of the step size is about 10% of the overall TTS (see Table 2).

The advantage of increasing the step size of the decompression time is that the decompression stop time can be computed in a lesser amount of iterations, thus reducing the computational time.

## 4. Results

The optimised VPM-B algorithm was verified in several ways. Firstly, the fitted linear least-square models were reviewed by their corresponding correlation coefficients, which are quantities that give the quality of a least-squares fitting to the original data. For a perfect fit, the correlation coefficient is

**Table 2:** Increase of the decompression step size according to the decompression time; short decompression times are calculated in small steps and long decompression times are calculated in big steps

Decompression time (min)	Step size resolution (min)
1–10	1
11–20	2
21–30	3
>30	5

$\pm 1$ . If there is no linear correlation or a weak one, the correlation coefficient is close to 0. In this paper no calculated correlation coefficient fell below 0.99.

Secondly, the decompression schedules computed by the VPM-B algorithms with Boyle's Law compensation convergence were compared to regular VPM-B profiles. All simulated dives resulted in similar decompression schedules. Table 3 shows the mean deviation, standard deviation and range of 50 simulated dive profiles between the VPM-B algorithms with converged Boyle's Law compensation algorithm and the original VPM-B algorithm.

Table 4 shows the TTS differences between the optimised VPM-B algorithm (with all changes explained earlier) and the original VPM-B algorithm. Since the optimised VPM-B algorithm computes slightly bigger bubble radii during the Boyle's Law compensation algorithm and the step size during the TTS calculation is dynamically adjusted to a bigger value, the optimised algorithm consequently computes longer runtimes.

Table 5 illustrates the speed improvements versus dive runtime changes according to the example of a 70msw/20min with TX20/40 dive with nominal conservatism (without decompression gas). All improvements are switched on successively.

## 5. Discussion

The introduced modifications of the VPM-B decompression model demonstrated an approach for calculating the VPM-B model on a 8-bit low power

**Table 3:** VPM-B with converged Boyle's Law compensation compared to the original VPM-B

Decompression steps	Mean deviation (VPM-B Boyle's Law compensation converged – VPM-B) in min	Standard deviation in min	Range
Total TTS	1.14	1.18	+5/0
3msw	0.48	0.84	+3/–1
6msw	0.35	0.64	+2/–1
9msw	0.07	0.68	+1/–1
12msw	0.08	0.54	+1/–1
15msw	0.09	0.38	+1/–1
18msw	0.16	0.37	+1/–1
21msw	0.03	0.33	+1/–1
24msw	0.04	0.20	+1/–1
27msw	–0.05	0.22	+1/–1
30msw	0.05	0.22	+1/–1
33msw	–0.06	0.25	+1/–1
36msw	0.10	0.32	+1/0
39msw	0.00	0.00	0
42msw	0.00	0.00	0
45msw	0.00	0.00	0
48msw	0.00	0.00	0

**Table 4:** Optimised VPM algorithm compared to VPM-B

Depth (msw)	Time (min)	Mix	Descent (m/min)	Ascent (m/min)	Conservatism level	TTS VPM-B (min)	TTS converged VPM-B (min)
55	15	EAN21	18	9	0	58	63
50	20	EAN21	18	9	0	71	75
45	30	EAN21	18	9	0	99	100
40	40	EAN21	18	9	0	118	120
35	60	EAN21	18	9	0	153	155
70	15	TX19/40	25	7	1	99	105
65	20	TX19/40	25	7	1	125	130
60	20	TX19/40	25	7	1	112	115
55	25	TX19/40	25	7	1	126	130
50	15	TX19/40	25	7	1	61	63
40	45	EAN32	15	10	2	94	96
35	45	EAN32	15	10	2	81	82
30	60	EAN32	15	10	2	94	96
28	45	EAN32	15	10	2	62	63
25	75	EAN32	15	10	2	101	101
90	10	TX17/40	30	8	1	101	110
85	15	TX17/40	30	8	1	145	150
80	17	TX17/40	30	8	1	158	165
75	20	TX17/40	30	8	1	168	175
70	23	TX17/40	30	8	1	178	185

**Table 5:** Speed improvements versus dive runtime changes for a 70m/20min with TX20/40 dive at nominal conservatism

Algorithm	Number of CPU cycles	Calculation time at 8Mhz (sec)	Dive runtime (min)
VPM-B	238 214 043	29.78	115
VPM-B + dynamic programming (DP)	142 214 320	17.78	115
VPM-B + DP + Boyle's Law compensation convergence (BLCC)	65 363 296	8.17	116
VPM-B + DP + BLCC + dynamic deco step size	49 225 972	6.15	120

microcontroller in quasi real time. This paper has explained a way to replace the time intensive root finder method within the Boyle's Law compensation algorithm by a set of predictive models.

In this paper, the process of creating a lookup table of predictive models for dives between 30 and 100msw with a bottom time between 20 and 100min has been explained (outside that range, the original algorithm has to be used). The correlations between *C* values and bubble radii for each decompression step and inert gas were projected in linear equations, which were created by linear regression. Additionally, the conservatism was improved by parallel shifting these linear equations to the biggest critical outlier in the dataset, in order to be consistently more conservative than the original decompression algorithm. The modifications concern only the Boyle's Law compensation algorithm, which calculates the bubble growth for one decompression step to another.

Furthermore, dynamic adjustment of the step size within the iterative process of the decompression

schedule calculation was added to the optimised VPM-B algorithm to save additional computation time. It is important to clarify that this adjustment is only applied to predict the TTS. The tissue saturation is still calculated in normal intervals (i.e. every second), so that a later TTS calculation – when the decompression time converges to 0 during the ascent – will produce again a more accurate TTS (in sequential order: long TTS; big step size; inexact decompression schedule, but more conservative/short TTS; small step size; more exact decompression schedule).

Finally, it is essential that the modifications described in this paper should not be seen as an improvement of the VPM-B. It is simply a way to approximate intermediate results so that decompression schedules can be calculated quickly, but with similar output to the original algorithm, as compared to the folded RGBM, which is computed like a Haldanian decompression model, but produces decompression schedules similar to RGBM (Wienke and O'Leary, 2002).

## 6. Conclusion

This paper introduces a new speed optimised approach for real-time calculation of VPM-B decompression schedules addressing diving computers featuring a low power microcontroller. It was accomplished in three independent steps, first by converging the cubic root equation of the Boyle's Law compensation algorithm with a set of predictive models to calculate the adapted bubble radius without using a cubic root solver. Additionally, pre-calculation of the exponential terms of the Haldane and Schreiner equations was done, in order to reduce processing time and dynamic adjustment of the step size within the iterative process of the decompression schedule calculation. The modified algorithm was tested on an Atmel ATmega644P, running at 8MHz. Calculating decompression schedules with this enhancement was approximately five times faster than with the original algorithm.

## Acknowledgement

The authors would like to thank Dr Ranjana Sahai for correcting this paper.

## References

- Baker EC. (2000). VPM: Solving for radius in the impermeable regime. Available at ftp.decompression.org, accessed on 12 December 2010.
- Baker EC. (2001). Some introductory 'lessons' about dissolved gas decompression modelling. Available at ftp.decompression.org, accessed on 12 December 2010.
- Baker EC. (2002). VPM-B program update explanation. Available at ftp.decompression.org, accessed on 12 December 2010.
- Baker EC. (2003). VPM-B Fortran source code. Available at ftp.decompression.org, accessed on 12 December 2010.
- Bellman R. (1957). *Dynamic Programming*. Princeton, NJ: Princeton University Press, 342 pp.
- Bühlmann AA. (1984). *Dekompression-Dekompression Krankheit*. Springer-Verlag.
- Bühlmann AA. (2002). *Tauchmedizin*. Springer-Verlag.
- Ehm OF, Hahn M, Hoffmann U and Wenzel J. (2003). *Tauchennochsicherer – Tauchmedizin fuer Freizeittaucher, Berufstaucher und Aerzte*. Mueller Rueschlikon Verlag AG.
- MathWorks Inc. (2008). MATLAB R2008a Product Help. Natick, MA: MathWorks.
- Press WH, Flannery BP, Teukolsky SA and Vetterling WT. (1992). *Numerical Recipes in Fortran 77: The Art of Scientific Computing*. Cambridge: Cambridge University Press, 933 pp.
- Watts K. (2007). VPM for dummies. Available at rebreather-world.com/general-and-new-to-rebreather-articles/10721-vpm-for-dummies.html, accessed on 12 December 2010.
- Wienke BR. (1990). Reduced gradient bubble model. *International Journal of Bio-Medical Computing* **26**: 237–256.
- Wienke BR. (2003). *Reduced Gradient Bubble Model in Depth*. Flagstaff, AZ: Best Publishing, 96 pp.
- Wienke BR and O'Leary TR. (2002). Deep stops and deep helium. RGBM Technical Series 9. NAUI Technical Diving Operations, Tampa, Florida. Available at www.tek-dive.com/portal/upload/deep.pdf, accessed on 22 January 2011.
- Workman RD. (1965). Calculation of decompression schedules for nitrogen-oxygen and helium-oxygen dives. Interim Report Research Report 6-65. Washington, DC: U.S. Navy Experimental Diving Unit, 40 pp.
- Yount DE. (1979). Skins of varying permeability: A stabilization mechanism for gas cavitation nuclei. *Journal of the Acoustical Society of America* **65**: 1429–1439.
- Yount DE and Hoffman DC. (1983). Decompression theory: A dynamic critical-volume hypothesis. In: Bachrach AJ and Matzen MM. (eds.). *Underwater Physiology VIII: Proceedings of the Eighth Symposium on Underwater Physiology*. Undersea Medical Society, Bethesda, 131–146.
- Yount DE and Hoffman DC. (1986). On the use of a bubble formation model to calculate diving tables. *Aviation, Space and Environmental Medicine* **57**: 149–156.
- Yount DE and Strauss RH. (1976). Bubble formation in gelatin: A model for decompression sickness. *Journal of Applied Physics* **47**: 5081–5089.
- Yount DE and Yeung CM. (1981). Bubble formation in supersaturated gelatin: A further investigation of gas cavitation nuclei. *Journal of the Acoustical Society of America* **69**: 702–708.
- Yount DE, Baker EC and Maiken EB. (1999). Implications of the varying permeability model for reverse dive profiles. Presented at the Reverse Dive Profiles Workshop, 29–30 October 1999, Smithsonian Institution, Washington, D.C.