LINZ
CENTER OF
MECHATRONICS
GMBH

Project Documentation DemoApplication

August 1, 2017

Contents

I X2C Model 2

1 Version Information 2
1.1 X2C . . . 2
1.2 Operating System . . . 2
1.3 Scilab . . . 2

2 Model Structure 2
2.1 Xcos Model . . . 2
2.2 Subsystems . . . 3

3 Model Parameter 4
3.1 Sample Time . . . 4

4 Mask Parameter 5

II Frame Program Documentation 7

5 File Index 7
5.1 File List . . . 7

6 File Documentation 7
6.1 inc/Hardware.h File Reference . . . 7
6.1.1 Detailed Description . . . 8
6.1.2 Function Documentation . . . 8
6.2 inc/Main.h File Reference . . . 8
6.2.1 Detailed Description . . . 8
6.2.2 Function Documentation . . . 8
6.3 inc/X2cDataTypes.h File Reference . . . 9
6.3.1 Detailed Description . . . 9

III Used X2C-Blocks 10

7 Project Specific Blocks 10

8 Internal Library Blocks 10
AutoSwitch . . . 10
Constant . . . 11
Delay . . . 12
I . . . 13
Negation . . . 14
Sin3Gen . . . 15
SinGen . . . 17

Part I

X2C Model

1 Version Information

1.1 X2C

- X2Cfull: Version 1194

1.2 Operating System

- OS: Windows 7 6.1

1.3 Scilab

- Scilab: Version 5.5.2.1427793548
- Java: Version 1.6.0_41

2 Model Structure

2.1 Xcos Model

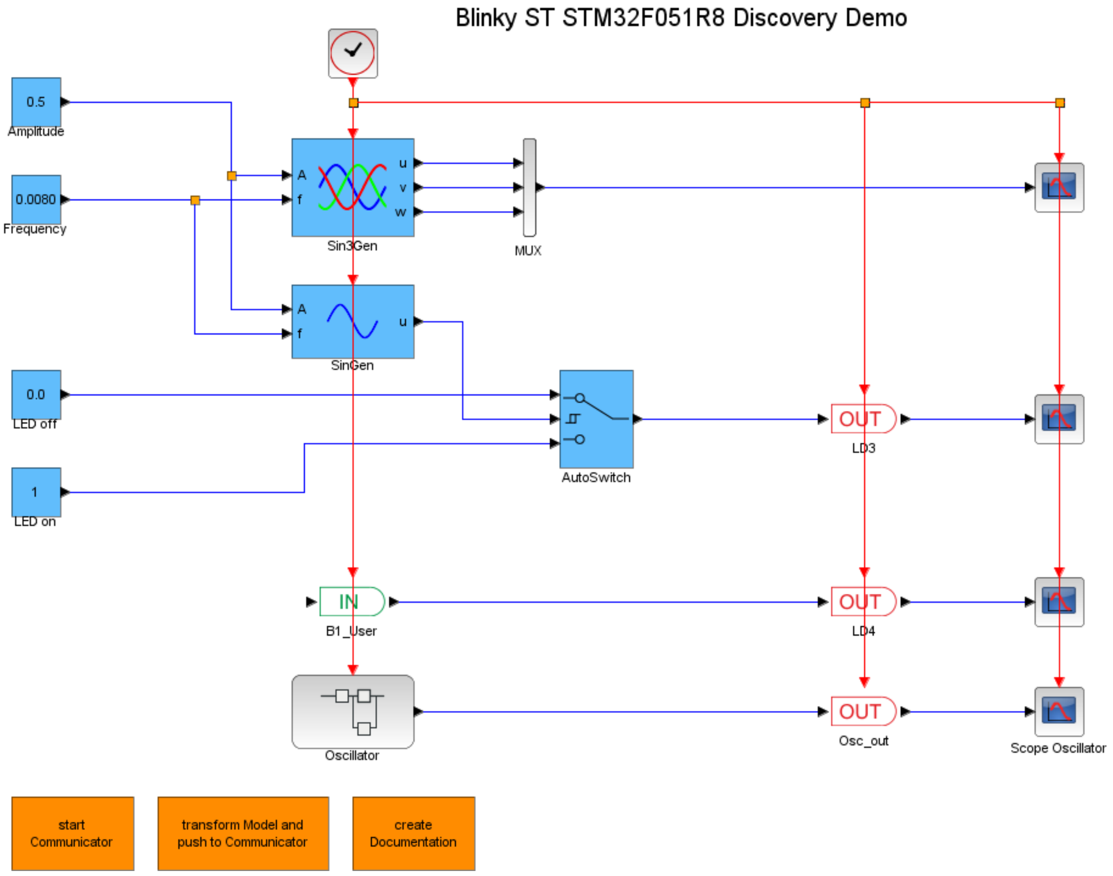


Figure 1: DemoApplication

2.2 Subsystems

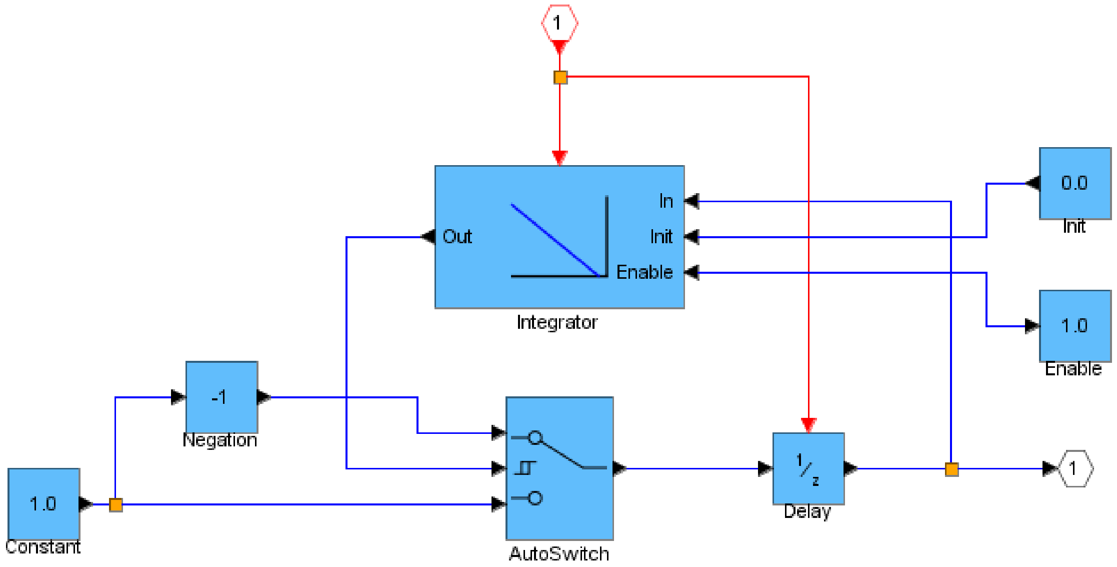


Figure 2: DemoApplication_Oscillator

3 Model Parameter

3.1 Sample Time

Sample Time	
T_S	$100 \mu s$

4 Mask Parameter

Constant: Amplitude	
Value	0.5
Used Implementation	FiP16

AutoSwitch: AutoSwitch	
Thresh_up	0.0
Thresh_down	0.0
Used Implementation	FiP16

Constant: Frequency	
Value	0.0080
Used Implementation	FiP16

Constant: LED off	
Value	0.0
Used Implementation	FiP16

Constant: LED on	
Value	1.0
Used Implementation	FiP16

AutoSwitch: Oscillator__AutoSwitch	
Thresh_up	0.5
Thresh_down	-0.5
Used Implementation	FiP16

Constant: Oscillator__Constant	
Value	1.0
Used Implementation	FiP16

Delay: Oscillator__Delay	
ts_fact	1.0
Used Implementation	FiP16

Constant: Oscillator__Enable	
Value	1.0
Used Implementation	Bool

Constant: Oscillator__Init	
Value	0.0
Used Implementation	FiP16

I: Oscillator__Integrator	
Ki	50.0
ts_fact	1.0
Used Implementation	FiP16

Negation: Oscillator__Negation	
Used Implementation	FiP16

Sin3Gen: Sin3Gen	
fmax	1000.0
Offset	0.0
ts_fact	1.0
Used Implementation	FiP16

SinGen: SinGen	
fmax	1000.0
Offset	0.0
Phase	0.0
ts_fact	1.0
Used Implementation	FiP16

Part II

Frame Program Documentation

5 File Index

5.1 File List

Here is a list of all documented files with brief descriptions:

inc/Hardware.h
Hardware initialization 7

inc/Main.h
Main application 8

inc/X2cDataTypes.h
X2C data type definitions 9

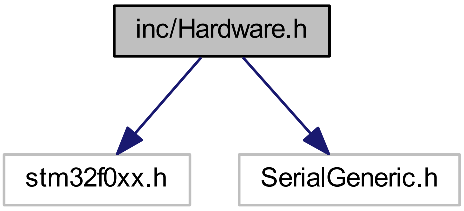
6 File Documentation

6.1 inc/Hardware.h File Reference

Hardware initialization.

`#include <stm32f0xx.h>`
`#include "SerialGeneric.h"`

Include dependency graph for Hardware.h:



Functions

- void initHardware (void)
  Initialization of hardware peripherials.
- void initClock (void)
  Initialization of system clock.
- void initSerial (tSerial ∗serialSTM32F0)
  Initialization of serial interface.
- void ADC1_COMP_IRQHandler (void)
  Interrupt service routine for ADC end of conversion interrupt.

6.1.1 Detailed Description

Hardware initialization.

6.1.2 Function Documentation

6.1.2.1 void initClock (void)

Initialization of system clock.
Configuration:

- Internal high speed clock with PLL
- 48 MHz system clock

6.2 inc/Main.h File Reference

Main application.

Functions

- int main (void)

  Main function.
- void mainTask (void)

  Main control task.

6.2.1 Detailed Description

Main application.

6.2.2 Function Documentation

6.2.2.1 int main (void)

Main function.

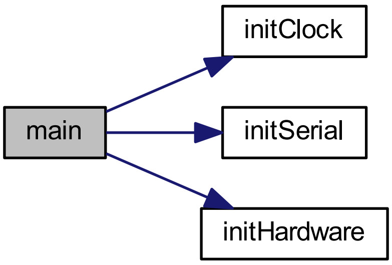
Returns

> The main function will never return due to the never ending loop.

- initialize system clock
- initialize "integrated monitor":
  - configuration of LNet protocol:
    - Node-ID: 1
    - Buffer size: 255
- initialize serial interface
  - configuration of USART2:
  - Baudrate: 115.2kB/s
  - Data bits: 8
  - Parity: none
  - Stop bits: 1
- initialize X2C

- initialize hardware
- never ending loop -> interrupt driven algorithm

Here is the call graph for this function:



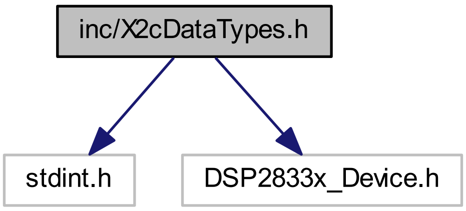
6.2.2.2 void mainTask (void)

Main control task.
Calling rate = 100us

- assign inports
- update X2C
- update outports

6.3 inc/X2cDataTypes.h File Reference

X2C data type definitions.
`#include <stdint.h>`
`#include <DSP2833x_Device.h>`
Include dependency graph for X2cDataTypes.h:



6.3.1 Detailed Description

X2C data type definitions.

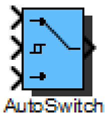
Part III

Used X2C-Blocks

7 Project Specific Blocks

8 Internal Library Blocks

Block: AutoSwitch



Inports	
In1	Input #1
Switch	Input #2: Threshold signal
In3	Input #3

Outports	
Out	Either value of input #1 or input #3 dependent on value of input #2

Mask Parameters	
Thresh_up	Threshold level for rising switch signal
Thresh_down	Threshold level for falling switch signal

Description:

Switch between In1 and In3 dependent on Switch signal:
Switch signal rising: Switch >= Threshold up –> Out = In1
Switch signal falling: Switch < Threshold down –> Out = In3

Implementations:

FiP8	8 Bit Fixed Point Implementation
FiP16	16 Bit Fixed Point Implementation
FiP32	32 Bit Fixed Point Implementation
Float32	32 Bit Floating Point Implementation
Float64	64 Bit Floating Point Implementation

Block: Constant

Outports	
Out	Constant output

Mask Parameters	
Value	Constant factor

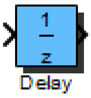
Description:

Constant value.

Implementations:

Bool	Boolean Integration
FiP8	8 Bit Fixed Point Implementation
FiP16	16 Bit Fixed Point Implementation
FiP32	32 Bit Fixed Point Implementation
Float32	32 Bit Floating Point Implementation
Float64	64 Bit Floating Point Implementation

Block: Delay



Inports	
In	Input In(k)

Outports	
Out	Output Out(k)=In(k-1)

Mask Parameters	
ts_fact	Multiplication factor of base sampling time (in integer format)

Description:

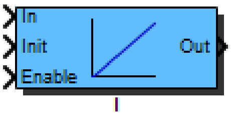
Output delay by one sample time interval.

This block can be used to enable feedback loops in the model.

Implementations:

FiP16	16 Bit Fixed Point Implementation
FiP32	32 Bit Fixed Point Implementation
Float32	32 Bit Floating Point Implementation
Float64	64 Bit Floating Point Implementation

Block: I



Inports	
In	Control error input
Init	Value which is loaded at initialization function call
Enable	Enable == 0: Deactivation of block; Out set to 0 Enable 0->1: Preload of integral part Enable == 1: Activation of block

Outports	
Out	Control value

Mask Parameters	
Ki	Integral Factor
ts_fact	Multiplication factor of base sampling time (in integer format)

Description:

I controller:
G(s) = Ki/s = 1/(Ti*s)

Each fixed point implementation uses the next higher integer datatype for the integrational value storage variable.
A rising flank at the *Enable* inport will preload the integrational part with the value present on the *Init* inport.

Transfer function (zero-order hold discretization method):

$$G(z) \quad = \quad K_I T_s \frac{1}{z-1}$$

Implementations:

FiP8	8 Bit Fixed Point Implementation
FiP16	16 Bit Fixed Point Implementation
FiP32	32 Bit Fixed Point Implementation
Float32	32 Bit Floating Point Implementation
Float64	64 Bit Floating Point Implementation

Block: Negation

Inports	
In	Input

Outports	
Out	Negated input value

Description:

Negation of input signal.

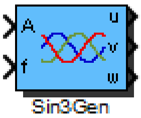
Calculation:

$$Out = -In$$

Implementations:

FiP8	8 Bit Fixed Point Implementation
FiP16	16 Bit Fixed Point Implementation
FiP32	32 Bit Fixed Point Implementation
Float32	32 Bit Floating Point Implementation
Float64	64 Bit Floating Point Implementation

Block: Sin3Gen



Inports	
A	Amplitude
f	Frequency

Outports	
u	Sine wave output phase u
v	Sine wave output phase v
w	Sine wave output phase w

Mask Parameters	
fmax	Maximum Frequency in Hz
Offset	Offset
ts_fact	Multiplication factor of base sampling time (in integer format)

Description:

Generation of a 3 sine waves with amplitude (A) and frequency (f).

Calculation fixed point implementation:

$$\begin{aligned} u_k &= A_k \cdot \sin\left(2 f_k \cdot f_{max} \cdot kT_S\right) + A_{Offset} \\ v_k &= A_k \cdot \sin\left(2 f_k \cdot f_{max} \cdot kT_S - \frac{2\pi}{3}\right) + A_{Offset} \\ w_k &= A_k \cdot \sin\left(2 f_k \cdot f_{max} \cdot kT_S + \frac{2\pi}{3}\right) + A_{Offset} \end{aligned}$$

For sine calculation a lookup table with 256 entries is used. This results in a short computation time but with the downside of reduced accuracy for the FiP32 implementation.

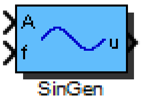
Calculation floating point implementation (parameter *f_max* is ignored):

$$\begin{aligned} u_k &= A_k \cdot \sin\left(2\pi f_k \cdot kT_S\right) + A_{Offset} \\ v_k &= A_k \cdot \sin\left(2\pi f_k \cdot kT_S - \frac{2\pi}{3}\right) + A_{Offset} \\ w_k &= A_k \cdot \sin\left(2\pi f_k \cdot kT_S + \frac{2\pi}{3}\right) + A_{Offset} \end{aligned}$$

Implementations:

FiP8	8 Bit Fixed Point Implementation
FiP16	16 Bit Fixed Point Implementation
FiP32	32 Bit Fixed Point Implementation
Float32	32 Bit Floating Point Implementation
Float64	64 Bit Floating Point Implementation

Block: SinGen



Inports	
A	Amplitude
f	Frequency

Outports	
u	Sine wave output

Mask Parameters	
fmax	Maximum Frequency in Hz
Offset	Offset
Phase	Phase [-Pi..Pi]
ts_fact	Multiplication factor of base sampling time (in integer format)

Description:

Generation of a sine wave with amplitude (A) and frequency (f).

Calculation fixed point implementation:

$$u_k = A_k \cdot \sin\left(2 f_k \cdot f_{max} \cdot kT_S + \phi_{Phase}\right) + A_{Offset}$$

For sine calculation a lookup table with 256 entries is used. This results in a short computation time but with the downside of reduced accuracy for the FiP32 implementation.

Calculation floating point implementation (parameter *f_max* is ignored):

$$u_k = A_k \cdot \sin\left(2\pi f_k \cdot kT_S + \phi_{Phase}\right) + A_{Offset}$$

Implementations:

FiP8	8 Bit Fixed Point Implementation
FiP16	16 Bit Fixed Point Implementation
FiP32	32 Bit Fixed Point Implementation
Float32	32 Bit Floating Point Implementation
Float64	64 Bit Floating Point Implementation