Speed Up Automotive, Industrial, and IoT Applications with NXP Model-Based Design Toolbox

Daniel Scurtu
A POSITION OF STRENGTH TO BETTER SERVE OUR 26,000+ CUSTOMERS

We accelerate breakthroughs that advance the world through our semiconductor technology leadership

EMPLOYEES IN

30+ COUNTRIES

~29,000
EMPLOYEES

Headquartered in Eindhoven, Netherlands

60+
Year History

~11,000
Engineers

9,500
Patent Families

$8.61B
Annual Revenue

1 Posted revenue for 2020 – Please refer to the Financial Information page of the Investor Relations section of our website at www.nxp.com/investor for additional information
A POSITION OF STRENGTH TO BETTER SERVE OUR 26,000+ CUSTOMERS

We accelerate breakthroughs that advance the world through our semiconductor technology leadership

EMPLOYEES IN

30+ COUNTRIES

Headquartered in Eindhoven, Netherlands

~29,000

EMPLOYEES

60+

Year History

~11,000

Engineers

9,500

Patent Families

$8.61B

Annual Revenue

1 Posted revenue for 2020 – Please refer to the Financial Information page of the Investor Relations section of our website at www.nxp.com/investor for additional information
MOBILE

Giving wearable and mobile devices easier access to the services that make modern life more convenient without compromising security and safety.

Transforming how people and devices connect

COMMUNICATION INFRASTRUCTURE

Powering insights and inspiring performance with hardware solutions for handling 5G connectivity across the emerging communications spectrum.

Delivering real-time responsiveness at the speed of 5G

AUTOMOTIVE

Enabling carmakers to develop smarter solutions for complex autonomy, connectivity, and electrification challenges

Accelerating the shift to greater mobility

SMART CITY

Simplifying how people access and interact with local services to achieve new standards of sustainability, efficiency, mobility, and economic growth.

Anticipating the demands of tomorrow

INDUSTRIAL

Reducing wasted time, money, and effort by helping business run more efficiently.

Enabling more efficient data processing
MOBILE
Giving wearable and mobile devices easier access to the services that make modern life more convenient without compromising security and safety.

Transforming how people and devices connect

COMMUNICATION INFRASTRUCTURE
Powering insights and inspiring performance with hardware solutions for handling 5G connectivity across the emerging communications spectrum.

Delivering real-time responsiveness at the speed of 5G

SMART CITY
Simplifying how people access and interact with local services to achieve new standards of sustainability, efficiency, mobility, and economic growth.

Anticipating the demands of tomorrow

INDUSTRIAL
Reducing wasted time, money, and effort by helping business run more efficiently.

Enabling more efficient data processing

NXP - EDGE PROCESSING

NXP
Real Time Drivers

SW Libraries

Documentation

Enablement Tools

Reference Design Solution
MCUXpresso – IDE – Debugger - Toolchains

Pins Tool
Clocks Tool
Peripheral Tool

FreeMASTER
FreeMASTER Lite

Serial Connection - UART
Debug Connection - JTAG
MCUXpresso – IDE – Debugger - Toolchains

- Pins Tool
- Clocks Tool
- Peripheral Tool

FreeMASTER

FreeMASTER Lite

Serial Connection - UART

Debug Connection - JTAG
MCUXpresso - Pin – Clock - Peripheral Tools

MODEL-BASED DESIGN TOOLS

Why do customers ask for MBDT for NXP's PARTS?

FreeMASTER

FreeMASTER Lite

Serial Connection - UART

Debug Connection - JTAG

IMX RT1060 EVK
MCUXpresso - Pin – Clock - Peripheral Tools

Model-Based Design Tools for Simulink

Integrate
Design
Prototype
Deploy
Test

Why do customers ask for MBDT for NXP’s PARTS?

FreeMASTER
FreeMASTER Lite

Serial Connection - UART
Debug Connection - JTAG

I.MX RT1060 EVK
FAST - Time To Market

EASY To Use - Reuse

SIMULATION

RICH MATH IP/Libraries

LEVERAGE NXP Eco-System

MCUXpresso - Pin – Clock - Peripheral Tools

FreeMASTER

FreeMASTER Lite

Serial Connection - UART

Debug Connection - JTAG
BATTERY MANAGEMENT SYSTEM

S32K142 + MC33772B

NXP BATT – 6EMULATOR
BATTERY MANAGEMENT SYSTEM

S32 Design Studio + Pin – Clock - Peripheral Tools

Serial Connection - UART

Debug Connection - JTAG

BMS Monitor

FreeMASTER

FreeMASTER Lite

Model-Based Design Tools for Simulink
NXP – MATLAB SIMULINK TOOLBOXES

A COMPLETE SOLUTION
NXP Support Package MPC574x version 1.0.0 by NXP Model-Based Design Toolbox Team
This package represents the MATLAB installer add-on for the FREE of cost NXP Model-Based Design Toolboxes.
Following are the steps to install NXP's Model-Based Design Toolboxes for MPC574x Automotive Microcontrollers family 1. Go to Add-On Manager in MATLAB and select the NXP Support Package MPC574x.

NXP Support Package S32K1xx version 2.0.0 by NXP Model-Based Design Toolbox Team
This package represents the MATLAB installer add-on for the FREE of cost NXP Model-Based Design Toolboxes.
Note: It is recommended to install and delete all files/folders of the older versions of NXP Model-Based Design S32K1xx Toolboxes before installing the new versions.
The steps to install NXP Support Package S32K1xx:
1. Go to Add-On Manager and select the NXP Support Package S32K1xx toolbox.
2. Select Open Folder option.

NXP Support Package S32V234 version 2.0.0 by NXP Model-Based Design Toolbox Team
This package represents the MATLAB installer add-on for the NXP Vision Toolboxes for S32V234.
To install the NXP Vision Toolbox for S32V234 Automotive Vision Processor, please follow these steps:
1. Go to Add-On Manager and select the NXP Support Package S32V234 toolbox.
2. Select Open Folder option.

NXP Support Package K4Vx version 1.0.0 by NXP Model-Based Design Toolbox Team
This package represents the MATLAB installer add-on for the FREE of cost NXP Model-Based Design Toolboxes.
Following are the steps to install NXP's Model-Based Design Toolboxes for K4Vx Series of MCUs:
1. Go to Add-On Manager in MATLAB and select the NXP Support Package K4Vx toolbox.
2. Select Open Folder option.

NXP Support Package S32R version 1.5.0 by NXP Model-Based Design Toolbox Team
This package represents the MATLAB installer add-on for the NXP RADAR Toolbox for S32R.
To install the NXP RADAR Toolbox for S32R, please follow these steps:
1. Go to Add-On Manager and select the NXP Support Package S32R toolbox.
2. Select Open Folder option to navigate to the Toolbox.

NXP Support Package S12ZVMK version 1.0.0 by NXP Model-Based Design Toolbox Team
This package represents the MATLAB installer add-on for the FREE of cost NXP Model-Based Design Toolboxes.
Note: It is recommended to uninstall and delete all files/folders of the older versions of the NXP Model-Based Design S12ZVMK Toolbox before installing the new versions.
The steps to install NXP Support Package S12ZVMK:
1. Go to Add-On Manager in MATLAB and select the NXP Support Package S12ZVMK toolbox.

NXP Support Package IMXRT1xxxx version 1.1.0 by NXP Model-Based Design Toolbox Team
This package represents the MATLAB installer add-on for the FREE of cost NXP Model-Based Design Toolboxes.
Following are the steps to install NXP Model-Based Design Toolboxes for IMXRT from Crossover MCUs:
1. Go to Add-On Manager in MATLAB and select the NXP Support Package IMXRT1xxxx toolbox.
NXP Model-Based Design Tools

Model-Based Design Tools for Matlab and Simulink Support

S32K1xx
- How to
- Tutorials
- Videos

MPC57xx
- How to
- Tutorials
- Videos

S12ZVM
- How to
- Tutorials
- Videos

i.MX RT
- How to
- Tutorials
- Videos

Kinetis V
- How to
- Tutorials
- Videos

Other Solutions
- Tips and Tricks

The Model-Based Design Toolbox provides an integrated development environment and toolchain for configuring and generating all of the

# NXP Model-Based Design Tools

## Model-Based Design Tools for Matlab and Simulink Support

### S32K1xx
- How to
- Tutorials
- Videos

### MPC57xx
- How to
- Tutorials
- Videos

### S12ZVM
- How to
- Tutorials
- Videos

### i.MX RT
- How to
- Tutorials
- Videos

### Kinetis V
- How to
- Tutorials
- Videos

### Other Solutions
- Tips and Tricks

The Model-Based Design Toolbox provides an integrated development environment and toolchain for configuring and generating all of the
MATLAB Simulation: RADAR Target Generation, Signal Acquisition, CNN Training & Predictions

Scenario 1: Human Locomotion

UWB Micro Doppler Signature

- NXP BOOTH
- NXP.COM
- NXP COMMUNITY
Thank you
i.MX RT1060 EVK
Vector Control of PMSM

Slow Control Loop
- Speed req
- PI Controller RTCESL
- Merge
- Open Loop Startup
- \( \omega \)
- \( \omega_{OL} \)
- \( \theta_{OL} \)

Fast Control Loop
- \( i_{sd} \)
- \( u_{sd} \)
- \( u_{\text{scl}} \)
- SVM RTCESL
- ADC
- \( \theta \)
- \( \theta_{\text{EST}} \)
- Tracking Observer RTCESL
- Back-EMF Observer RTCESL
- \( u_{\text{sm}} \)
- \( i_{sq} \)
- \( u_{sq} \)
- \( i_{sp} \)
- \( u_{sp} \)
- \( \alpha \beta / \text{abc} \)
- \( \text{PWM} \)
- \( \text{VSI} \)
- \( u_{DC} \)
- \( \text{PMSM} \)

Peripheral Hardware

NXP

NXP
MCU Configurations: Core, Systems and Peripherals

MCU Inputs

Applications

Algorithms

Kernels

(HW Independent)
/*
 * File: e6l_main.c
 * Code generated for Simulink model 'evkimxrt1060_gpio_polling'.
 * Model version : 1.02
 * Simulink Coder version : 9.4 (R2020b) 29-Jul-2020
 * MBDT for iMXRT Series Version : Version 1.1.0 (R2019a-R2020b) 12-October-2020
 * C++ source code generated on: Thu Oct 8 16:44:14 2020
 *
 * Target selection: e6l.tlc
 * Embedded hardware selection: ARM Cortex
 * Code generation objective: Execution efficiency
 * Validation result: All passed
 */

#include "evkimxrt1060_gpio_polling.h"
#include "rtwtypes.h"

volatile int IsrOverrun = 0;
static boolean_T OverrunFlag = 0;

void rt_OneStep(void)
{
    /* Check for overrun. Protect OverrunFlag against preemption */
    if (OverrunFlag++)
    {
        IsrOverrun = 1;
        OverrunFlag = 0;
        return;
    }

    evkimxrt1060_gpio_polling_step();
}
Clocks Diagram

Run Mode: RUN
On-chip oscillator 24MHz Enabled

Clock Sources
- RTC OSC mode: Enabled
- 24MHz clock source: Enabled
- Power down mode: Disabled
- OSC mode: External crystal 24MHz
- RC oscillator: Disabled
- OSC ready...inter_value: 128
- CLK1 external clock source: Enabled
- CLK1 pins mode: Enabled
- SAI1 MCLK: Disabled

Clock Outputs
- GPT1 high frequency clock: 75 MHz
- GPT2 high frequency clock: 75 MHz
- LCDIF Clock: 67.5 MHz
- LPRD_CLXROOT: 60 MHz
- LPSPI_CLKROOT: 105.6 MHz
- LVDS1_CLK: 1.2 MHz
- MOS MCLK: 63.52 MHz
- PERCLK_CLK_ROOT: 100 MHz (75 MHz ±0.1%)

PERCLK_CLK_ROOT (peripheral clock output)
- Peripheral main clock
- Peripheral clock
- APB divider
- AHd divider Frequency: 600 MHz
- IPG divider Frequency: 150 MHz
- PERCLK clock selector
- IPG root clock
- PERCLK clock divider
- PERCLK clock divider Frequency: 75 MHz
- PERCLK root clock

CONSUMED BY:
- GPT1, GPT2, PIT

ERRORS:
- The requirement cannot be satisfied or is in conflict with other requirements. Try to select different clock path/option or unlock/change some of the original settings.
- 75 MHz
- REQUIRED: 100 MHz ±0.1%

VALUE: 75 MHz
Model-based design (MBD) is a mathematical and visual method of addressing problems associated with designing complex control, signal processing and communication systems. It is used in many motion control, industrial equipment, aerospace, and automotive applications. 

Model-based design - Wikipedia
Step 1 – System Requirements
Model-in-the-Loop
- Software requirements
- Control system requirements
- Overall application control strategy

Step 2 – Modeling/Simulation
Software-in-the-Loop
- Control algorithm design
- Code generation preparation
- Control system design
- Start testing algorithm

Step 3 – Rapid Prototype
Processor-in-the-Loop
- Controller code generation
- Determine execution time on MCU
- Verify algorithm on MCU
- See memory/stack usage on MCU

Step 4 – Target MCU Implementation
MCU Final Application
- Validation/verification phase
- Controller code generation
- Test system in target environment using tools for data logging and parameter tuning
MATHWORKS EXTENSIVE TOOLS ECOSYSTEM

Control & Design

- **Control Design**
  - Tuning & Auto tuning, Frequency Response

- **Simscape**
  - Motors, IGBT, Diodes, Thermal, Spice

Event Based Modeling

- **Stateflow**
  - Logic design, Scheduling

Code Generation & Interfacing

- **Coders**
  - Code generation, legacy code integration, XCP, HIL, PIL, External mode

- **Tuning & Visualizing**
  - Fault-Tolerant Fuel Control System Dashboard

Certification & Validation

- **IEC Certification Kit**
  - IEC 61508, ISO26262, EN50128, IEC62304

- **Simulink Requirements**
  - Requirements, Trace PRD, Track Status
```c
#include "S32K144.h"

#if defined (__ghs__)
#define __INTERRUPT_SVC __Interrupt
#define __NO_RETURN __Pragma("ghs warning 111")
#define __interrupt __Pragma("diag suppress Pre11")
#endif

#define __attribute__((interrupt("SVC")))
#endif

int counter, accumulator = 0, limit_value = 1000000;

int main(void)
{
    counter = 0;

    for (; ; )
    {        
        counter++;

        if (counter >= limit_value)
        {          
            __asm__ volatile( "svc @0;" );
            counter = 0;
        }

        /* to avoid the warning message for GHS and JAR: statement is unreachable*/
        __NO_RETURN
        return 0;
    }

    __INTERRUPT_SVC void SVC_Handler()
    
    accumulator = counter;
```