

Re: header files including other files

Source: <http://coding.derkeiler.com/Archive/General/comp.arch.embedded/2007-09/msg00228.html>

- *From:* Tim Wescott <tim@xxxxxxxxxxxxxxxxxxxx>
 - *Date:* Thu, 06 Sep 2007 23:10:29 -0500
-

On Thu, 06 Sep 2007 22:37:34 +0000, Peter Harrison wrote:

Sorry if this is out of place here. Comp.lang.C would probably be better but I left my flameproof underwear at work.

Really I guess I am asking advice about source code organisation.

I have a C language project – a small robot. I want to use the various types defined in `stdint.h` such as `uint8_t`, `uint16_t` and so on as I expect the code to get ported to other processors.

Now, in any particular module, the module header file declares exported functions and variables and so needs to know the types. Is it considered acceptable to include the `stdint.h` header in the module header or is it better to include it in the source code. Now I have written it out loud, I suppose the latter is best as the header file never gets compiled, just included with other compilation units.

In a similar vein, the hardware definitions for the location of the buttons, LEDs, motor drivers etc all live in a single header file called `hardware.h`. In here, I have included the processor definitions header as the actual names of the ports and pins are closely dependent upon the actual processor used. This is the only place in the project where the processor header is included. Is it better to do this?

I would be grateful for anyone who can point me at a good resource recommending how best to organize modules and their headers. From a lot of browsing about, everyone seems to just do their own thing but there must be good and bad practice out there to learn from.

Pete

Unless you are doing something *_very_ weird* you should *_always_ include* predecessor header files in successor header files. Your header files should be designed so that you can plop one into an empty "C" file and compile it with no errors. Furthermore, they should be designed so that you can put them into a "C" file in any order and have a successful,

Re: header files including other files

predictable compile.

So the other thing you should do is guard the contents of your header file. So, for instance, at the start of header.h you should have

```
////////////////////////////////////  
// file: header.h  
// -- etc --  
//  
  
#ifndef HEADER_H  
#define HEADER_H  
  
// (put your header stuff here)  
  
#endif // HEADER_H
```

If you don't do this, then the second time the compiler encounters header.h (because you've included it everywhere) all hell will break loose.

--
Tim Wescott
Control systems and communications consulting
<http://www.wescottdesign.com>

Need to learn how to apply control theory in your embedded system?
"Applied Control Theory for Embedded Systems" by Tim Wescott
Elsevier/Newnes, <http://www.wescottdesign.com/actfes/actfes.html>

.