LPC51U68 CPU Support Package Guide

Version: 4.0

Contents

LPC51U68 Support Package

This guide describes the following features of the LPC51U68 CPU support package:

- How to create LPC51U68 projects
- How to open LPC51U68 sample projects
- LPC51U68 specific project properties
- LPC51U68 specific project templates
- Supported LPC51U68 devices

Creating LPC51U68 Projects

Creating an LPC51U68 C/C++ executable project

To create a new minimal C/C++ executable project:

Select the **File** > **New** > **New Project** menu item.
Select the **A C/C++ executable for NXP LPC51U68** project template.
Set the required project name and location directory.
Click **Next**.
If required, change any of the default project settings.
Click **Finish** to create the project.

Creating an LPC51U68 library project

To create a new library project:

Select the **File** > **New** > **New Project** menu item.
Select the **A library for NXP LPC51U68** project template.
Set the required project name and location directory.
Click **Next**.
If required, change any of the default project settings.
Click **Finish** to create the project.

Creating an LPC51U68 externally built executable project

To create a new project that will allow you to debug an existing externally built executable file:

Select the **File** > **New** > **New Project** menu item.
Select the **An externally built executable for NXP LPC51U68** project template.
Set the required project name and location directory.
Click **Next**.
Set the **Load File** project property to point to the executable file you want to download and debug.
If required, change any of the other default project settings.
Click **Finish** to create the project.

Creating an LPC51U68 CrossWorks Tasking Library executable project

To create a new C/C++ executable project configured to use the CrossWorks Tasking Library:

Select the **File** > **New** > **New Project** menu item.
Select the **A CrossWorks Tasking Library executable for NXP LPC51U68** project template.
Set the required project name and location directory.
Click **Next**.

If required, change any of the other default project settings.
Click **Finish** to create the project.

Creating an LPC51U68 assembly code only executable project

Please note, this template does not add any C/C++ startup code or libraries and is therefore not suitable for creating projects that include C/C++ code.

To create a new assembly code only executable project without:

Select the **File** > **New** > **New Project** menu item.
Select the **An assembly code only executable for NXP LPC51U68** project template.
Set the required project name and location directory.
Click **Next**.
If required, change any of the other default project settings.
Click **Finish** to create the project.

Opening LPC51U68 Sample Solutions

LPC51U68 Samples Solution

This solution contains general sample projects that run on LPC51U68 devices. To open the LPC51U68 Samples Solution:

Select the **Tools** > **Show Installed Packages** menu item.
Select the **NXP LPC51U68 CPU Support Package** link.
Select the **Samples Solutions** > **LPC51U68 Samples Solution** link.

LPC51U68 CMSIS-DSP Samples Solution

This solution contains sample projects that use the CMSIS-DSP library running on LPC51U68 devices. To open the LPC51U68 CMSIS-DSP Samples Solution:

Select the **Tools** > **Show Installed Packages** menu item.
Select the **NXP LPC51U68 CPU Support Package** link.
Select the **Sample Solutions** > **LPC51U68 CMSIS-DSP Samples Solution** link.

LPC51U68 Project Properties

Projects creating using the project templates in this support package have the following device specific project properties:

Heap Size

The heap size is set to be 256 bytes when a project is created. The heap size can be modified using the **Heap Size** project property.

Section Placement

You can select the memory configuration you require using the **Section Placement** project property.

For LPC51U68 projects, the set of placements are:

Flash - The application runs in internal Flash memory *(default)*.
Flash Vectors In RAM - The application runs in internal Flash memory and exception vectors are copied to RAM memory.
Flash Copy To RAM - The application starts in internal flash and copies itself to run from internal RAM memory.
RAM - The application runs from internal RAM memory only.

Stack Sizes

The main stack size is set to be 256 bytes when a project is created.

The process stack size is set to be 0 bytes when a project is created.

The main and process stack sizes can be modified using the **Main Stack Size** and **Process Stack Size** project properties.

To change the location of the stacks, edit the section placement file and place the *.stack* and *.stack_process* sections as required.

Startup From Reset

By default, the application will only startup from power-on/reset in *Release* configuration. This acts as a safety net in case you accidently download a program in FLASH during development that crashes and prevents the debugger from taking control of the target over the debug interface thus rendering the device unusable.

For LPC51U68 projects, the **Startup From Reset** project property can be set to one of the following:

No - The application will not startup from reset.

Release Only - The application will only startup from reset when built in *Release* configuration *(default)*.
Yes - The application will always startup from reset.

Target Processor

Once a project has been created you can target different devices by modifying the **Target Processor** project property. See the **LPC51U68 Devices** section for details on the files, preprocessor definitions and macro definitions used when a device is selected.

LPC51U68 Project Templates

The project template system simplifies the creation of new projects with the IDE, it also system makes it easy to create new projects with a text editor or script. All that needs to be specified is the project name, the support packages that the project is dependent on, the target processor and the source files you want to add to the project. For example, create a file called *example.hzp* with the following contents:

```
<!DOCTYPE CrossStudio_Project_File>
<solution Name="Example Solution">
  <project Name="Example Project" template_name="LPC51U68_EXE">
    <configuration Name="Common" package_dependencies="LPC51U68" Target="LPC51U68JBD64" />
    <folder Name="Source Files">
      <file file_name="file1.c" />
      <file file_name="file2.c" />
    </folder>
  </project>
</solution>
```

You can also add any other property settings that the project requires such as preprocessor definitions or include paths using the property save name, for example:

```
<!DOCTYPE CrossStudio_Project_File>
<solution Name="Example Solution">
  <project Name="Example Project" template_name="LPC51U68_EXE">
    <configuration Name="Common" package_dependencies="LPC51U68" Target="LPC51U68JBD64"
 c_preprocessor_definitions="MYDEF1=1;MYDEF2=TWO" c_user_include_directories="$(ProjectDir)/
include1;$(ProjectDir)/include2" />
    <folder Name="Source Files">
      <file file_name="file1.c" />
      <file file_name="file2.c" />
    </folder>
  </project>
</solution>
```

Available LPC51U68 project templates

Template Name	Template Description
LPC51U68_ASM_EXE	LPC51U68 Assembly Code Only Executable
LPC51U68_CTL_EXE	LPC51U68 CTL Executable
LPC51U68_EXE	LPC51U68 C/C++ Executable
LPC51U68_EXT_EXE	LPC51U68 Externally Built Executable
LPC51U68_LIB	LPC51U68 Library

LPC51U68 Devices

This package supports the following LPC51U68 devices:

- LPC51U68JBD48
- LPC51U68JBD64

LPC51U68JBD48

Device Details	
CMSIS Header File	$(TargetsDir)/LPC51U68/CMSIS/fsl_device_registers.h
CMSIS Include Path	$(TargetsDir)/LPC51U68/CMSIS/.
CMSIS System File	$(TargetsDir)/LPC51U68/CMSIS/system_LPC51U68.c
Family	LPC51U68
Loader File	$(TargetsDir)/LPC51U68/Loader/LPC51U68_Loader.elf
Memory Map File	$(TargetsDir)/LPC51U68/XML/ LPC51U68JBD48_MemoryMap.xml
Register Definition File	$(TargetsDir)/LPC51U68/XML/LPC51U68_Registers.xml
Vectors File	$(TargetsDir)/LPC51U68/Source/LPC51U68_Vectors.s

Preprocessor Definitions
ARM_MATH_CM0PLUS
CPU_LPC51U68JBD48
__LPC51U68_FAMILY

Memory Segments	
FLASH	0x00000000 - 0x0003FFFF
RAM	0x20000000 - 0x2000FFFF
RAM2	0x04000000 - 0x04007FFF

Project Macros
DeviceIncludePath=$(TargetsDir)/LPC51U68/CMSIS/.
DeviceHeaderFile=$(TargetsDir)/LPC51U68/CMSIS/fsl_device_registers.h
DeviceLoaderFile=$(TargetsDir)/LPC51U68/Loader/LPC51U68_Loader.elf
DeviceRegisterDefinitionFile=$(TargetsDir)/LPC51U68/XML/LPC51U68_Registers.xml
DeviceSystemFile=$(TargetsDir)/LPC51U68/CMSIS/system_LPC51U68.c
DeviceVectorsFile=$(TargetsDir)/LPC51U68/Source/LPC51U68_Vectors.s
DeviceFamily=LPC51U68

LPC51U68JBD64

Device Details	
CMSIS Header File	$(TargetsDir)/LPC51U68/CMSIS/fsl_device_registers.h
CMSIS Include Path	$(TargetsDir)/LPC51U68/CMSIS/.
CMSIS System File	$(TargetsDir)/LPC51U68/CMSIS/system_LPC51U68.c
Family	LPC51U68
Loader File	$(TargetsDir)/LPC51U68/Loader/LPC51U68_Loader.elf
Memory Map File	$(TargetsDir)/LPC51U68/XML/LPC51U68JBD64_MemoryMap.xml
Register Definition File	$(TargetsDir)/LPC51U68/XML/LPC51U68_Registers.xml
Vectors File	$(TargetsDir)/LPC51U68/Source/LPC51U68_Vectors.s

Preprocessor Definitions
ARM_MATH_CM0PLUS
CPU_LPC51U68JBD64
__LPC51U68_FAMILY

Memory Segments	
FLASH	0x00000000 - 0x0003FFFF
RAM	0x20000000 - 0x2000FFFF
RAM2	0x04000000 - 0x04007FFF

Project Macros
DeviceIncludePath=$(TargetsDir)/LPC51U68/CMSIS/.
DeviceHeaderFile=$(TargetsDir)/LPC51U68/CMSIS/fsl_device_registers.h
DeviceLoaderFile=$(TargetsDir)/LPC51U68/Loader/LPC51U68_Loader.elf
DeviceRegisterDefinitionFile=$(TargetsDir)/LPC51U68/XML/LPC51U68_Registers.xml
DeviceSystemFile=$(TargetsDir)/LPC51U68/CMSIS/system_LPC51U68.c
DeviceVectorsFile=$(TargetsDir)/LPC51U68/Source/LPC51U68_Vectors.s
DeviceFamily=LPC51U68