Arduino

How to call C functions from Arduino sketch?

Asked 7 years, 10 months ago Active 11 months ago Viewed 16k times

11

I would like to know if there is a way to call functions that are contained within C files using an Arduino sketch?

My C file declares and defines a function. To save putting the messy function definition into my Arduino sketch, I'd like to call the function straight from the sketch.

Is there a standard way to do this using Arduino and C? Here is the sketch:

```
#include "crc16.h";

void setup(){

}

void loop(){

  CalculateCRC16("<09M", 4);

}
```



```
uint16_t crctable[256] =
{
```

```
{
    0x0000, 0x1189,.....



uint16_t // Returns Calculated CRC value
CalculateCRC16( // Call example CalculateCRC16("<09M", 4);
    const void *c_ptr, // Pointer to byte array to perform CRC on
    size_t len)        // Number of bytes to CRC
{

    uint16_t crc = 0xFFFF // Seed for CRC calculation
    const uint8_t *c = c_ptr;

    while (len--)
        crc = (crc << 8) ^ crctable[((crc >> 8) ^ *c++)];

    return crc;
}
```

arduino-uno software c

Share Improve this question

Follow

edited Mar 27 '14 at 14:14

asked Mar 27 '14 at 12:00

user_name

317 1 3 11

Is there a reason why your file has to use C instead of C++? – Peter Bloomfield Mar 27 '14 at 12:03

Actually, yes. When I try to compile the file using C++, there are errors, but it is error free in C. The error is caused by the lines: `const void *c_ptr` and `const uint8_t *c = c_ptr;` . The error message mentions an invalid conversion between types. – user_name Mar 27 '14 at 12:07

4 Could you please post the 2 code files (or a simplified minimal version of them) that produce the error, and copy&paste the error message in full? – drodri Mar 27 '14 at 12:34

The error messages aren't so pretty: `In function uint16_t CalculateCRC16(uint16_t, const void*, size_t)': 46 invalid conversion from const void*' to const uint8_t*' In function int main()': 57 system' undeclared (first use this function) (Each undeclared identifier is reported only once for each function it appears in.)` – user_name Mar 27 '14 at 14:18

When a function was called in the c file, how I notice this in the original Arduino file (ino)? Can I create any callback function form the c file to the ino file? – tk97tk Feb 21 '21 at 21:02

3 Answers

Active | Oldest | Votes

```
void loop(){
  CalculateCRC16("<09M", 4);
}
```

And the crc16.h file could be (some minor fixes, the #pragma once, a cast):

```
#pragma once

#include <stdio.h>
#include <stdint.h>

uint16_t crctable[2] ={ 0x0000, 0x1189};

uint16_t CalculateCRC16( // Call example CalculateCRC16("<09M", 4);
    const void *c_ptr, // Pointer to byte array to perform CRC on
    size_t len)        // Number of bytes to CRC
{
    uint16_t crc = 0xFFFF; // Seed for CRC calculation
    const uint8_t *c = (const uint8_t *)c_ptr;

    while (len--)
        crc = (crc << 8) ^ crctable[((crc >> 8) ^ *c++)];

    return crc;
}
```

Share Improve this answer Follow

answered Mar 27 '14 at 14:33

drodri
1,344 10 12

Thanks, it works just fine now. Could you please explain the need of the pragma? – user_name Mar 27 '14 at 14:44

1 Sure, it is a good practice, though it is not needed in your example. It avoids the same header file to be included twice in a compilation file. Imagine a.cpp->(b.h and c.h) and b.h->c.h. That will duplicate the contents of c.h while compiling a.cpp. The #pragma once avoid this. Also guard directives #ifndef _MY_FILE_H_INCLUDED #define _MY_FILE_H_INCLUDED are common for this. Note, however, that as Peter R. Bloomfield points out, it might be better to put the implementation of CalculateCRC16 in a cpp file, and leave just the declaration in the header file. – drodri Mar 27 '14 at 15:45

Ok, I can see that becoming an issue when the code gets more and more complicated. Thanks for the advice. – user_name Mar 27 '14 at 15:52

4

Your CRC function can easily be converted to C++ so that it can go into a *.cpp file. All you need to do is use an explicit cast when you initialise your `c` pointer. Here's the 'proper' C++ way to do it:

The problem is basically that C can be a little more permissive about letting you convert pointers implicitly between types. To do it in C++, you need to tell the compiler explicitly that the conversion is intentional.

Share Improve this answer Follow

answered Mar 27 '14 at 14:31

Peter Bloomfield
10.8k 9 44 87

1

Yes, just copy its declaration line in your sketch:

```
extern "C" {
    void myfunction(int arg);
}
```

Share Improve this answer Follow

answered Mar 27 '14 at 13:30

jfpoilpret
8,759 6 31 53
