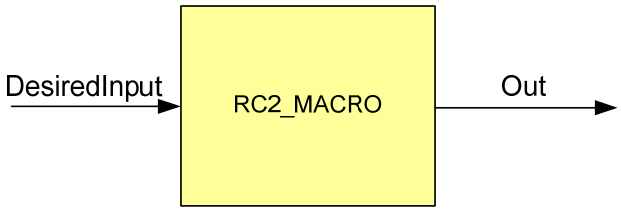
RMP2CNTL	*Ramp 2 Control Module*

Description

This module implements a ramp up and ramp down macro. The output variable *Out* follows the desired ramp value *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: rmp2cntl.h

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

C Interface

Object Definition

The structure of RMP2 object is defined by following structure definition

```
typedef struct { _iq15 DesiredInput;          // Input: Desired ramp input (Q15)
                 _iq15 Ramp2Max;              // Parameter: Maximum limit (Q15)
                 _iq15 Ramp2Min;              // Parameter: Minimum limit (Q15)
                 Uint32  Ramp2DelayCount;     // Variable: Incremental delay (Q0)
                 Uint32 Ramp2Delay;           // Parameter: Delay in # of sampling period (Q0)
                 _iq15 Out;                   // Output: Ramp2 output (Q15)
               } RMP2;

typedef RMP2 *RMP2_handle;
```

Item	Name	Description	Format*	Range(Hex)
Input	DesiredInput	Desired ramp input	Q15	80000000-7FFFFFFF
Output	Out	Ramp 2 output	Q15	80000000-7FFFFFFF
RMP2 parameter	Ramp2Max	Maximum limit	Q15	80000000-7FFFFFFF
	Ramp2Min	Minimum limit	Q15	80000000-7FFFFFFF
	Ramp2Delay	Delay in no. of sampling period	Q0	00000000-7FFFFFFF
Internal	Ramp2DelayCount	Incremental delay	Q0	00000000-7FFFFFFF

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

Special Constants and Data types

RMP2
The module definition is created as a data type. This makes it convenient to instance an interface to the ramp2 control. To create multiple instances of the module simply declare variables of type RMP2.

RMP2_handle
User defined Data type of pointer to RMP2 module

RMP2_DEFAULTS
Structure symbolic constant to initialize RMP2 module. This provides the initial values to the terminal variables as well as method pointers.

Methods

RC2_MACRO (RMP2_handle);

This definition implements one method viz., the rmp2 control computation macro. The input argument to this macro is the module handle.

Module Usage

Instantiation
The following example instances two RMP2 objects
RMP2 rmp1, rmp2;

Initialization
To Instance pre-initialized objects
RMP2 rmp1 = RMP2_DEFAULTS;
RMP2 rmp2 = RMP2_DEFAULTS;

Invoking the computation macro
RC2_MACRO(rmp1);
RC2_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

        RC2_MACRO (rmp1);                    // Call compute macro for rmp1
        RC2_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:

Case 1: When *DesiredInput* > *Out*.

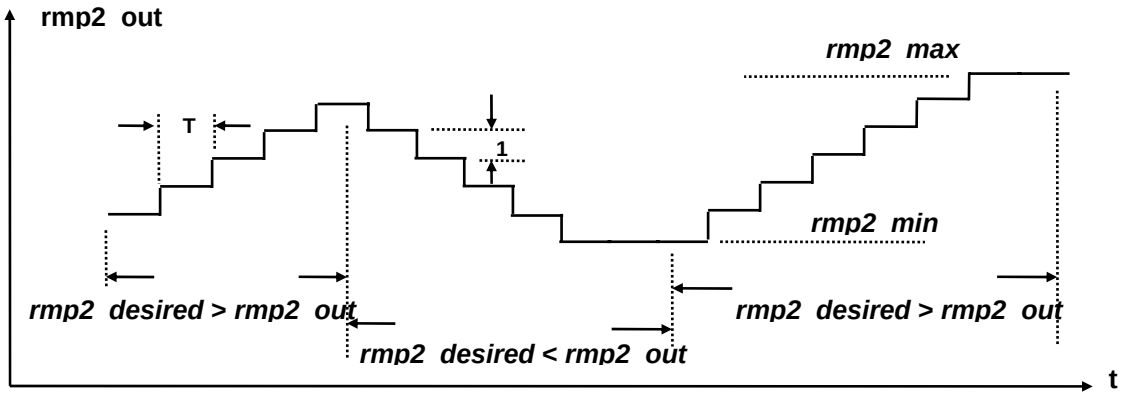
Out= *Out*+ 1, for t = n **.** Td, n = 1, 2, 3, ….. and (*Out*+ 1) < *Ramp2Max*
= *Ramp2Max*, for (*Out*+ 1) > *Ramp2Max*

where, Td = *Ramp2Delay* **.** Ts
Ts = Sampling time period

Case 2: When *DesiredInput* < *Out*.

Out= *Out*- 1, for t = n **.** Td, n = 1, 2, 3, ….. and (*Out*- 1) > *Ramp2Min*
= *Ramp2Min*, for (*Out*- 1) < *Ramp2Min*

where, Td = *Ramp2Delay* **.** Ts
Ts = Sampling time period



Example:
Out=0(initial value), DesiredInput=1000(user specified),
Ramp2Delay=500(user specified), sampling loop time period Ts=0.000025 Sec.
This means that the time delay for each ramp step is Td=500x0.000025=0.0125 Sec.
Therefore, the total ramp time will be Tramp=1000x0.0125 Sec=12.5 Sec