



**General RT VIs  
Documentation**

**January 2009**

**Contents**

Introduction..... 3  
 CAT2CAD.vi ..... 4  
 CAD2CAT.vi ..... 5  
 CAT2Offset.vi ..... 6  
 Offset2CAT.vi ..... 7  
 Ticks2speed.vi ..... 8  
 Speed2ticks.vi ..... 9  
 Time2ticks.vi ..... 10  
 Ticks2time.vi ..... 11  
 One\_shot\_rt.vi..... 12  
 DrivvenPID.vi..... 13  
 ProfileLoad.vi ..... 14  
 ProfileRead.vi ..... 15  
 Interp1DTableRT.vi..... 16  
 Interp2DTableRT.vi..... 17  
 XYInterp.vi ..... 18

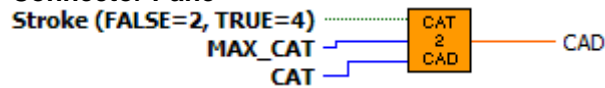
## **Introduction**

Each Drivven software toolkit installation includes a tool palette group called General RT VIs. This group includes several sub VIs which can assist with data conversion tasks, such as converting CAD to CAT or FPGA clock ticks to engine speed. Below are the documentation of each sub VI.


## CAT2CAD.vi

Converts Crank Angle Ticks (CAT) to Crank Angle Degrees (CAD). This VI can be used to convert CurrentPosition or a captured position (in CAT) of the EPT to CAD.


### Connector Pane




### Controls and Indicators

 **Stroke (FALSE=2, TRUE=4)** For 2-Stroke engines, set to FALSE.

For 4-Stroke engines, set to TRUE.

 **MAX\_CAT** Maximum Crank Angle Ticks per engine cycle. See definition of CAT within this VI.

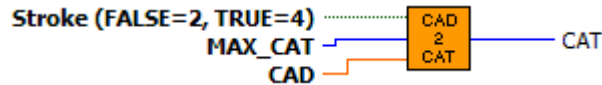
 **CAT** CAT: Crank Angle Ticks. Single unit of angular measure reported by the CurrentPosition output of the EPT VI. Reported as a power-of-two angular ticks of crank position travel, having a resolution dependent on EXTRAP and the number of physical teeth per crankshaft rotation. For example, if using the N-M EPT VI, which has an extrapolation of 7, the number of CAT per crank tooth would be  $2^7=128$ , and CurrentPosition would be evenly incremented by 128 CAT from one physical tooth to the next. If a 60-2 pattern were used, the maximum number of CAT per crankshaft rotation (cycle) would be  $60*128=7680$ . If the engine was a 4-stroke, the total number of CAT per engine cycle would be  $2*60*128=15360$ .

 **CAD** Crank Angle Degrees, calculated from the CAT input.

## CAD2CAT.vi

Converts Crank Angle Degrees (CAD) to Crank Angle Ticks (CAT).

### Connector Pane



### Controls and Indicators

**TF** **Stroke (FALSE=2, TRUE=4)** For 2-Stroke engines, set to FALSE.

For 4-Stroke engines, set to TRUE.

**U16** **MAX\_CAT** Maximum Crank Angle Ticks per engine cycle. See definition of CAT within this VI.

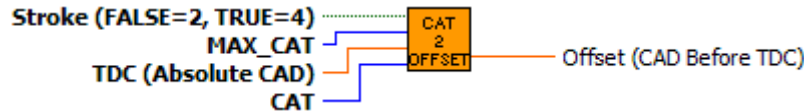
**SGL** **CAD** Crank Angle Degrees to be converted to CAT.

**U16** **CAT** CAT: Crank Angle Ticks. Single unit of angular measure reported by the CurrentPosition output of the EPT VI. Reported as a power-of-two angular ticks of crank position travel, having a resolution dependent on EXTRAP and the number of physical teeth per crankshaft rotation. For example, if using the N-M EPT VI, which has an extrapolation of 7, the number of CAT per crank tooth would be  $2^7=128$ , and CurrentPosition would be evenly incremented by 128 CAT from one physical tooth to the next. If a 60-2 pattern were used, the maximum number of CAT per crankshaft rotation (cycle) would be  $60*128=7680$ . If the engine was a 4-stroke, the total number of CAT per engine cycle would be  $2*60*128=15360$ .

## CAT2Offset.vi

Calculates an offset angle in Crank Angle Degrees (CAD) with respect to an absolute TDC from an absolute angle in Crank Angle Ticks (CAT).

### Connector Pane



### Controls and Indicators

**TF** **Stroke (FALSE=2, TRUE=4)** For 2-Stroke engines, set to FALSE.

For 4-Stroke engines, set to TRUE.

**U16** **MAX\_CAT** Maximum Crank Angle Ticks per engine cycle.

A Crank Angle Tick (CAT) is a single unit of angular measure reported by the CurrentPosition output of the EPT VI. Reported as a power-of-two angular ticks of crank position travel, having a resolution dependent on EXTRAP and the number of physical teeth per crankshaft rotation. For example, if using the N-M EPT VI, which has an extrapolation of 7, the number of CAT per crank tooth would be  $2^7=128$ , and CurrentPosition would be evenly incremented by 128 CAT from one physical tooth to the next. If a 60-2 pattern were used, the maximum number of CAT per crankshaft rotation (cycle) would be  $60 \times 128 = 7680$ . If the engine was a 4-stroke, the total number of CAT per engine cycle would be  $2 \times 60 \times 128 = 15360$ .

**SGL** **TDC (Absolute CAD)** Absolute position, in CAD, of a cylinder TDC with respect to absolute position 0 of the EPT. The EPT VIs locate absolute 0 position depending on the pattern type of the trigger wheel. For example, the EPT Encoder VI locates absolute position 0 at the rising edge of the first crank pulse following the rising edge of the cam pulse. This value should always be positive in terms of absolute CAD after absolute position 0.

**U16** **CAT** The absolute crank angle ticks with respect to absolute 0 position of the EPT.

**SGL** **Offset (CAD Before TDC)** This is the offset in terms of CAD before the TDC value specified to this VI.

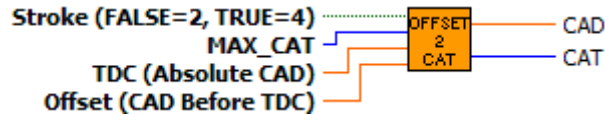
Positive values are before TDC (also referred to as Advance).

Negative values are after TDC (also referred to as Retard).

## Offset2CAT.vi

Converts an offset angle with respect to an absolute TDC to an absolute angle in Crank Angle Degrees (CAD) and Crank Angle Ticks (CAT). The Result in CAT can be wired directly to an FPGA-based fuel or spark control sub VI parameter for scheduling commands.

### Connector Pane



### Controls and Indicators

**TF** **Stroke (FALSE=2, TRUE=4)** For 2-Stroke engines, set to FALSE.

For 4-Stroke engines, set to TRUE.

**U16** **MAX\_CAT** Maximum Crank Angle Ticks per engine cycle.

A Crank Angle Tick (CAT) is a single unit of angular measure reported by the CurrentPosition output of the EPT VI. Reported as a power-of-two angular ticks of crank position travel, having a resolution dependent on EXTRAP and the number of physical teeth per crankshaft rotation. For example, if using the N-M EPT VI, which has an extrapolation of 7, the number of CAT per crank tooth would be  $2^7=128$ , and CurrentPosition would be evenly incremented by 128 CAT from one physical tooth to the next. If a 60-2 pattern were used, the maximum number of CAT per crankshaft rotation (cycle) would be  $60*128=7680$ . If the engine was a 4-stroke, the total number of CAT per engine cycle would be  $2*60*128=15360$ .

**SGL** **TDC (Absolute CAD)** Absolute position, in CAD, of a cylinder TDC with respect to absolute position 0 of the EPT. The EPT VIs locate absolute 0 position depending on the pattern type of the trigger wheel. For example, the EPT Encoder VI locates absolute position 0 at the rising edge of the first crank pulse following the rising edge of the cam pulse. This value should always be positive in terms of absolute CAD after absolute position 0.

**SGL** **Offset (CAD Before TDC)** This is the offset in terms of CAD before the TDC value specified to this VI.

Positive values are before TDC (also referred to as Advance).

Negative values are after TDC (also referred to as Retard).

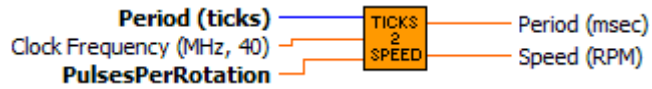
**SGL** **CAD** The calculated absolute CAD with respect to the absolute position 0 of the EPT VI.

**U16** **CAT** The calculated absolute CAT with respect to the absolute position 0 of the EPT VI. This value is typically wired to a Driven FPGA sub VI for scheduling a fuel or spark output.

## Ticks2speed.vi

This VI converts the period between trigger wheel teeth, in clock ticks, to rotational speed in RPM.

### Connector Pane



### Controls and Indicators

- U32 **Period (ticks)** The period, in clock ticks, between two pulses (or teeth) of a rotating trigger wheel.
  
- SGL **Clock Frequency (MHz, 40)** The clock Frequency at which Period is measured. Default = 40 MHz.
  
- SGL **PulsesPerRotation** Number of pulses (or teeth) during a single rotation of a trigger wheel.
  
- SGL **Period (msec)** The conversion of Period in clock ticks to milliseconds.
  
- SGL **Speed (RPM)** The calculated rotating speed of the trigger wheel in RPM.








## Speed2ticks.vi

This VI converts rotational shaft speed in RPM to a pulse Period. Period is calculated based on the specified number of trigger PulsesPerRotation of the shaft. Period is provided in two forms, milliseconds and clock ticks. The Period (ticks) output is calculated based on a specified Clock Frequency domain where the Period value will be used.

### Connector Pane



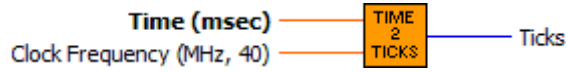
### Controls and Indicators

-  **Speed (RPM)** The rotational speed, in RPM, of the shaft.
-  **Clock Frequency (MHz, 40)** The Clock Frequency at which Period (ticks) will be used. Default = 40 MHz.
-  **PulsesPerRotation** Number of pulses (or teeth) during a single rotation of a trigger wheel.
-  **Period (msec)** The period, in milliseconds, between two pulses (or teeth) of a rotating trigger wheel.
-  **Period (ticks)** The period, in clock ticks, between two pulses (or teeth) of a rotating trigger wheel.




## Time2ticks.vi

Converts time in milliseconds to clock ticks, depending on the Clock Frequency where the result will be used.

### Connector Pane



### Controls and Indicators

-  **Time (msec)** Time, in milliseconds, to be converted to clock ticks.
-  **Clock Frequency (MHz, 40)** The Clock Frequency at which Ticks will be used. Default = 40 MHz.
-  **Ticks** Ticks of the clock equivalent to Time in milliseconds.




## Ticks2time.vi

Converts time, in clock ticks, to milliseconds, depending on the Clock Frequency where the result will be used.

### Connector Pane



### Controls and Indicators

-  **Ticks** Clock Ticks at a specified Clock Frequency to be converted to Time in milliseconds.
-  **Clock Frequency (MHz, 40)** The Clock Frequency at which Ticks is derived. Default = 40 MHz.
-  **Time (msec)** Time in milliseconds equivalent to the specified clock Ticks at the specified Clock Frequency.

## One\_shot\_rt.vi

Generates a one-loop, one-shot output pulse upon the rising edge of the input Boolean.

### Connector Pane



### Controls and Indicators



I Boolean input signal.



O One-loop one-shot output pulse.

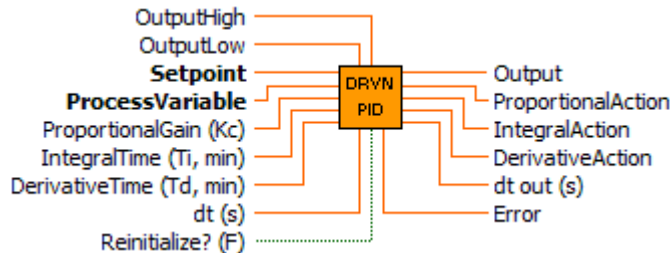
## DrivenPID.vi

Simple Ideal PID control algorithm according to:

$$\text{Output} = [ Kc * e(t) ] + [ (Kc / Ti) * \int e(t)dt ] + [ (Kc * Td) * dPV/dt ]$$

The PID algorithm features output range limiting with integrator anti-windup and bumpless controller output for PID gain changes. Each action is provided as an output.

### Connector Pane



### Controls and Indicators

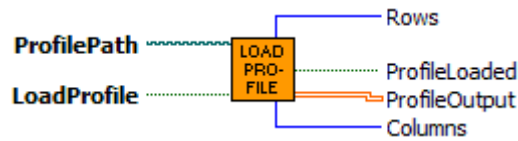
- SGL** **OutputHigh** Output high is the maximum value for controller output.
- SGL** **OutputLow** Output low is the minimum value for controller output.
- SGL** **Setpoint** The setpoint value of the process variable that is being controlled. This is the desired value for the process variable.
- SGL** **ProcessVariable** The measured value of the process variable that is being controlled. This is the feedback value of the feedback control loop.
- SGL** **ProportionalGain (Kc)** Specifies the proportional gain of the controller. The default is 1. In the equation that defines the PID controller, KC represents the proportional gain.
- SGL** **IntegralTime (Ti, min)** Specifies the integral time in minutes. The default is 0.01.
- SGL** **DerivativeTime (Td, min)** Specifies the derivative time in minutes. The default is 0.
- SGL** **dt out (s)** Actual time interval in seconds; either the value specified for dt (s) or the computed interval if dt (s) was specified as -1.0.
- TF** **Reinitialize? (F)** Set to TRUE to reinitialize internal parameters (such as integrated error) to default values of 0.0.
- SGL** **Output** The control output of the PID algorithm that is applied to the controlled process.
- SGL** **ProportionalAction** Proportional Action =  $Kc * e(t)$
- SGL** **IntegralAction** Integral Action =  $(Kc / Ti) * \int e(t)dt$
- SGL** **DerivativeAction** Derivative Action =  $(Kc * Td) * dPV/dt$
- SGL** **Error** Error = Setpoint - ProcessVariable

## ProfileLoad.vi







Accepts a path to a spreadsheet file and outputs a 2D array of values from the spreadsheet along with row and column sizes.

The output is useful for feeding the Driven ReadProfile VI. The ReadProfile VI will read a single row each time it is called. Each row of output can be used as set point profiles to various control algorithms.

### Connector Pane



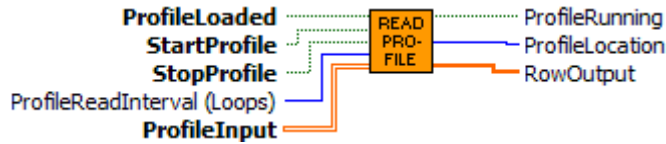
### Controls and Indicators

-  **ProfilePath** Complete path to the spreadsheet file.
-  **LoadProfile** The spreadsheet file is loaded upon the rising edge of LoadProfile.
-  **ProfileLoaded** Indicates that a profile has been loaded.
-  **ProfileOutput** Complete 2D array from the loaded spreadsheet file. Should be wired to the ProfileInput of the Driven ReadProfile VI.
-  **Rows** Number of rows in the 2D spreadsheet file.
-  **Columns** Number of columns in the 2D spreadsheet file.









## ProfileRead.vi

Accepts a 2D ProfileInput and reads a single row each time it is called. Each row of output can be used as set point profiles to various control algorithms.

### Connector Pane



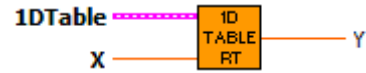
### Controls and Indicators

-  **ProfileLoaded** Accepts the ProfileLoaded output from the Driven LoadProfile VI.
-  **StartProfile** Upon the rising edge of StartProfile, if ProfileLoaded is TRUE, then the VI begins to read a row from the 2D profile each time this VI is called.
-  **StopProfile** Upon the rising edge of StopProfile, this VI stops reading rows from the 2D profile.
-  **ProfileReadInterval (Loops)** Specifies the number of VI calls between each ProfileRead. If a row is to be read each call, then 0 should be wired to this input.
-  **ProfileInput** Complete 2D array which is read on each VI call. Should be wired from the ProfileOutput of the Driven LoadProfile VI.
-  **ProfileRunning** Indicates that a profile is being read on each VI call.
-  **ProfileLocation** Indicates the most recent row that was read by this VI.
-  **RowOutput** The current row output from the 2D profile array.






## Interp1DTableRT.vi

Calculates Y from XArray and YArray by 1D interpolation according to the X input.

### Connector Pane



### Controls and Indicators

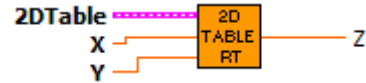
-  **X** Index into the XArray.
-  **1DTable** 1D lookup table. The number of elements within each array should be the same. The values of XArray should be monotonically increasing.
-  **XArray** Array of values which correspond to each value of the YArray. The values of XArray should be monotonically increasing.
-  **YArray** Array of output values which correspond to each value of the XArray.
-  **Y** Interpolated output.




## Interp2DTableRT.vi


Calculates Z from XArray, YArray and ZArray by 2D interpolation according to the X and Y inputs.

### Connector Pane





### Controls and Indicators


 **X** Index into the XArray.

 **Y** Index into the YArray.

 **2DTable**

 **XArray** Array of values which correspond to each row of the ZArray. The values of XArray should be monotonically increasing.

 **YArray** Array of values which correspond to each column of the ZArray. The values of YArray should be monotonically increasing.

 **ZArray** 2D Array of output values which correspond to each value of the XArray and YArray.

 **Z** Interpolated output.

## XYInterp.vi

Calculates Y from an interpolation operation bracketed by X0, X1, Y0 and Y1 according to:

$$Y = Y0 + (X - X0) * (Y1 - Y0) / (X1 - X0)$$

If  $(X1 - X0) = 0$ , then Y is set to Y0.

