Table-Driven Modeling in VHDL-AMS - Summary

General approach

Given table-data are prepared by a function PREPARE_TABLE_DATA for the real-valued one- and multi-dimensional and the complex-valued case resp. The data are prepared for interpolation between the given knots. The preparation is done regarding interpolation and extrapolation methods.

For an efficient evaluation, the preparation of the data is done once. Afterwards, the interpolation function INTERPOLATION can be called several times. The function INTERPOLATION delivers a value for the current argument.

Depending on the types of the arguments different version of the functions PREPARE_TABLE_DATA and INTERPOLATION are provided in a package TABLE_INTERPOLATION_PKG. The package also provides declaration of the enumeration types INTERPOLATION_METHOD and EXTRAPOLATION_METHOD to describe the applied methods.

One-dimensional real-valued interpolation

The knot data are represented by the arrays[1] X and Y that characterize the independent and dependent data points resp. One data point is given by a tupel (X(index), Y(index)). The function PREPARE_TABLE_DATA prepares the data for an efficient evaluation.

```
function PREPARE_TABLE_DATA (
   X      : REAL_VECTOR;
   Y      : REAL_VECTOR;
   METHOD : INTERPOLATION_METHOD := TDM_PWL;
   LEFT   : EXTRAPOLATION_METHOD := TDM_LINEAR;
   RIGHT  : EXTRAPOLATION_METHOD := TDM_LINEAR)
return TABLE_DATA_REAL;
```

The function orders the given knot data. The METHOD parameter describes the interpolation method. The LEFT and RIGHT parameters describe the extrapolation for arguments that are less than the smallest or greater than the greatest the greatest X value.

The

```
function INTERPOLATION    (
   XV     : REAL;
   T      : TABLE_DATA_REAL)
return REAL;
```

determines a value based on the table-data T for the argument XV.

Algorithmic details for one-dimensional real-valued interpolation

The interpolation is based on piecewise polynomial description between the knots. It is assumed that the knots for the interpolation are described by the independent values

$$x_0 < x_1 < x_2 < \ldots < x_{m-1} < x_m$$

and the associated real dependent values

$$y_0 \quad y_1 \quad y_2 \quad \cdots \quad y_{m-1} \quad y_m \quad .$$

Interpolation is done for arguments between x_0 and x_m.

[1] Arrays are organized in ascending order.

Table 1: Description of interpolation methods

Interpolation method	Description	Comment
TDM_CLOSEST_POINT	Closest point lookup	$f(x) = \begin{cases} y_{i-1} & for \quad x_{i-1} \le x < \frac{x_{i-1}+x_i}{2} \\ y_i & for \quad \frac{x_{i-1}+x_i}{2} \le x < x_i \\ y_m & for \quad x = x_m \end{cases}$
TDM_STEPWISE_CONSTANT	Step-wise constant lookup	$f(x) = \begin{cases} y_{i-1} & for \quad x_{i-1} \le x < x_i \\ y_m & for \quad x = x_m \end{cases}$
TDM_PWL	Linear interpolation	$f(x) = \begin{cases} y_{i-1} + (x - x_i) \cdot \frac{y_i - y_{i-1}}{x_i - x_{i-1}} & for \quad x_{i-1} \le x < x_i \\ y_m & for \quad x = x_m \end{cases}$
TDM_QUADRATIC	Quadratic interpolation	$f(x) = y_{i-1} + a1(i) \cdot (x - x_{i-1}) + a2(i) \cdot (x - x_{i-1})^2$ $for \quad x_{i-1} \le x < x_i$ where the a1 und a2 are determined in such a way that the function values and the first order derivatives at inner knots are continuous and $f(x) = y_m \quad for \quad x = x_m$
TDM_CUBIC	Cubic interpolation	$f(x) = y_{i-1} + a1(i) \cdot (x - x_{i-1}) + a2(i) \cdot (x - x_{i-1})^2 + a3(i) \cdot (x - x_{i-1})^3$ $for \quad x_{i-1} \le x < x_i$ where the a1, a2 and a3 are determined in such a way that the function values and the first order and second order derivatives at inner knots are continuous and $f(x) = y_m \quad for \quad x = x_m$

The extrapolation methods describe the behaviour in the case of range violations. Different methods can be applied for extrapolation to the left ($x < x_0$) and to the right ($x_m < x$). The following table summarizes the behaviour for the different methods.

Table 2: Description of extrapolation methods

Extrapolation method	Description	Comment
TDM_CONSTANT	Constant extrapolation	$f(x) = \begin{cases} y_0 & for \quad x < x_0 \\ y_m & for \quad x > x_m \end{cases}$
TDM_LINEAR	Linear extrapolation	If the interpolation method is TDM_CLOSEST_POINT or TDM_STEPWISE_CONSTANT the first and last value resp. will be used $f(x) = \begin{cases} y_0 & for \quad x < x_0 \\ y_m & for \quad x > x_m \end{cases}$ In the other cases the slope (first order derivative) at the first and last point of the first and last interpolation interval resp. will be used to

		characterize a line that goes through the fist and last point resp. Thus, it will be used $f(x) = \begin{cases} y_0 + f'(x_0) \cdot (x - x_0) & for \quad x < x_0 \\ y_m + f'(x_m) \cdot (x - x_0) & for \quad x > x_m \end{cases}$
TDM_ERROR	Error	An error is detected for $x < x_0$ and $x_m < x$ resp.
TDM_EXTRAP	The specified interpolation algorithm is used for out of range extrapolation	If the interpolation method is TDM_CLOSEST_POINT or TDM_STEPWISE-CONSTANT the first and last value resp. will be used $f(x) = \begin{cases} y_0 & for \quad x < x_0 \\ y_m & for \quad x > x_m \end{cases}$ In the other cases let q_1 and q_m be the functions that describe the behaviour in the first and last interpolation interval. That means $f(x) = \begin{cases} q_1(x) & for \quad x < x_0 \\ q_m(x) & for \quad x > x_m \end{cases}$
TDM_PERIODIC	Periodic extrapolation	The argument is transformed into the interpolation range and the function value is determined based on the interpolation method $f(x) = f\left(x_0 + (x - x_0) \bmod (x_m - x_0)\right)$

Quadratic spline interpolation

To determine the coefficients of the polynomials that describe the interpolation as shown in Table 1 one additional condition is required. This condition determines the first order derivative $f'(x_0) = a1(1)$ at the first point. The condition depends on the method that describes the extrapolation to the left. A special requirement is derived if one of the extrapolation methods is periodic.

The additional condition leads to the conclusion that the first order derivative at the first point is continuous.

Table 3: Determination of additional conditions for quadratic spline interpolation

Extrapolation method	Description	Comment
TDM_PERIODIC	Extrapolation to the left and/or to the right is periodic	The coefficients of the polynomial are determined in such a way that the first order derivatives at the first and last interpolation point are the same. That means $f'(x_0) = f'(x_m)$
TDM_CONSTANT (equals LEFT)	Constant extrapolation to the left	The first order derivative of the first point of the first interpolation interval is set to zero $f'(x_0) = 0$

TDM_LINEAR, TM_ EXTRAP, TDM_ERROR	All other cases	It is required that the second order derivative at the first point is zero. Thus, the derivative is determined by the slope of the line between the first two points $f'(x_0) = \frac{y_1 - y_0}{x_1 - x_0}$

Cubic spline interpolation

To determine the coefficients of the polynomials that describe the interpolation as shown in Table 1 two additional conditions are required. The conditions depend on the extrapolation methods. The conditions shall assure that first and second order derivatives at the first and last data point are continuous. Table 4 summarizes the conditions.

Table 4: Determination of additional conditions for cubic spline interpolation

Abbreviation for the method	Description	Comment
TDM_PERIODIC	Extrapolation to the left and/or to the right is periodic	The coefficients of the polynomial are determined in such a way that the first and second order derivatives at the first and last interpolation point are the same. That means $f'(x_0) = f'(x_m)$ $f''(x_0) = f''(x_m)$
TDM_CONSTANT	Constant extrapolation	If the method is required for extrapolation to the left the first order derivative of the first point of the first interpolation interval is set to zero $f'(x_0) = 0$ If the method is required for extrapolation to the right the first order derivative of the last point of the last interpolation interval is set to zero $f'(x_m) = 0$
TDM_LINEAR	Linear extrapolation	If the method is required for extrapolation to the left the second order derivative of the first point of the first interpolation interval is set to zero $f''(x_0) = 0$ If the method is required for extrapolation to the right the second order derivative of the last point of the last interpolation interval is set to zero $f''(x_m) = 0$
TDM_EXTRAP, TDM_ERROR	Other methods	The corresponding condition for linear extrapolation is used.

Insufficient number of data points in the case of one-dimensional real-valued interpolation

a. Only one data point is given

If only one data point is given (*m=0*) we get as result in all cases expect one or both of the extrapolation methods are TDM_ERROR $f(x) = y_0 \quad for \quad x = x_0$.

b. Quadratic interpolation and less than three data points

If quadratic spline interpolation is required and less than three data points are given linear interpolation is used. If quadratic spline interpolation is required and one of the extrapolation methods is TDM_PERIODIC an even number of data points is required. If in this case the number of data points is odd and greater three it is tried to carry out cubic spline interpolation. Otherwise, an error is reported.

c. Cubic interpolation and less than four data points

If cubic spline interpolation is required and less than four data points are given quadratic spline interpolation is used.

One-dimensional complex-valued interpolation

The knot data are represented by the arrays X and Y that characterize the independent and dependent data points resp. One data point is given by a tupel (X(index), Y(index)). The function PREPARE_TABLE_DATA prepares the data for an efficient evaluation.

```
function PREPARE_TABLE_DATA (
   X      : REAL_VECTOR;
   Y      : COMPLEX_VECTOR;
   METHOD : INTERPOLATION_METHOD := TDM_PWL;
   LEFT   : EXTRAPOLATION_METHOD := TDM_LINEAR;
   RIGHT  : EXTRAPOLATION_METHOD := TDM_LINEAR)
return TABLE_DATA_COMPLEX;
```

The function orders the given knot data. The METHOD parameter describes the interpolation method. In addition to the methods given by Table 1 rational interpolation TDM_RATIONAL is supported. The LEFT and RIGHT parameters describe the extrapolation for arguments that are less than the smallest or greater than the greatest the greatest X value.

The

```
function INTERPOLATION    (
   XV     : REAL;
   T      : TABLE_DATA_COMPLEX)
return REAL;
```

determines a value based on the table-data T for the argument XV.

Algorithmic details for one-dimensional complex-valued interpolation

Interpolation shall be done for given points (x_i, y_i) . The knots for the interpolation are described by the independent values $x_0 < x_1 < x_2 < \ldots < x_{m-1} < x_m$

and the associated complex dependent values $y_0 \quad y_1 \quad y_2 \quad \cdots \quad y_{m-1} \quad y_m$.

In this way, frequency characteristics can be given by data points where the independent values characterize the frequencies and the dependent values the associated complex values. The complex values are given by their real and imaginary parts.

For the interpolation methods given by Table 1, real and imaginary part are independently interpolated using the same interpolation and extrapolation method for the real and imaginary part. That means, the interpolation task is reduced to two real-valued one-dimensional interpolation tasks.

Furthermore, rational interpolation shall be provided if the interpolation method TDM_RATIONAL is used. In this case, real and imaginary parts of the given complex values are simultaneously considered. The rational interpolation consists of the representation of the given values as a quotient of two polynomials:

$$f(x_v) = \frac{a_0 + a_1 \cdot x_v + \ldots + a_p \cdot {x_v}^p}{b_0 + b_1 \cdot x_v + \ldots + b_q \cdot {x_v}^q} \qquad (1)$$

It is recommended to apply the Stoer Bulirsch algorithm for rational interpolation. The Stoer Bulirsch algorithm is a recursive tabular technique aimed at (only) determining the value of f for a given argument x_v. Thus, the coefficients a_i, b_j are not determined by this algorithm. For an odd number of given data points the algorithm evaluates a fractional rational function with $p = q$. For an even number of data points the degree of the denominator q is larger by one than p. In the case of rational interpolation only the extrapolation methods TDM_EXTRAP (extrapolation using the function for the first and last interpolation interval resp.) and TDM_ERROR are supported.

Special conditions as passivity, causality, and stability of a transfer function corresponding to the interpolation function are not checked.

Multi-dimensional real-valued interpolation

The knot data are represented by the matrix X and the array Y that characterize the independent and dependent data points resp. One data point is given by a (n+1)-tupel (X(index, 0), X(index,1), X(index, n-1), Y(index)). The function PREPARE_TABLE_DATA prepares the data for an efficient evaluation.

```
function PREPARE_TABLE_DATA (
   X      : REAL_MATRIX;
   Y      : REAL_MATRIX;
   METHOD : INTERPOLATION_METHOD_VECTOR;
   LEFT   : EXTRAPOLATION_METHOD_VECTOR;
   RIGHT  : EXTRAPOLATION_METHOD_VECTOR)
return TABLE_DATA_REAL;
```

The function orders the given knot data. The METHOD parameter describes the interpolation method for each dimension given by X. The LEFT and RIGHT parameters describe the extrapolation for arguments that are less than the smallest or greater than the greatest the greatest X value for each dimension given by X. The

```
function INTERPOLATION    (
   XV     : REAL_VECTOR;
   T      : TABLE_DATA_REAL)
return REAL;
```

determines a value based on the table-data T for the argument XV.

Algorithmic details for multi-dimensional real-valued interpolation

The REAL_MATRIX is declared as

```
type REAL_MATRIX is array (NATURAL range <>, NATURAL range <>) of REAL;
```

Let X be of type REAL_MATRIX. Then, X'LEFT(2) is called the outermost dimension. X'RIGHT(2) is the innermost dimension.[2]

Starting with the innermost dimension one-dimensional interpolations are carried out to determine real values that are used for the interpolation regarding the next dimension. The process ends at the outermost dimension. Interpolation and extrapolation methods for each dimension are given by the associated values of the METHOD, LEFT, and RIGHT arrays.

The interpolation for an argument `XV_ND` of type `REAL_VECTOR (0 to N-1)` shall be determined. Thus, the multi-dimensional interpolation can be carried out in the subsequent manner

```
Declare Y of type REAL_VECTOR (0 to M_OUTER)
IDY = 0
for I in 0 to N_INDEX_INNER loop
    Y(IDY) is given by one-dimensional INTERPOLATION based on
           X_INNER (START_INDEX_INNER(I) to END_INDEX_INNER(I)) and
           A_INNER (START_INDEX_INNER(I) to END_INDEX_INNER(I), 0 to ORDER)
                   where ORDER depends on METHOD(N-1)
           for XV=XV_ND(N-1)
           with METHOD(N-1), LEFT(N-1) and RIGHT(N-1)
    IDY = IDY + 1
end loop

for I in 0 to M_INDEX_OUTER loop
    Y(IDY) is given by one-dimensional INTERPOLATION based on
           X_OUTER (START_INDEX_OUTER(I) to END_INDEX_OUTER(I)) and
           Y       (START_INDEX_OUTER(I) to END_INDEX_OUTER(I))
           for XV=XV_ND(DIMENSION_INDEX_OUTER(I))
           with METHOD, LEFT and RIGHT
                for index DIMENSION_INDEX_OUTER(I)

    IDY = IDY + 1
end loop
```

The result of the multi-dimensional interpolation is the result `Y(IDY-1)` of the last one-dimensional interpolation.

References

[1] Dahlquist, G.; Björck, A.: Numerical Methods in Scientific Computing:Volume I. Philadelphia: siam, 2008.

[2] J. Stoer and R. Bulirsch: Introduction to Numerical Analysis. Berlin: Springer, 2002. (first edition 1980) (see section 2.2.3 – formula (2.2.3.8))

Ha./2. 5. 2011

[2] It is suggested that the fastest chanching of independent values occurs for the innermost dimension.

Appendix

Package declaration TABLE_INTERPOLATION_PKG based on VHDL-2008

```
library IEEE;
use IEEE.MATH_REAL.all;
use IEEE.MATH_COMPLEX.all;


package TABLE_INTERPOLATION_PKG is


   type COMPLEX_VECTOR is array (NATURAL range <>) of COMPLEX;
   type REAL_MATRIX is array (NATURAL range <>, NATURAL range <>) of REAL;

   type INTERPOLATION_METHOD is
           (TDM_PWL,       TDM_CLOSEST_POINT, TDM_STEPWISE_CONSTANT,
            TDM_QUADRATIC, TDM_CUBIC,         TDM_RATIONAL);

   type EXTRAPOLATION_METHOD is
           (TDM_LINEAR, TDM_CONSTANT, TDM_ERROR, TDM_EXTRAP, TDM_PERIODIC);

   type INTERPOLATION_METHOD_VECTOR is array (NATURAL range <>)
           of INTERPOLATION_METHOD;

   type EXTRAPOLATION_METHOD_VECTOR is array (NATURAL range <>)
           of EXTRAPOLATION_METHOD;


   type TABLE_DATA_REAL is
   record
     X_INNER               : REAL_VECTOR;
     A_INNER               : REAL_MATRIX;
     START_INDEX_INNER     : INTEGER_VECTOR;
     END_INDEX_INNER       : INTEGER_VECTOR;
     X_OUTER               : REAL_VECTOR;
     START_INDEX_OUTER     : INTEGER_VECTOR;
     END_INDEX_OUTER       : INTEGER_VECTOR;
     DIMENSION_INDEX_OUTER : INTEGER_VECTOR;
     METHOD                : INTERPOLATION_METHOD_VECTOR;
     LEFT                  : EXTRAPOLATION_METHOD_VECTOR;
     RIGHT                 : EXTRAPOLATION_METHOD_VECTOR;
   end record;


   type TABLE_DATA_COMPLEX is
   record
     X       : REAL_VECTOR;
     y       : COMPLEX_VECTOR;
     A_RE    : REAL_MATRIX;
     A_IM    : REAL_MATRIX;
     METHOD  : INTERPOLATION_METHOD;
     LEFT    : EXTRAPOLATION_METHOD;
     RIGHT   : EXTRAPOLATION_METHOD;
   end record;
```

```
    function PREPARE_TABLE_DATA (
                               X      : REAL_VECTOR;
                               Y      : REAL_VECTOR;
                               METHOD : INTERPOLATION_METHOD := TDM_PWL;
                               LEFT   : EXTRAPOLATION_METHOD := TDM_LINEAR;
                               RIGHT  : EXTRAPOLATION_METHOD := TDM_LINEAR)
    return TABLE_DATA_REAL;

    function PREPARE_TABLE_DATA  (
                               X      : REAL_VECTOR;
                               Y      : COMPLEX_VECTOR;
                               METHOD : INTERPOLATION_METHOD := TDM_PWL;
                               LEFT   : EXTRAPOLATION_METHOD := TDM_ERROR;
                               RIGHT  : EXTRAPOLATION_METHOD := TDM_ERROR)
    return TABLE_DATA_COMPLEX;


    function PREPARE_TABLE_DATA (
                               X      : REAL_MATRIX;
                               Y      : REAL_VECTOR;
                                METHOD : INTERPOLATION_METHOD_VECTOR;
                                LEFT   : EXTRAPOLATION_METHOD_VECTOR;
                               RIGHT  : EXTRAPOLATION_METHOD_VECTOR)
    return TABLE_DATA_REAL;


    function INTERPOLATION    (XV     : REAL;
                               T      : TABLE_DATA_REAL)

    return REAL;


    function INTERPOLATION    (XV     : REAL;
                               T      : TABLE_DATA_COMPLEX)
    return COMPLEX;


    function INTERPOLATION    (XV     : REAL_VECTOR;
                               T      : TABLE_DATA_REAL)
    return REAL;

end package TABLE_INTERPOLATION_PKG;
```