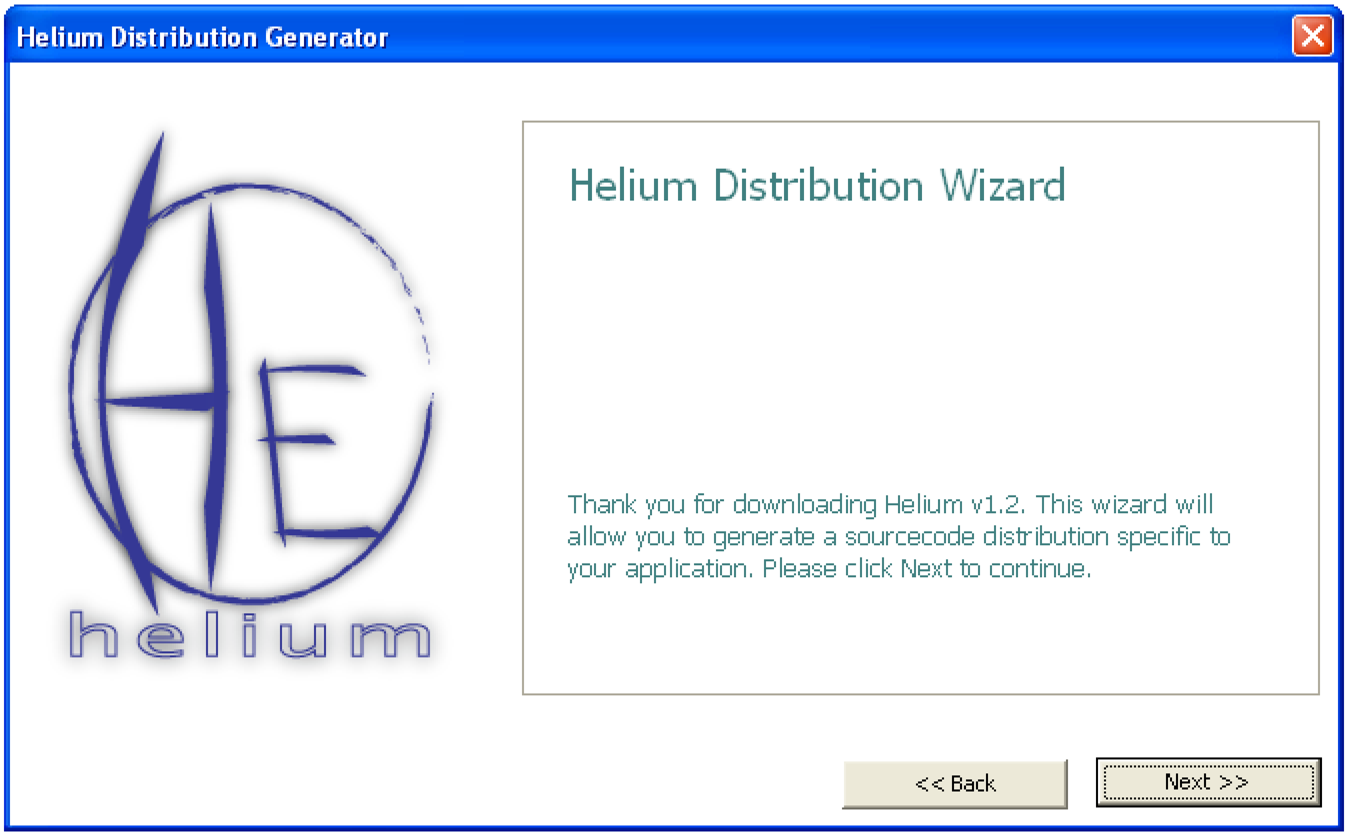
helium

helium configuration utility readme v1.2

The Helium Configuration Wizard organizes the Helium source code distribution in an easy to use fashion. It creates prototypes for all the tasks that the user code will need and generates boot code for the microcontroller that initializes all of the user's tasks on startup using the `TaskCreate` API function. For an explanation of a task's structure in the Helium RTOS, please refer to the Helium README file available in the Documentation section of the website, http://helium.sourceforge.net.

Configuring Helium

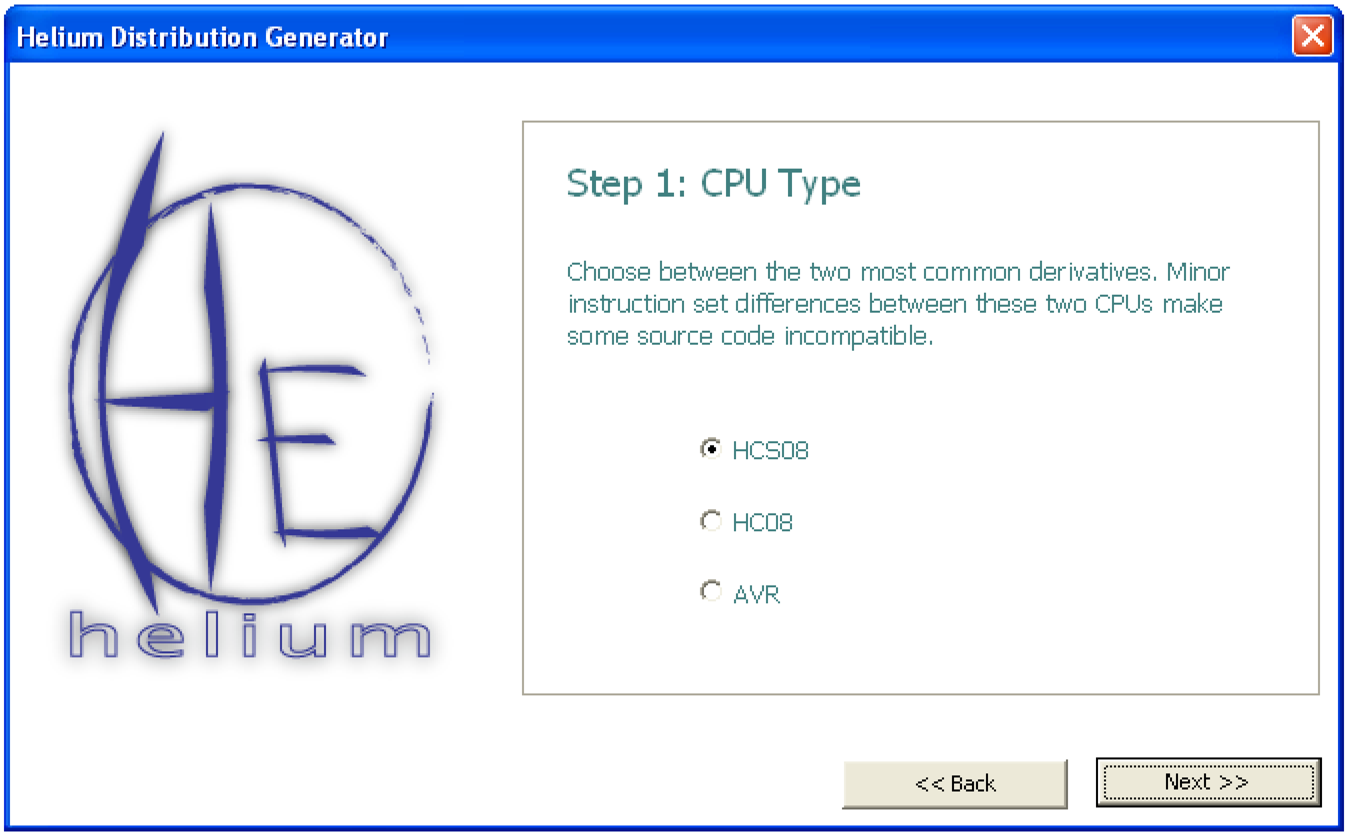
When the configuration wizard is first opened, it will display a welcome screen like the one below. Click (Next >>) to continue.



Helium Distribution Wizard

Thank you for downloading Helium v1.2. This wizard will allow you to generate a sourcecode distribution specific to your application. Please click Next to continue.

The first step allows the user to select his CPU type. Click (Next >>) to continue.



Helium Distribution Generator

Step 1: CPU Type

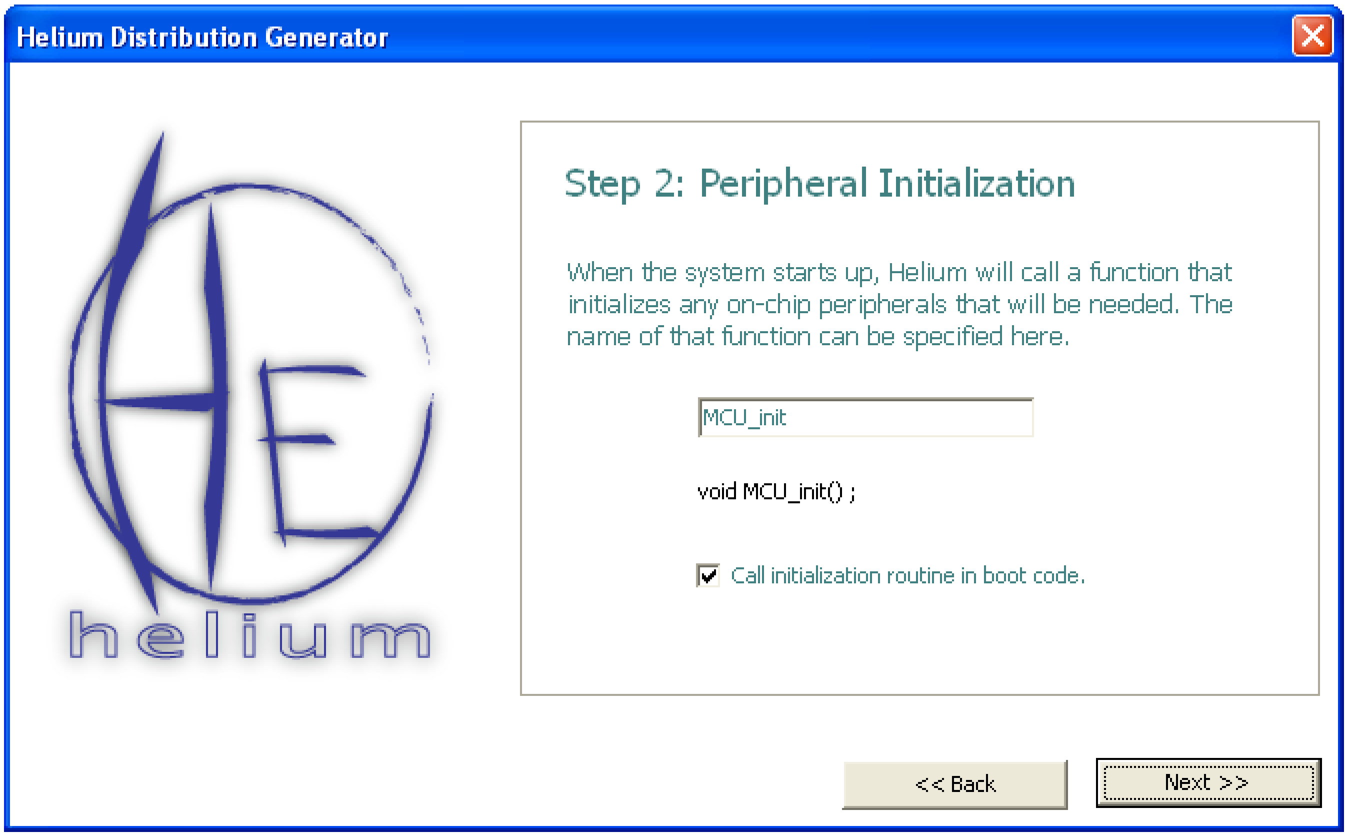
Choose between the two most common derivatives. Minor instruction set differences between these two CPUs make some source code incompatible.

[✓ HCS08]

HC08

AVR

The second step allows the user to specify a function that will initialize peripheral devices on his MCU. If this option is enabled, Helium will simply call the function specified in Step 2 on bootup. The user is responsible for writing that function. The user may also choose not to use this feature and place all init code inside Helium's `main` function.



Step 2: Peripheral Initialization

When the system starts up, Helium will call a function that initializes any on-chip peripherals that will be needed. The name of that function can be specified here.

MCU_init

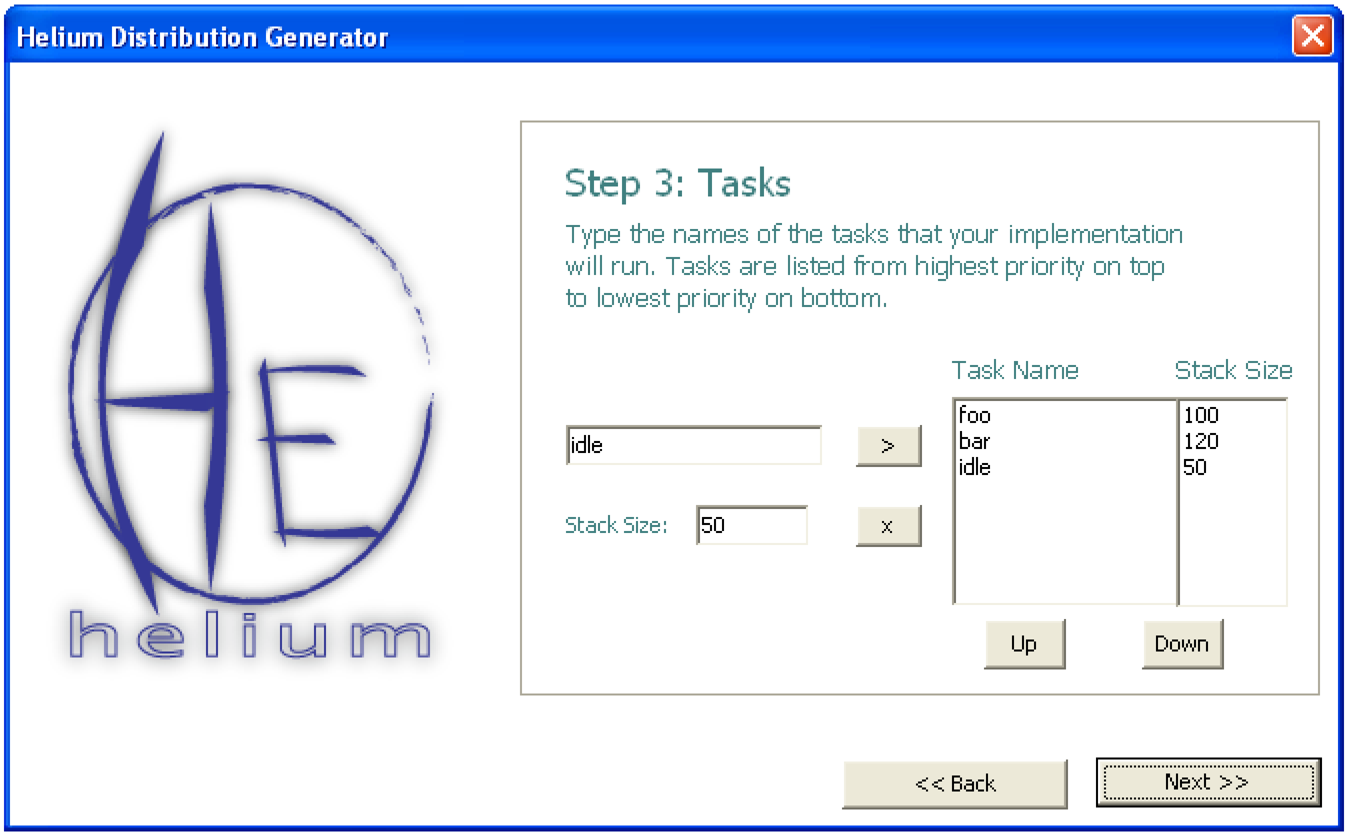
void MCU_init() ;

[✓] Call initialization routine in boot code.

Step three allows the user to set up a task hierarchy for Helium. For each task, the configuration wizard will generate a C file with an empty function body, create a stack for it, and initialize the task on bootup. A task's stack size should be at least 20 bytes on an HC(S)08 CPU and at least 50 bytes on an AVR. Those sizes are used by the Helium scheduler. If the task declares any local variables or calls any functions, the stack sizes will need to be increased.

In the figure below, the user has created three tasks: `foo`, `bar`, and an idle task. `foo` and `bar` are tasks that will actually do work. Of course, both will block for some period of time during their execution, possibly waiting for data from a peripheral device or waiting for the other task to complete a calculation. If both tasks are blocked at the same time, the idle task will be scheduled. Without an idle task, Helium would not know what task to schedule when all other tasks are blocked, and the operating system would crash. The idle task just needs to run in an infinite loop, so it can have the minimum size stack for the CPU.

The idle task would not be necessary if the CPU was put to sleep when no task was ready to run. This scheme would save power, but Helium does not currently support it.



Helium Distribution Generator

Step 3: Tasks

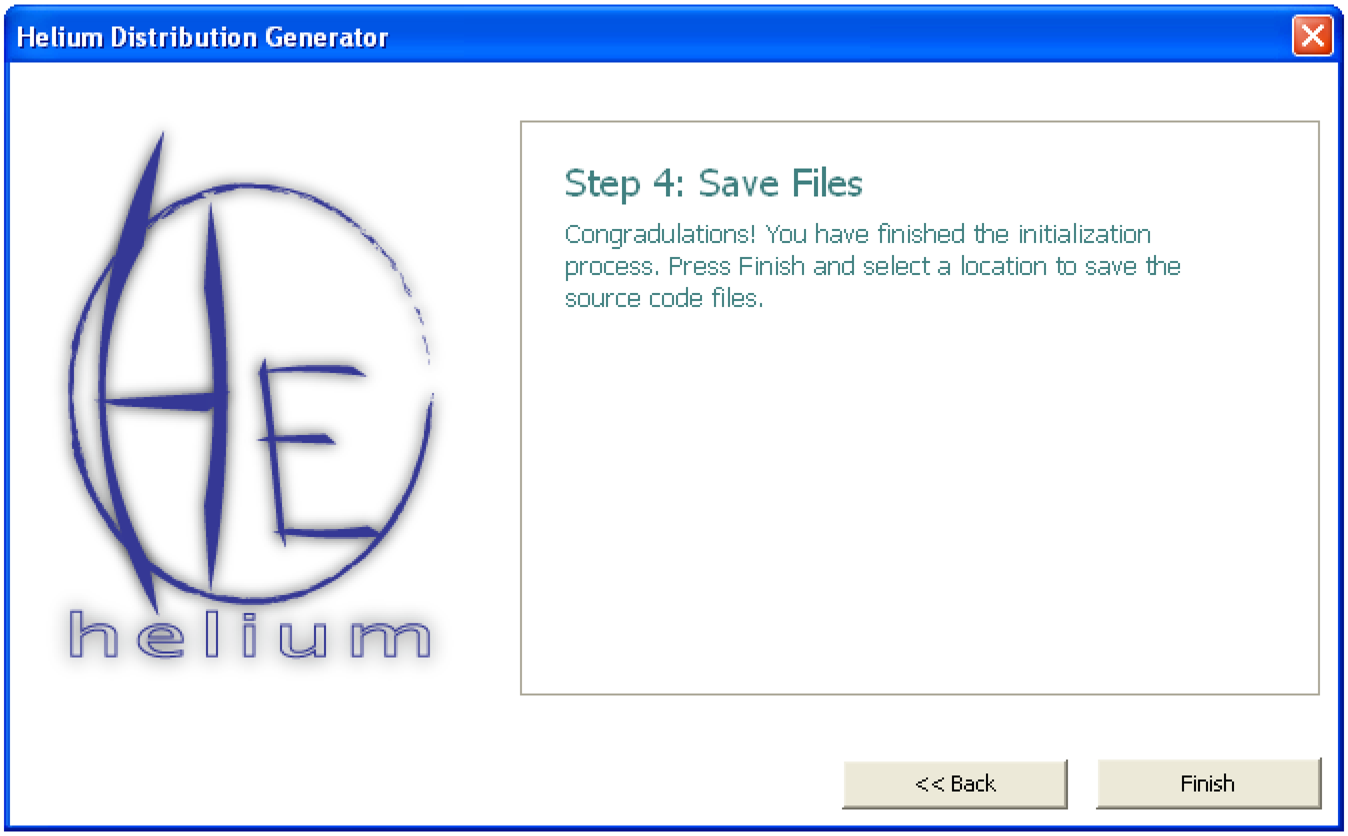
Type the names of the tasks that your implementation will run. Tasks are listed from highest priority on top to lowest priority on bottom.

idle

Stack Size: 50

Task Name	Stack Size
foo	100
bar	120
idle	50

After the task list is populated, the wizard will save the source code. Click (Finish) and select a location for `main.c`. All other source code files will be placed in the same directory as `main.c`.



Step 4: Save Files

Congradulations! You have finished the initialization process. Press Finish and select a location to save the source code files.

After the source files are saved, they can be imported into a project in the IDE for the appropriate microcontroller. If an MCU initialization routine was specified in Step 2, that routine will also have to be imported or written from scratch. The Helium distribution should then compile with no errors.