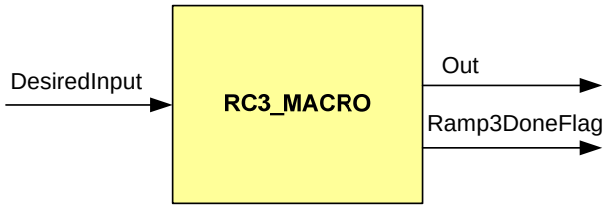
RMP3CNTL	*Ramp 3 Control Module*

Description

This module implements a ramp down macro. The output flag variable *Ramp3DoneFlag* is set to 0x7FFFFFFF when the output variable *Out* equals the input variable *DesiredInput.*

.



Availability

This IQ module is available in one interface format:

1) The C interface version

Module Properties

Type: Target Independent, Application Independent

Target Devices: 28x Fixed Point or Piccolo

C Version File Names: rmp3cntl.h

IQmath library files for C: IQmathLib.h, IQmath.lib

C Interface

Object Definition

The structure of RMP3 object is defined by following structure definition

```
typedef struct { Uint32 DesiredInput;          // Input: Desired ramp input (Q0)
                 Uint32 Ramp3Delay;            // Parameter: Delay in # of sampling period (Q0)
                 Uint32 Ramp3DelayCount;       // Variable: Counter for rmp3 delay (Q0)
                 int32 Out;                    // Output: Ramp3 output (Q0)
                 int32 Ramp3Min;               // Parameter: Minimum ramp output (Q0)
                 Uint32 Ramp3DoneFlag;         // Output: Flag output (Q0) ction
               } RMP3;

typedef RMP3 *RMP3_handle;
```

Item	Name	Description	Format*	Range(Hex)
Input	DesiredInput	Desired ramp input	Q0	80000000-7FFFFFFF
Outputs	Out	Ramp 3 output	Q0	80000000-7FFFFFFF
	Ramp3DoneFlag	Flag output	Q0	0 or 7FFFFFFF
RMP3 parameter	Ramp3Min	Minimum limit	Q0	80000000-7FFFFFFF
	Ramp3Delay	Delay in no. of sampling period	Q0	00000000-7FFFFFFF
Internal	Ramp3DelayCount	Counter for rmp3 delay	Q0	00000000-7FFFFFFF

*GLOBAL_Q valued between 1 and 30 is defined in the IQmathLib.h header file.

Special Constants and Data types

RMP3
The module definition is created as a data type. This makes it convenient to instance an interface to the ramp3 control. To create multiple instances of the module simply declare variables of type RMP3.

RMP3_handle
User defined Data type of pointer to RMP3 module

RMP3_DEFAULTS
Structure symbolic constant to initialize RMP3 module. This provides the initial values to the terminal variables as well as method pointers.

Methods

RC3_MACRO(RMP3_handle);

This definition implements one method viz., the rmp3 control computation macro. The input argument to this macro is the module handle.

Module Usage

Instantiation
The following example instances two RMP3 objects
RMP3 rmp1, rmp2;

Initialization
To Instance pre-initialized objects
RMP3 rmp1 = RMP3_DEFAULTS;
RMP3 rmp2 = RMP3_DEFAULTS;

Invoking the computation macro
RC3_MACRO(rmp1);
RC3_MACRO(rmp2);

Example

The following pseudo code provides the information about the module usage.

```
main()
{

}

void interrupt periodic_interrupt_isr()
{
        rmp1.DesiredInput = input1;          // Pass inputs to rmp1
        rmp2.DesiredInput = input2;          // Pass inputs to rmp2

        RC3_MACRO (rmp1);                    // Call compute macro for rmp1
        RC3_MACRO (rmp2);                    // Call compute macro for rmp2

        out1 = rmp1.Out;                     // Access the outputs of rmp1
        out2 = rmp2.Out;                     // Access the outputs of rmp2

}
```

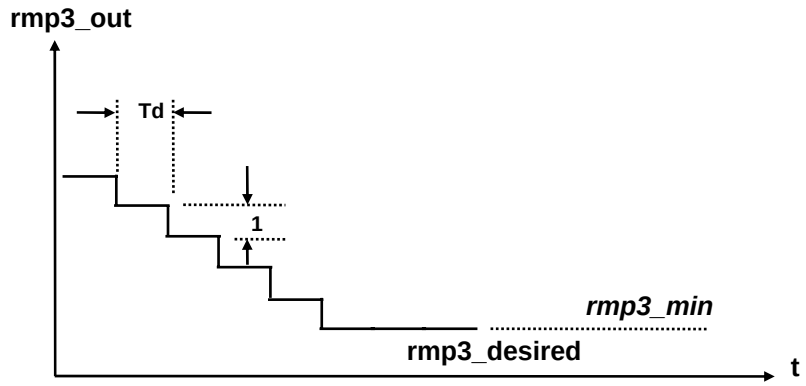
Technical Background

Implements the following equations:

Out = *Out* - 1, for t = n **.** Td, n = 1, 2, 3, ….. and (*Out* - 1) > *Ramp3Min*
= *Ramp3Min*, for (*Out* - 1) < *Ramp3Min*

Ramp3DoneFlag = 7FFFh, when *Out* = *DesiredInput* or *Ramp3Min*

where, Td = *Ramp3Delay* **.** Ts
Ts = Sampling time period



Example:
Out=500(initial value), DesiredInput=20(user specified),
Ramp3Delay=100(user specified), sampling loop time period Ts=0.000025 Sec.
This means that the time delay for each ramp step is Td=100x0.000025=0.0025 Sec. Therefore, the total ramp down time will be Tramp=(500-20)x0.0025 Sec=1.2 Sec