Speed Up Development With C2000™ Real-Time MCUs Using SysConfig

Nima Eskandari
Applications Engineer
C2000 Real-Time MCUs
This white paper explains how the C2000 SysConfig graphical user interface (GUI) tool is developed to facilitate the development process for designers.

At a glance

1. **C2000 SysConfig system initialization code generation can speed up your software development**
   - Reliable and pre-verified code generated by the C2000 SysConfig tool is the main feature that facilitates software development for designers. Device configuration errors are caught by tool and the developer is notified of the unsupported setting.

2. **Integrated PinMux tool and PinMux initialization code generation closes the gap between hardware and software designers**
   - The C2000 SysConfig tool has built-in support for solving the device PinMux against system requirements, and provides device visualization aids.

3. **Portable device initialization and code generation allows developers to create flexible and easily modifiable code**
   - The device configuration developed in C2000 SysConfig can be easily ported between device families. Also, modifications in the device configuration can be easily propagated through the rest of the already developed application code.

Designers utilizing C2000™ Real-Time MCU can configure their device using C2000 SysConfig that generates reliable code and visual aids to simplify and speed up the development process. C2000 SysConfig catches incorrect device initialization settings which significantly facilitates the debugging process.

Developing embedded software for initializing an MCU can be difficult for inexperience and experienced designers. New embedded devices with more advanced features and peripherals enter the market every year, making the task of configuring the device much more demanding. Small mistakes in configuring the device and its peripherals can cause the development process to slow down and frustrate the designers.

With advancements in embedded technology, both the size and capabilities of the embedded devices have grown. This makes the task of device recourse management very important. Knowing which peripherals and which device pins are used and which ones are available for further development is useful for both hardware and software designers.

Selecting the correct family of devices for a specific application can also be difficult because the requirement for the system can change. Portability of initialization settings and embedded software between device families eases the minds of developers, because they can reuse their previous investment in software development when migrating between different devices.

This white paper explores how the C2000 SysConfig tool can simplify the tasks that embedded hardware and software developers face when designing a system with C2000 real-time MCUs.

**C2000 SysConfig system initialization code generation can speed up your software development**

Developing initialization code is the first step in software development for an embedded application. This can be a tedious task and many mistakes are made due to small coding errors and not following the exact instructions given in the device’s technical reference.
Manual. Once the device is correctly initialized, the designer can continue on to develop the software for their specific application. Confidence in the correctness of the device initialization is much higher when the code developed has been previously verified by device experts. C2000 real-time MCUs can be initialized through C2000 SysConfig which generates reliable and pre-verified code for configuring your device. Device configuration errors are caught by the tool, and the developer is notified of the unsupported settings. C2000 SysConfig tool can also configure the device PinMux and visualize the device pins for each package.

C2000 SysConfig is delivered through C2000Ware (C2000 real-time MCU software development kit) and can be used with Code Composer Studio™ (CCS) IDE’s built-in SysConfig (System Configuration) tool or with any other supported IDE through the SysConfig tool’s standalone version.

Peripheral configuration

The C2000 SysConfig allows designers to configure their peripherals through the SysConfig GUI.

Device peripherals are listed in the C2000 SysConfig tool so the designer is aware of the peripherals available in their specific device package. The configurable options for each peripheral is listed, which allows the designer to see all the different available modes. The device level inter-connects are displayed in the tool that shows the available list of signals for each MUX previously described only in the technical documentation.

More complicated peripherals such as the Configurable Logic Block (CLB), which is capable of creating custom logic inside the device, or the Dual Code Security Module (DCSM), which is used of securing the customer’s intellectual property, are also included in the C2000 SysConfig ecosystem. These add-on tools will show up
automatically in the tool and the designer has the options of using them in their application. The additional auto-generated artifacts from these tools will be presented to the designer seamlessly.

**Code generation**

Automatic code generation allows developers to be confident in the correctness of their initialization code for their embedded application. As configurable options are changed, the designers can view the differences in the auto-generated code. This allows developers to understand how the embedded software changes when the requirements for the device configuration are altered.

C2000 SysConfig's auto-generated code is structured so that the designer can pick which section of the code they wish to add to their application. The designer can choose to add all of the auto-generated code or they can choose to use only the PinMux initialization feature, or they can choose to use only the initialization for a specific peripheral. This eliminates the concerns that an experienced designer would have for adopting this tool, when they already have existing and verified code. Designers who already have existing code, can compare their initialization code with the auto-generated one by C2000 SysConfig as a verification step since the auto-generated code was designed by C2000 real-time MCU experts.

*Figure 3. C2000 SysConfig CCS project overview.*
Figure 4. Code generation and difference identification.
Error detection

Embedded devices often have many supported modes, but the device must be configured exactly as instructed by the technical documentation for each mode to operate correctly. Also, the device silicon Errata documentation notes the unsupported modes which sometimes are missed by the designers.

It is common that the development process for configuring a device is slowed down due to errors in the designer’s code. These errors could be due to mistakes in programming when transferring knowledge from the technical documentation into the application software. C2000 SysConfig is capable of catching configuration errors and notifying the user of the incorrect setup.

Also, similar to error generation, warnings are generated as needed when a configuration is not necessarily wrong but requires further attention.

Device level dependencies

Device level dependencies can be missed by developers. Not configuring all dependencies required for a peripheral to operate correctly is a common mistake. The C2000 SysConfig tool identifies the dependencies in the device and ensures that these dependencies are configured by the designer.

C2000 real-time MCUs are highly configurable and their inter-connections, which reduces signal chain latency and eliminates requirements for external components, create dependencies between peripherals inside the device. For example, the Analog-to-Digital Converter (ADC), the enhanced Pulse Width Modulator (ePWM) and Comparator Subsystem (CMPSS) could all be inter-connected to one another inside the device.

Figure 5. Error detection.

Figure 6. Identifying dependencies.
Device level error detection

Error detection in the designer’s configuration is **NOT** limited to one peripheral at a time. Incorrect setups can be detected across dependent modules. This ensures that all dependent peripherals are configured correctly.

**Figure 7. Detecting errors in dependencies.**

**Integrated PinMux tool and PinMux initialization code generation closes the gap between hardware and software designers**

Solving the device PinMux for a given application can be a difficult task. Resource management, knowing which peripherals and pins have been used and which ones are free, is extremely important in determining if the MCU is capable of all of the application’s requirements. The C2000 SysConfig tool has built-in support for device PinMux, and can solve the device PinMux, and also auto-generates PinMux initialization code along with a summary CSV file. C2000 SysConfig will also visualize the device package and show the used and free package pins.

**Figure 8. Device package PinMux support.**
The pinmux.csv file bridges the gap between the hardware and software designer.

**Portable device initialization and code generation allows developers to create flexible and easily modifiable code**

The device configuration set in C2000 SysConfig for a specific device family and package can be ported to other device families and packages. When migrating between device families, C2000 SysConfig will automatically update it’s code generation templates and output the correct embedded software for the new device family. This can simplify the task of porting device initialization code for designers who are migrating between device families.

C2000 SysConfig also allows designers to name their device resources to their application specific name. This allows the initialization code and the runtime application code to be much more flexible to changes in PinMux and resource instances. For example, if the designer decides to change the GPIO number used for a task in the application, no change would be necessary to the runtime application code if the task GPIO was named through C2000 SysConfig.

In the case that some resources are not available when migrating between device families or device packages, the user is notified through warning and errors generated by the tool. If the tool is able to reassign a new resource, which can accomplish the same task, the designer is notified through a warning. If the tool is NOT able to reassign a new resource, the tool generates an error and forces the designer to either remove the requirement or reassign a new resource manually, which meets the same requirements.
Figure 10. Migrating between device families and device packages.

Conclusion
C2000 SysConfig is a powerful graphical user interface tool that configures the C2000 real-time MCUs and auto-generates embedded software, visualization diagrams, and debug artifacts that helps designers significantly with their development process. The reliable and pre-validated initialization software generated by the C2000 SysConfig tool can speed up development and help designers avoid lengthy debug sessions.

Additional resources
- TI Cloud Tools
  - SysConfig
  - Resource Explorer
- C2000Ware for C2000 MCUs
  - C2000 SysConfig and examples for C2000 real-time MCUs
- Code Composer Studio (CCS)
  - Integrated development environment (IDE) that supports TI’s Microcontroller and Embedded Processors
  - SysConfig tool is delivered integrated in CCS (built-in SysConfig support)
- SysConfig Standalone Version
  - SysConfig standalone version can be used alongside other IDEs which do not have the built-in SysConfig tool
- Texas Instruments: C2000 SysConfig
  - Step by step instructions for C2000 SysConfig
- C2000 SysConfig Lab 0
IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI's products are provided subject to TI's Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI's provision of these resources does not expand or otherwise alter TI's applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.