CLA Math Library

USER'S GUIDE

CLA-MATH-LIB-UG-V400

Copyright

Texas Instruments
12203 Southwest Freeway
Houston, TX 77477
http://www.ti.com/c2000

Revision Information

This is version V400 of this document, last updated on November 09, 2011.

Table of Contents

1 Introduction

The Texas Instruments® TMS320C28x Control Law Accelerator math library is a collection of optimized floating-point math functions for controllers with the CLA. This source code library includes several C callable assembly math functions. This revision of the library is meant to work with the CLA C compiler(codegen version 6.1.0 and above). All source code is provided so it can be modified to suit the user's requirements.

Examples are provided with this package to show the user how to integrate the library into their projects and use any of the routines.

Chapter 2 provides a host of resources on the CLA in general, the C compiler as well as training material.

Chapter 3 describes the directory structure of the package.

Chapter 4 provides step-by-step instructions on how to integrate the library into a project and use any of the math routines.

Chapter 5 describes each function in the library.

Chapter 6 provides a revision history of the library.

Examples have been provided for each library routine. They can be found in the *examples* directory. For the current revision, all examples have been written for the *F2806x* device and tested on an *F28069 controlSTICK* platform. Each example has a script **"SetupDebugEnv.js"** that can be launched from the *Scripting Console* in CCS. These scripts will setup all the watch variable and graphs for each example.

2 Other Resources

There is a live Wiki page for answers to CLA frequently asked questions(FAQ). Links to other CLA references such as training videos will be posted here as well. CLA Wiki Page.

The following Wiki provides details on the C compiler for the CLA (available with codegen v6.1.0 and above): CLA C Compiler Wiki Page.

The same information may be found in the **F2806x Firmware Development Package Users Guide v130**.

Also check out the TI Piccolo page: http://www.ti.com/piccolo

And don't forgete the TI community website: ''http://e2e.ti.com''

Building the CLA library and examples requires **Codegen Tools V6.1.0 or later**

3 Library Structure

By default, the library and source code is installed into the following directory:

```
C:\TI\controlSUITE\libs\math\CLAmath\VERSION
```

VERSION indicates the current revision of the CLAmath library. Figure. 3.1 shows the directory structure while the subsequent table 3.1 provides a description for each folder.

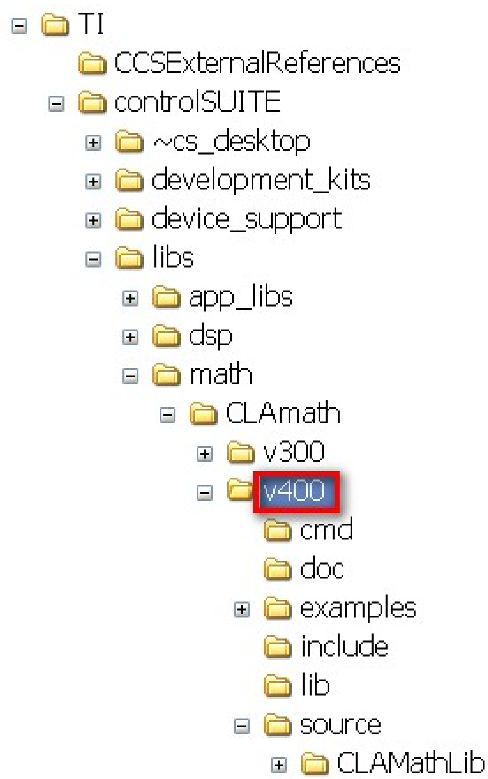

Figure 3.1: Directory Structure of the CLAMath Library

Folder	Description
<base>	Base install directory. By default this is C:/TI/controlSUITE/libs/math/CLAmath/VERSION For the rest of this document <base> will be omitted from the directory names
<base>/doc	Documentation for the current revision of the library including revision history
<base>/examples	Examples that illustrate the library functions. At the time of writing these examples were built for the F2806x platform using CCS4 platform but they can be imported into CCS5
<base>/include	Header files for the CLAMath library
<base>/lib	Pre-built CLAMath libraries
<base>/source	Source files and project for the library. Allows the user to reconfigure, modify and re-build the library to suit their particular needs

Table 3.1: CLAMath Library Directory Structure Description

4 Using the CLA Math Library

The source code and project for the CLA math library is provided. If you import the library project into CCSv4 you will be able to view and modify the source code for all the math routines and lookup tables (see Fig. 4.1)

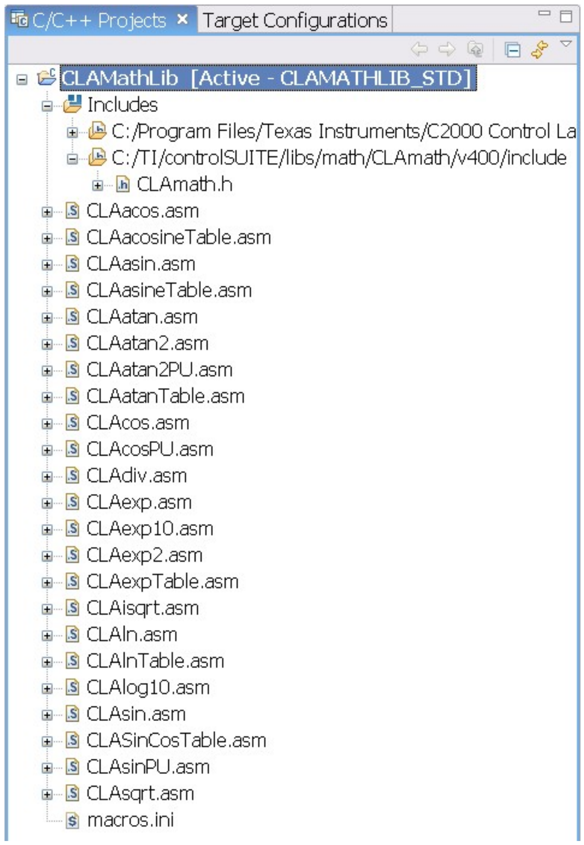

Figure 4.1: CLA Math Library Project View

The current version of the library has two build configurations (Fig. 4.2) **CLAMATHLIB_STD** and **CLAMATHLIB_FPU32_SUPPORT**. The difference between the two is the **CLAMATH-LIB_FPU32_SUPPORT** configuration is built with the **–fpu_support=fpu32** run-time support option turned on. This allows the CLA math library to be integrated into a project which has the **fpu32** option turned on. Each build config, when compiled, yields differently titled libraries: **CLAmath.lib** for the standard build configuration and **CLAmath_fpu32.lib** for the floating-point supported build.

NOTE: IF YOU TRY TO LINK IN THE STANDARD BUILD LIBRARY INTO ANOTHER PROJECT WHICH HAS FPU32 SUPPORT TURNED ON YOU WILL GET A COMPILER ERROR ABOUT MISMATCHING INSTRUCTION SET ARCHITECTURES, HENCE THE NEED FOR THE FPU32_SUPPORT BUILD CONFIGURATION

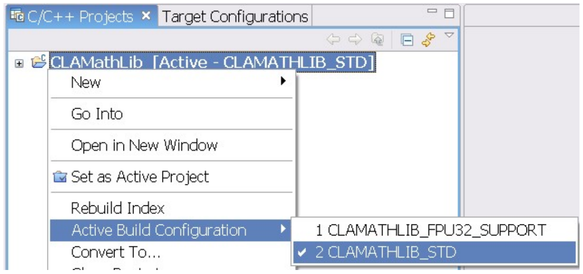

Figure 4.2: Library Build Configurations

To begin integrating the library into your project you need to follow these easy steps

1. Go to the **Project Properties->C/C++ Build->C2000 Compiler->Include Options** (see Fig. 4.3) and add the relative path, **INSTALLROOT_TO_CLAMATH_VERSION** (VERSION is the current version of the library), to the list of search directories. The macro INSTALLROOT_TO_CLAMATH_VERSION is specified in the **macros.ini** file in each example, however, you may have to redefine the path in your project depending on where the library is situated in the local machine.

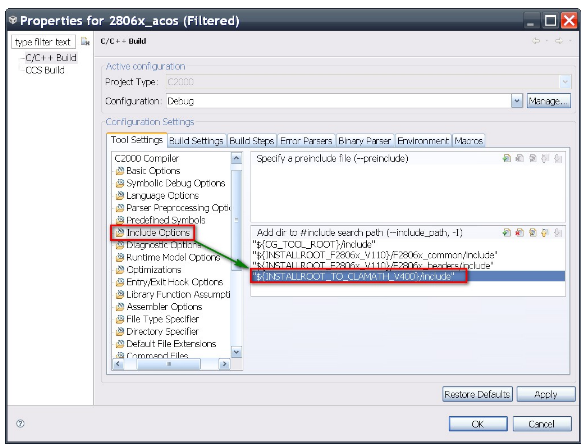

Figure 4.3: Adding the Include Search Path for the Library

2. Enable the **–cla_support** option in the **Runtime Model Options** to **cla0** as shown in Fig. 4.4

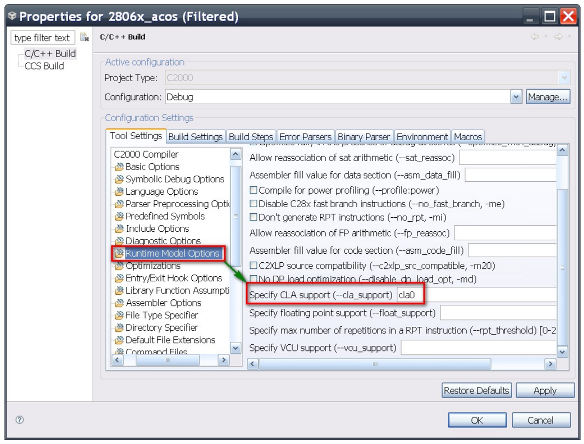

Figure 4.4: Turning on CLA support

3. Add the name of the library and its location to the **File Search PAth** as shown in Fig. 4.5.

   NOTE: IF YOUR PROJECT HAS FPU32 SUPPORT TURNED ON YOU WILL NEED TO ADD THE **CLAmath_fpu32.lib** LIBRARY IN THE UPPER BOX

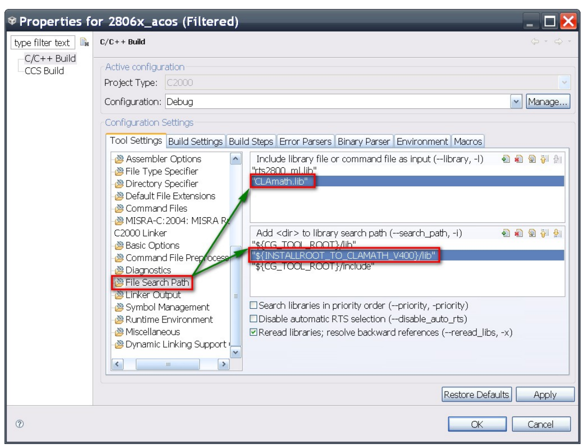

Figure 4.5: Adding the library and location to the file search path

5 Mathematical Functions

The following functions are included in this release of the CLAmath Library. The source code for these functions can be found in the *source/CLAMathLib* folder.

Trigonometric	
CLAcos	CLAsin
CLAsincos	CLAatan
CLAatan2	CLAatan2PU
CLAcosPU	CLAsinePU
CLAacos	CLAasin
Logarithmic	
CLAln	CLAlog10
Exponential	
CLAexp	CLAexp10
CLAexp2	
Miscellanous	
CLAdiv	CLAisqrt
CLAsqrt	

Table 5.1: List of Functions

5.1 Arc-Cosine

Prototype:
float CLAacos(float fVal)

Parameters:
fVal Input Value

Returns:
Angle in radians

Description:
This function calculates the arc-cosine of an argument value i.e. $acos(fVal)$ or $\cos^{-1}(fVal)$, in the following manner

1. Calculate absolute of the input X
2. Use the upper 6-bits of input "X" value as an index into the table to obtain the coefficients for a second order equation
3. Calculate the angle using the folowing equation:

$$\begin{aligned}\cos^{-1}(Ratio) &= A0 + A1 * fVal + A2 * fVal * fVal \\ &= A0 + fVal(A1 + A2 * fVal)\end{aligned}$$

4. The final angle is determined as follows:

$$\begin{aligned}&if(X < 0)\\ &\quad Angle = Pi - Angle\end{aligned}$$

Equation:
$\theta = cos^{-1}(fVal)$

5.2 Arc-Sine

Prototype:
float CLAasin(float fVal)
Parameters:
fVal Input Value
Returns:
Angle in radians
Description:
This function calculates the arc-sine of an argument i.e. $asin(fVal)$ or $\sin^{-1}(fVal)$ in the following manner

1. Calculate absolute of the input X
2. Use the upper 6-bits of input "X" value as an index into the table to obtain the coefficients for a second order equation
3. Calculate the angle using the folowing equation:

$$\begin{aligned}\sin^{-1}(Ratio) &= A0 + A1 * fVal + A2 * fVal * fVal \\ &= A0 + fVal(A1 + A2 * fVal)\end{aligned}$$

4. The final angle is determined as follows:

$$\begin{aligned}&if(X < 0) \\ &\quad Angle = -Angle\end{aligned}$$

Equation:
$\theta = \sin^{-1}(fVal)$

5.3 Arc-Tangent of a ratio

Prototype:
float CLAatan2(float fVal1, float fVal2)

Parameters:
fVal1 First Input Value

fVal2 Second Input Value

Returns:
Angle in radians

Description:
This function calculates the arc-tangent of the ratio of two input variables i.e. $atan(\frac{fVal1}{fVal2})$ or $\tan^{-1}(\frac{fVal1}{fVal2})$ in the following manner

1. $$\begin{aligned} if(|fVal1| >= |fVal2|) \\ Numerator &= |fVal2| \\ Denominator &= |fVal1| \\ else \\ Numerator &= |fVal1| \\ Denominator &= |fVal2| \end{aligned}$$

2. Ratio = $\frac{Numerator}{Denominator}$
NOTE: RATIO RANGE = 0.0 TO 1.0
3. Use the upper 6-bits of the "Ratio" value as an index into the table, **CLAatan2Table**, to obtain the coefficients for a second order equation
4. Calculate the angle using the folowing equation:

$$\begin{aligned} \tan^{-1}(Ratio) &= A0 + A1 * Ratio + A2 * Ratio * Ratio \\ &= A0 + Ratio(A1 + A2 * Ratio) \end{aligned}$$

5. The final angle is determined as follows:

$$\begin{aligned} if(fVal1 >= 0 \; and \; fVal2 &>= 0 \; and \; |fVal1| >= |fVal2|) \\ Angle &= arctan(\frac{|fVal2|}{|fVal1|}) \\ if(fVal1 >= 0 \; and \; fVal2 &>= 0 \; and \; |fVal1| < |fVal2|) \\ Angle &= PI/2 - arctan(\frac{|fVal2|}{|fVal1|}) \\ if(fVal1 < 0 \; and \; fVal2 &>= 0 \; and \; |fVal1| < |fVal2|) \\ Angle &= PI/2 + arctan(\frac{|fVal2|}{|fVal1|}) \\ if(fVal1 < 0 \; and \; fVal2 &>= 0 \; and \; |fVal1| >= |fVal2|) \\ Angle &= PI - arctan(\frac{|fVal2|}{|fVal1|}) \\ if(fVal2 < 0) \end{aligned}$$

$$Angle = -Angle$$

Equation:

$\theta = \tan^{-1}(\frac{fVal1}{fVal2})$

5.4 Arc-Tangent of a Ratio per Unit

Prototype:
float CLAatan2PU(float fVal1, float fVal2)

Parameters:
fVal1 First Input Value

fVal2 Second Input Value

Returns:
Angle per 2π radians

Description:
This function calculates the arc-tangent of a ratio per unit i.e. $\frac{atan(\frac{fVal1}{fVal2})}{2*\pi}$ or $\frac{\tan^{-1}(\frac{fVal1}{fVal2})}{2*\pi}$ in the following manner

1.
$$
\begin{aligned}
if(|fVal1| >= |fVal2|) & \\
Numerator &= |fVal2| \\
Denominator &= |fVal1| \\
else & \\
Numerator &= |fVal1| \\
Denominator &= |fVal2|
\end{aligned}
$$

2. $Ratio = \frac{Numerator}{Denominator}$
NOTE: RATIO RANGE = 0.0 TO 1.0
3. Use the upper 6-bits of the "Ratio" value as an index into the table, **CLAatan2Table**, to obtain the coefficients for a second order equation
4. Calculate the angle using the folowing equation:

$$
\begin{aligned}
\tan^{-1}(Ratio) &= A0 + A1 * Ratio + A2 * Ratio * Ratio \\
&= A0 + Ratio(A1 + A2 * Ratio)
\end{aligned}
$$

5. The final angle is determined as follows:

$$
\begin{aligned}
if(fVal1 >= 0 \; and \; fVal2 &>= 0 \; and \; |fVal1| >= |fVal2|) \\
Angle &= arctan(\frac{|fVal2|}{|fVal1|}) \\
if(fVal1 >= 0 \; and \; fVal2 &>= 0 \; and \; |fVal1| < |fVal2|) \\
Angle &= PI/2 - arctan(\frac{|fVal2|}{|fVal1|}) \\
if(fVal1 < 0 \; and \; fVal2 &>= 0 \; and \; |fVal1| < |fVal2|) \\
Angle &= PI/2 + arctan(\frac{|fVal2|}{|fVal1|}) \\
if(fVal1 < 0 \; and \; fVal2 &>= 0 \; and \; |fVal1| >= |fVal2|) \\
Angle &= PI - arctan(\frac{|fVal2|}{|fVal1|}) \\
if(fVal2 < 0) &
\end{aligned}
$$

$$\begin{aligned} Angle &= -Angle \\ AnglePU &= \frac{Angle}{2 \times \pi} \end{aligned}$$

Equation:

$$\theta_{PU} = \frac{\tan^{-1}(\frac{fVal1}{fVal2})}{2*pi}$$

5.5 Arc-Tangent

Prototype:
float CLAatan(float fVal)

Parameters:
fVal Input Value

Returns:
Angle in radians

Description:
This function calculates the arc-tangent of the argument i.e. $atan(fVal)$ or $\tan^{-1}(fVal)$ in the following manner

1.
$$\begin{aligned} if(1.0 \quad &>= \quad |fVal|) \\ Numerator &= \quad |fVal| \\ Denominator &= \quad 1.0 \\ else & \\ Numerator &= \quad 1.0 \\ Denominator &= \quad |fVal| \end{aligned}$$

2. $Ratio = \frac{Numerator}{Denominator}$
NOTE: RATIO RANGE = 0.0 TO 1.0
3. Use the upper 6-bits of the “Ratio” value as an index into the table, **CLAatan2Table** to obtain the coefficients for a second order equation
4. Calculate the angle using the folowing equation:
$$\begin{aligned} \tan^{-1}(Ratio) &= A0 + A1 * Ratio + A2 * Ratio * Ratio \\ &= A0 + Ratio(A1 + A2 * Ratio) \end{aligned}$$

5. The final angle is determined as follows:
$$\begin{aligned} if(fVal >= 0 \; and \; 1.0 \quad &>= \quad abs(fVal)) \\ Angle \quad &= \quad \tan^{-1}(\frac{abs(fVal)}{1.0}) \\ if(fVal >= 0 \; and \; 1.0 \quad &< \quad abs(fVal)) \\ Angle \quad &= \quad PI/2 - \tan^{-1}(\frac{1.0}{abs(fVal)}) \\ if(fVal < 0) & \\ Angle \quad &= \quad -Angle \end{aligned}$$

Equation:
$\theta = \tan^{-1}(fVal)$

5.6 Cosine

Prototype:
float CLAcos(float fAngleRad)
Parameters:
fAngleRad Input angle in radians
Returns:
cosine of the angle(float)
Description:
This function calculates the cosine of an anlge i.e. $cos(rad)$, where rad is the input angle in radians and rad = K + X.

Using Taylor series expansion around the value K we get,

$$\begin{aligned}
cos(rad) = cos(K) &- sin(K) \times X \\
&- cos(K) \times \frac{X^2}{2!} \\
&+ sin(K) \times \frac{X^3}{3!} \\
&+ cos(K) \times \frac{X^4}{4!} \\
&- sin(K) \times \frac{X^5}{5!} \\
cos(rad) = cos(K) &+ X \times (-1.0 \times sin(K) \\
&+ X \times (-0.5 \times cos(K) \\
&+ X \times (0.166666 \times sin(K) \\
&+ X \times (0.04166666 \times cos(K) \\
&+ X \times (-0.00833333 \times sin(K)))))) \\
cos(rad) = cos(K) &+ X \times (-sin(K) \\
&+ X \times (Coef0 \times cos(K) \\
&+ X \times (Coef1_pos \times sin(K) \\
&+ X \times (Coef2 \times cos(K) \\
&+ X \times (Coef3_neg \times sin(K))))))
\end{aligned}$$

Equation:
$Y = cos(fAngleRad)$

5.7 Cosine Per-Unit

Prototype:
float CLAcosPU(float fAngleRadPU)

Parameters:
fAngleRadPU Input angle in radians(per 2π units)

Returns:
Cosine of the angle

Description:
This function calculates the cosine of a per-unit angle i.e. $cos(radPU)$, where radPU is the angle in radians(per 2π units) and radPU= K + X

Therefore rad= $radPU * 2 * \pi$

Using Taylor series expansion around the value K we get,

$$
\begin{aligned}
cos(rad) = cos(K) &- sin(K) \times X \\
&- cos(K) \times \frac{X^2}{2!} \\
&+ sin(K) \times \frac{X^3}{3!} \\
&+ cos(K) \times \frac{X^4}{4!} \\
&- sin(K) \times \frac{X^5}{5!} \\
cos(rad) = cos(K) &+ X \times (-1.0 \times sin(K) \\
&+ X \times (-0.5 \times cos(K) \\
&+ X \times (0.166666 \times sin(K) \\
&+ X \times (0.04166666 \times cos(K) \\
&+ X \times (-0.00833333 \times sin(K)))))) \\
cos(rad) = cos(K) &+ X \times (-sin(K) \\
&+ X \times (Coef0 \times cos(K) \\
&+ X \times (Coef1\_pos \times sin(K) \\
&+ X \times (Coef2 \times cos(K) \\
&+ X \times (Coef3\_neg \times sin(K))))))
\end{aligned}
$$

Equation:
$Y = cos(fAngleRadPU)$

5.8 Divide

Prototype:
float CLAdiv(float fNum, float fDen)

Parameters:
fNum Numerator
fDen Denominator

Returns:
(float) $\frac{fNum}{fDen}$

Description:
This fucntion uses the Newton Raphson approximation to converge on the answer.

$$
\begin{aligned}
Y^{'} &\approx \frac{1}{Den} \\
Y^{'} &= Y^{'} \times Den \\
Y^{''} &= Y^{'} - Y^{'} \times (2.0 - Y^{'} \times Den) \\
Y^{'''} &= Y^{''} \times Den \\
Y^{'''} &= Y^{''} - Y^{''} \times (2.0 - Y^{''} \times Den) \\
Y &= Y^{'''} \times Num
\end{aligned}
$$

Equation:
$Y = \frac{fNum}{fDen}$

5.9 Exponential

Prototype:
float CLAexp(float fVal)

Parameters:
fVal Input argument

Returns:
Exponential raised to the input argument

Description:
This function calculates the exponential rasied to the input argument i.e. e^x, where x is the input value. It is calculated as follows:

1. Calculate absolute of x
2. Identify the integer and mantissa of the input
3. Obtain the $e^{integer(x)}$ from the table **CLAExpTable**
4. Calculate the value of $e^{(mantissa)}$ by using the polynomial approx:

$$e^{X_m} = 1 + X_m \times (1 + X_m \times 0.5(1 + (\frac{X_m}{3}) \times (1 + \frac{X_m}{4} \times (1 + \frac{X_m}{5} \times (1 + \frac{X_m}{6} \times (1 + \frac{X_m}{7}))))))$$

5. The value of e^x is the product of results from (3) and (4)

Equation:
$Y = e^{fVal}$

5.10 Exponential rasied to a Ratio

Prototype:
float CLAexp2(float fNum, float fDen)

Parameters: ***fNum*** First argument
fDen Second argument

Returns: ***Value*** of the exponential raised to the ratio of the two input arguments

Description: This function calculates the exponential raised to a ratio of two numbers i.e. $e^{\frac{A}{B}}$, where A and B are the two input arguments. These are the steps in the calculation:

1. Calculate absolute of $x = \frac{A}{B}$
2. Identify the integer and mantissa of the input
3. Obtain the $e^{integer(x)}$ from the table **CLAExpTable**
4. Calculate the value of $e^{(mantissa)}$ by using the following polynomial approx:

$$e^{X_m} = 1 + X_m \times (1 + X_m \times 0.5(1 + (\frac{X_m}{3}) \times (1 + \frac{X_m}{4} \times (1 + \frac{X_m}{5} \times (1 + \frac{X_m}{6} \times (1 + \frac{X_m}{7}))))))$$

5. The value of e^x is the product of results from (3) and (4)

Equation: $Y = e^{\frac{fNum}{fDen}}$

5.11 Exponential(Base 10)

Prototype:

float CLAexp10(float fVal)

Parameters:

fVal Input argument

Returns:

Base 10 exponential of the input argument

Description:

This function calculates the base 10 exponential function of the input argument i.e. 10^x, where x is the input value. It is calculated as follows:

1. x = $|\frac{x}{log_(10)(e)}|$
2. Identify the integer and mantissa of the input
3. Obtain the $e^{integer(x)}$ from the table **CLAExpTable**
4. Calculate the value of $e^{(mantissa)}$ by using the polynomial approx:

$$e^{X_m} = 1 + X_m \times (1 + X_m \times 0.5(1 + (\frac{X_m}{3}) \times (1 + \frac{X_m}{4} \times (1 + \frac{X_m}{5} \times (1 + \frac{X_m}{6} \times (1 + \frac{X_m}{7}))))))$$

5. The value of e^x is the product of results from (3) and (4).

   It can be proven that $10^x = e^{\frac{x}{log_{10}e}}$ and since we have divided x by $log_{10}(e)$ in step (1), the result we obtain will be the desired 10^x

Equation:

$Y = 10^{fVal}$

5.12 Inverse Square Root

Prototype:
float CLAisqrt(float fVal)

Parameters:
fVal Input number

Returns:
Inverse Square root of input argument

Description:
This function calculates the inverse square root of the input argument i.e. $\frac{1}{\sqrt{X}}$, where X is the input argument

This fucntion uses the Newton Raphson approximation to converge on the answer.

$$\begin{aligned}
Y^{'} &\approx \frac{1}{\sqrt{X}} \\
Y^{''} &= Y^{'} \times (1.5 - Y^{'} \times Y^{'} \times X \times 0.5) \\
Y^{'''} &= Y^{''} \times (1.5 - Y^{''} \times Y^{''} \times X \times 0.5) \\
Y &= Y^{'''}
\end{aligned}$$

Equation:
$Y = \frac{1}{\sqrt{fVal}}$

5.13 Natural Logarithm

Prototype:

float CLAln(float fVal)

Parameters:

fVal Input argument

Returns:

Natural log of the input argument

Description:

This function calculates the natural log of the input argument i.e. $log_e(x)$, where x is the input value.

1. Calculate absolute of x
2. Identify the exponent of the input, store it float.
3. Identify the mantissa, X_m and use it to look up the polynomial coefficients in the table **CLALnTable**
4. Subtract the bias from the exponent and multiply it by Ln(2)
5. Calculate the value of $log_e(1+mantissa)$ by using the polynomial approx: $log_e(1+X_m) = a_0+X_m\times(a_1+X_m\times a_2)$
6. $Result = log_e(1 + X_m) + (Exponent - 127)\times(log_e(2))$

Equation:

$Y = log_e(fVal)$

5.14 Logarithm(Base 10)

Prototype:
float CLAlog10(float fVal)
Parameters:
fVal Input argument
Returns:
Base 10 log of the input argument
Description:
This function calculates the Log(base 10) of the input argument i.e. $log_{10}(x)$, where x is the input value

1. Calculate absolute of x
2. Identify the exponent of the input, store it float.
3. Identify the mantissa, X_m and use it to look up the polynomial coefficients in the table **CLALnTable**
4. Subtract the bias from the exponent and multiply it by Ln(2)
5. Calculate the value of $log_e(1 + mantissa)$ by using the polynomial approx: $log_e(1 + X_m) = a_0 + X_m \times (a_1 + X_m \times a_2)$
6. $Result = \frac{log_e(1+X_m)+(Exponent-127)\times(log_e(2))}{log_e(10)}$

Equation:
$Y = log_{10}(fVal)$

5.15 Sine

Prototype:
float CLAsin(float fAngleRad)
Parameters:
fAngleRad Input angle in radians
Returns:
Sine of the input angle
Description:
This function calculates the sine of an input angle i.e. $sin(rad)$, where rad is the input angle in radians and rad = K + X

Using Taylor series expansion around the value K we get,

$$
\begin{aligned}
Sin(rad) = Sin(K) &+ Cos(K) \times X \\
&- Sin(K) \times \frac{X^2}{2!} \\
&- Cos(K) \times \frac{X^3}{3!} \\
&+ Sin(K) \times \frac{X^4}{4!} \\
&+ Cos(K) \times \frac{X^5}{5!} \\
Sin(rad) = Sin(K) &+ X \times (Cos(K) \\
&+ X \times (-0.5 \times Sin(K) \\
&+ X \times (-0.166666 \times Cos(K) \\
&+ X \times (0.04166666 \times Sin(K) \\
&+ X \times (0.00833333 \times Cos(K))))) \\
Sin(rad) = Sin(K) &+ X \times (Cos(K) \\
&+ X \times (Coef0 \times Sin(K) \\
&+ X \times (Coef1 \times Cos(K) \\
&+ X \times (Coef2 \times Sin(K) \\
&+ X \times (Coef3 \times Cos(K)))))
\end{aligned}
$$

Equation:
$Y = sin(fAngleRad)$

5.16 Sine Per-Unit

Prototype:

float CLAsinPU(float fAngleRadPU)

Parameters:

fAngleRadPU Input angle in radians(per 2π units)

Returns:

Sine of the angle

Description:

This function calculates the sine of a per-unit angle i.e. $sin(radPU)$, where where radPU is the input angle in radians (per unit 2π) and radPU = K + X

Therefore rad= radPU*2*π

Using Taylor series expansion around the value K we get,

$$
\begin{aligned}
Sin(rad) = Sin(K) \quad & + \quad Cos(K) \times X \\
& - \quad Sin(K) \times \frac{X^2}{2!} \\
& - \quad Cos(K) \times \frac{X^3}{3!} \\
& + \quad Sin(K) \times \frac{X^4}{4!} \\
& + \quad Cos(K) \times \frac{X^5}{5!} \\
Sin(rad) = Sin(K) \quad & + \quad X \times (Cos(K) \\
& + \quad X \times (-0.5 \times Sin(K) \\
& + \quad X \times (-0.166666 \times Cos(K) \\
& + \quad X \times (0.04166666 \times Sin(K) \\
& + \quad X \times (0.00833333 \times Cos(K))))) \\
Sin(rad) = Sin(K) \quad & + \quad X \times (Cos(K) \\
& + \quad X \times (Coef0 \times Sin(K) \\
& + \quad X \times (Coef1 \times Cos(K) \\
& + \quad X \times (Coef2 \times Sin(K) \\
& + \quad X \times (Coef3 \times Cos(K)))))
\end{aligned}
$$

Equation:

$Y = sin(fAngleRadPU)$

5.17 Square Root

Prototype:

float CLAsqrt(float fVal)

Parameters:

fVal Input number

Returns:

Square root of input argument

Description:

This function calculates the square root of the input argument i.e. $\sqrt{X}$, where X is the input value

This fucntion uses the Newton Raphson approximation to converge on the answer.

$$\begin{aligned} Y^{'} &\approx \frac{1}{\sqrt{X}} \\ Y^{''} &= Y^{'} \times (1.5 - Y^{'} \times Y^{'} \times X \times 0.5) \\ Y^{'''} &= Y^{''} \times (1.5 - Y^{''} \times Y^{''} \times X \times 0.5) \\ Y &= Y^{'''} \times X \end{aligned}$$

Equation:

$Y = \sqrt{fVal}$

6 Revision History

V4.00: Major Update

- Source library re-built with CLA **C** compiler (codegen v6.1.0)
- Math macros from the previous release were retained and modified into C-callable assembly functions

V3.00: Major Update

- Twelve optimized floating point macros performing trigonometric, exponential and logarithmic operations were added to the CLAmath library
- Added a new macro library, *CLAmathBasic*, that implements 13 simple operations like basic arithmetic, type conversion and conditional statements

V2.00: Moderate Update

Two more functions, *atan* and *atan2* added to the list of available macros

V1.00a: Minor Update

Source code has not been altered. Changes made to prepare the package for controlSUITE release and improved usability in CCSv4.

V1.00: Initial Release

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DLP® Products	www.dlp.com	Broadband	www.ti.com/broadband
DSP	dsp.ti.com	Digital Control	www.ti.com/digitalcontrol
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Military	www.ti.com/military
Logic	logic.ti.com	Optical Networking	www.ti.com/opticalnetwork
Power Mgmt	power.ti.com	Security	www.ti.com/security
Microcontrollers	microcontroller.ti.com	Telephony	www.ti.com/telephony
RFID	www.ti-rfid.com	Video & Imaging	www.ti.com/video
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265