**Did You Know – These special user functions can be used in CyFlex computed expressions.**

Functions may be called in expressions using an "at" sign (@) followed by the name of the function, for example:

@dpt\_to\_vp( dewpoint\_temp )

The above function returns the water vapor pressure corresponding to the dewpoint temperature.

*Functions Supported in Expressions – This list can also be found in the CyFlex User Manual – “CyFlex Variables, Units, and Computed Expressions”*

| **Function** | **Description** |
| --- | --- |
| long\_3d\_comp(X\_VAR, TABLE\_NUM) | 3D long interpolation  Example:  @long\_3d\_comp( 1000[rpm], 5[none] )  this would use table file /cell/tables/loc\_two\_d\_5.tbl |
| long\_3d\_comp\_name( double value, char \*table\_name ) | 3d long interpolation test, using the table name  Example:  @long\_3d\_comp\_name( 1000[rpm], ‘loc\_two\_d\_5’ )  this would use table file /cell/tables/loc\_two\_d\_5.tbl |
| shrt\_3d\_comp(X\_VAR, TABLE\_NUM) | 3D short interpolation |
| long second\_of\_minuteL() | returns current "seconds" of time as an integer |
| char\* second\_of\_minute () | returns current "seconds" of time as a string with leading zeroes |
| long minute\_of\_hourL() | returns current "minutes" of time as an integer |
| char\* minute\_of\_hour() | returns current "minutes" of time as a string with leading zeroes |
| long day\_of\_month() | returns current "day" of month |
| long day\_of\_week() | returns current "day" of week |
| long day\_of\_year() | returns current "day" of year |
| long hour\_of\_dayL() | returns current "hour" of day (24 hour clock) as an integer |
| char\* hour\_of\_day() | returns current "hour" of day (24 hour clock) as a string with leading zeroes |
| long year\_month\_dayL() | returns an integer of format YYMMDD, e.g. 110621 |
| long day\_month\_yearL() | returns an integer of format DDMMYY, e.g. 210611 |
| char\* year\_month\_day() | returns string of format YYMMDD, e.g. “110621” |
| char\* day\_month\_year() | returns string of format DDMMYY with leading zeroes, e.g. “210611” |
| long year4\_month\_dayL() | returns an integer of format YYYYMMDD, e.g. 20110621 |
| char\* year4\_month\_day() | returns string of format YYYYMMDD with leading zeroes, e.g. “20110621” |
| char\* week\_of\_year() | returns string of format NN, where NN is the current week of the year numbered from 1 to 52, e.g. “47” |
| long week\_of\_yearL() | returns an integer of format NN, where NN is the current week of the year numbered from 1 to 52, e.g. 9 |
| double rh\_visc\_rat( double temp, double rh ) | determine viscosity ratio for air as a function of temperature and relative humidity (in %) |
| double kvisc( long code , double t ) | compute the kinematic viscosity (stokes)of lube or fuel oil at a given temperature (t)  code type  1 #1 fuel oil  2 #2 fuel oil  3 15W40 lube oil  4 10W lube oil  5 30W lube oil |
| double dpt\_to\_vp( temp ) | returns vapor pressure as function of dewpoint  Example:  @dpt\_to\_vp( outside\_temp )  @dpt\_to\_vp( 68[deg\_f] ) returns 2336 pascals |
| double cal\_table( x-value, table\_name ) | returns interpolated value from a calibration table  Examples:  @cal\_table( 100[mv], ‘cmp\_in\_p’ )  @cal\_table( x\_val, ‘cmp\_in\_p’ ) |
| char ascii\_time( time ) | returns MM/DD/YY HH:MM:SS format from a double value of ANSI time – time may be a constant with units, the label of a variable with units of time, or a computed expression  Examples:  @ascii\_time( 1150897824.87[sec] )  @ascii\_time( time )  @ascii\_time( “time – 1[hr]” ) |
| char ascii\_time4( time ) | Returns time with 4 digit year (available in 6.3.x and later revisions of CyFlex)  returns MM/DD/YYYY HH:MM:SS format from a double value of ANSI time – time may be a constant with units, the label of a variable with units of time, or a computed expression  Examples:  @ascii\_time4( 1150897824.87[sec] )  @ascii\_time4( time )  @ascii\_time4( “time – 1[hr]” ) |
| char xml\_time( time ) | returns CCYY-MM-DDTHH:MM:SS format from a double value of ANSI time – time may be a constant with units, the label of a variable with units of time, or a computed expression  Examples:  @xml\_time( 1150897824.87[sec] )  @xml\_time( time )  @xlm\_time( “time – 1[hr]” ) |
| unsigned int strlen( char \*string ) | Find the length of a string  Examples:  var1 = “abcde”  @strlen( var1 )  The above returns 5  Or  @strlen( ‘abcde’ )  The above returns 5 |
| int strcmp( char \*s1, char \*s2 ) | Compare two strings – functions the same as the standard C-library call  Example:  Var1 = “abc”  Var2 = “def”  @strcmp( Var1, Var2 )  The above returns -1  Var1 = “ghi”  Var2 = “ghi”  @strcmp( Var1, Var2 )  The above returns 0 because they match  Var1 = “jkl”  Var2 = “abc”  @strcmp( Var1, Var2 )  The above returns 1 |
| strcpy( char \*s1, char \*s2 ) | string copy  Examples:  Var1 = “111”  Var2 = “789”  @strcpy( Var1, Var2 )  The above results: Var1 = “789”  Var1 = “abc”  @strcpy( Var1, ‘8888’ )  The above results: Var1 = “8888” |
| char \*strstr( char \*s1, char \*s2 ) | Find the start of one string in another string  Examples:  Var1 = “0”  Var2 = “456”  @strstr( Var1, Var2 )  The above results: = “”  Var1 = “789”  Var2 = “8”  @strstr( Var1, Var2 )  The above results: = “89” |
| strncmp( char \*s1, char \*s2, int n ) | string compare – number of characters  Examples:  Var1 = “ABCDEFGH”  Var2 = “ABCDEFHI”  @strncmp( Var1, Var2, 4[none])  The above returns: 0  @strncmp( Var1, Var2, 8[none])  The above returns: -1  @strncmp( Var1, ‘ab’, 2[none] )  The above returns: 0 |
| strncpy( char \*s1, char \*s2, int n ) | Copy n characters from one string to another  Examples:  Var1 = “55555555”  Var2 = “789”  @strncpy( Var1, Var2, 3[none] )  The above results: Var1 = “7895555”  Var1 = “TRUE”  @strncpy( Var1, ‘FALSE’, 2[none] )  The above results: Var1 = “FAUE” |
| char\* strupr( char \*s1 ) | convert a string to upper case  Example:  Var1 = “abcdefgh”  @strupr( Var1 )  The above results: Var1 = “ABCDEFGH” |
| LOGICAL strcmp\_lbl\_lbl( char \*label1, char \*label2 ) | compare the values of 2 string variables  Example:  @strcmp\_lbl\_lbl( my\_s1, my\_s2 ) returns TRUE if the contents of string variable s1 is the same as the contents of string variable s2 |
| LOGICAL strcmp\_lbl\_lit( char \*label, char \*s1 ) | compare the value of a string variable with a string literal  Example:  @strcmp\_lbl\_lit( my\_s, ‘ctl\_spd’ ) returns TRUE if my\_s contains “ctl\_spd” |
| double str\_lbl\_value( char \*label ) | get the value of a variable whose label is contained in a string variable  Example:  @str\_lbl\_value( my\_s ) where my\_s contains the label ‘ctl\_spd’, returns 1000.5 |
| char \*str\_var\_string( char \*label, int format ) | get the value of a variable whose label is contained in a string variable - return the value as string with "format" places to the right of decimal – NOTE: for this function, the label must be contained in single quotes as shown below.  Example:  @str\_var\_string( ‘my\_s’, 2[none] ) where my\_s contains the label ‘ctl\_spd’, returns “1000.49”  #type label -> value contained  string my\_s -> ‘ctl\_spd’ (label of target variable)  real ‘ctl\_spd’ -> 1000.49 (this can be any type) |
| double str\_lbl\_def\_value( char \*label ) | return default (SI units) value of a variable whose label is contained in a string variable  Example:  @str\_lbl\_def\_value( my\_s ) where my\_s contains the label of a variable |
| short str\_lbl\_in\_str\_lbl\_out( char \*lable1, char \*label2) | where label1 and label2 are string variables which contain the labels i\_lbl\_1 and i\_lbl\_2, move the value of i\_lbl\_1 to i\_lbl\_2 using appropriate units conversion – return TRUE if successful  Example:  @str\_lbl\_in\_str\_lbl\_out( my\_pres1, my\_pres2 ) |
| long strstrx( char \*s1, char \*s2 ) | search s1 for the substring s2 and return the position in s1 where s2 starts  Example:  @strstrx( ‘a valuable lesson’, ‘val’ ) returns a 2 |
| double cal\_table\_min( char \*file ) | return the minimum engineering units of a calibration table file  Example:  @cal\_table\_min( ‘air\_mtr0\_p’ ) returns the minimum value of file /cell/tables/air\_mtr0\_p.tbl |
| double cal\_table\_max( char \*file ) | return the maximum engineering units of a calibration table file  Example:  @cal\_table\_max( ‘air\_mtr0\_p’ ) returns the maximum value of file /cell/tables/air\_mtr0\_p.tbl |
| double variable\_age( char \*label ) | return the time (in seconds) since a variable was updated  Example:  @variable\_age( ctl\_spd ) returns 1.00 if ctl\_spd variable was update 1 sec ago |
| double variable\_time( char \*label ) | return the time (time\_t) when a variable was last updated  Example:  @variable\_time( ctl\_spd ) returns 1150897824.87 |
| char\* units\_tag( char \*label, char \*tag ) | return a tag with value and units for a variable  Example:  @units\_tag( ‘ctl\_spd’, ‘speed’ ) returns “speed=1000.5[rpm]” |
| char\* value\_units( char \*label ) | return a string which contains the value[units] of a variable  Example:  @value\_units( ‘ctl\_spd’ ) returns “1000.5[rpm]” |
| short ctrl\_loop\_mode( char \*loop ) | return the open/closed mode of a control loop – where 0 =closed loop, 1=open loop and loop is the label of the feedback variable  Example:  @ctrl\_loop\_mode( fuel\_t ) |
| short file\_info( char \*filename, char \*type) | This function duplicates the C access() function. It determines the access permissions of a file.  type can be either ‘W\_OK’ (for writing), ‘R\_OK’ (for reading), or ‘X\_OK’ (for execute). The return will be TRUE if the specified access mode is permitted.  Example:  @file\_info( ‘/specs/PAM\_datapoint’, ‘W\_OK’ ) returns TRUE if the file is writeable |
| int array\_to\_stat( char \*arr\_label, char\*stat\_label, int n ) | This function will perform the standard statistical analysis on the first ‘n’ values found in the specified 1-dimensional array variable. The array variable must be of the REAL type.  Example:  @array\_to\_stat( ‘myarr’, ‘mystat’, 10[none] ) -- analyzes the first 10 values in myarr |
| short set\_array( char \*label, char \*value ) | This function will initialize all of the values of an array to the specified value.  Example:  @set\_array( ‘my\_array’, ‘0’ ) this fills the entire array with zeroes |
| double var\_to\_double( char \*label ) | This function searches memory for the specified variable label and returns a double floating point value equivalent to the contents of the variable, regardless of the data type.  The ‘label’ argument should not be included in quotes.  Example:  @var\_to\_double( Speed )  -999999 is return if the label is invalid or a string variable cannot be converted to a floating point value |
| LOGICAL is\_name\_ready( char \*reg\_name, char \*wait\_time ) | This function returns a ‘1’ (TRUE) if ‘reg\_name’ is a valid registered name of an application. The function will repeatedly test for the name for ‘wait\_time’ (in seconds) and return a ‘0’ (FALSE) if the name is not present after ‘wait\_time’ seconds.  Example”  @is\_name\_ready( ‘asam3\_1’, ‘10’ )  The name ‘asam3\_1’ is typically used for the instance of “asam3cli” that manages communications with a CUTY device. If “asam3cli” is running, then this function will return a TRUE value. |
| Char \*parse\_line( char \*label, char \*optional\_delim, int N | Parse a string variable value and return the contents of the Nth token in the string. (N starts at 0). The label may be either a STRING or STRING\_ARRAY variable. Example:  If the contents of variable ‘’my\_list’’ contains a string  ‘’ list0 list1 list2 list3’’ @parse\_line( ‘my\_list’, ‘ ‘, 2 ) Will return the string’’list2’’ Note that the label and the optional\_delim must be single-quoted. This function is available in 6.3.33 and later versions |

# *Revision History*

This document was revised as shown below.

|  |  |
| --- | --- |
| **Document Version** | **Reason for Change** |
| 1.0 | New document |
| 2.0 | Added section 8. Revision History  Applied CyberMetrix standard formatting to the document |
| 3.0 | Add @parse\_line user function information |