



## Re: Microcontroller Project

I looked into various compilers when I did a major (~25kloc) 8051 project a decade ago. The only one that didn't mask the limitations (and even advantages) of the 8051 was PL/M51.

I'd then suggest you take a look at some more modern compilers perhaps. Countless people use C on the 8051, I'm sure it's not that bad.

Is assembler any different? (especially for a beginner?)

Yes, because of the need to be keenly aware of the processor hardware limitations. The 8051 isn't an ordinary processor. It only has 128 bytes of RAM (the variant I used had 256) and the stack's gotta fit in there too. When the stack overwrites the registers (memory mapped) all sorts of fun things happen.

If the stack overflow thing is your only gripe then that's not much of a problem.

You'll get dynamic stack allocation errors on most micros, it's not that hard to avoid unless your program is doing something very strange. There are great many applications where the stack usage is very predictive and guaranteed. Typical rule of thumb is to ensure your stack space is twice as big as the largest you'll anticipate using and find during testing. If you use static allocation for everything (and you should be on a micro really) then you'll usually know exactly how much RAM you have left.

You can get stack overflow errors on the PIC too, and I've written plenty of C programs on that in less RAM than 128bytes.

All things considered a high level language will get a beginner up and running much quicker than assembler. Take a look at the popularity of the PIC-AXE, there is a good reason why it's popular.

An interpreter is a little different case. Were that the choice here, I wouldn't have any issue. Assembler is a PITA, but the 8051 has all that in spades. You really have to understand the memory map (IIRC, seven different maps some interrelated) to use it at all. A compiler will try to hide much of this.

I decided to try and implement the Hangman game itself. I have a rough idea as to what I must do, though I am confused as to

## Re: Microcontroller Project

how to go about  
it. Could you tell some good  
resources for 8051  
programming in C on the  
net?

If you are programming in C then your  
actual program will have little to  
do with the 8051.

Again, I disagree. Normally one would assume this, but the  
8051 is a  
"special" (as in; designed by a "short-bus" architect) case.

So what are you saying? That C compilers for the 8051 are no good?  
I find this very hard to believe.

I wouldn't advise a novice to use C on an 8051, no. If you're  
\*really\* good at C it can be done but given the strange  
architecture I don't see what it'll gain over assembler. I don't  
see any real code being portable so why bother?

There are a lot of 8051 C compilers around and I'm sure a lot of people  
use them without issue.

So you've been bitten on the stack overflow issue, but I would not use  
that an excuse to stay clear of C.

Many people have said that C on a low end PIC (another so-called  
"crippled" architecture) is useless, but excellent ANSI C compilers  
are available for them, and the low end ones too.

All sorts of things exist, many for no good reason. No, I don't  
see any advantage to C on an 8051. I can't speak to the PIC, but  
my guess is it's much the same story.

Well I can speak for the PIC, it has a notoriously horrible  
architecture that is very "C unfriendly" yet the modern C compilers  
like the HiTech C do a fantastic job.  
I am sure that thousands happily use C on the 8051 without problem.  
There is no real reason to avoid C on any micro when good compilers  
exists.

## Re: Microcontroller Project

You could write the C program on a PC first, and then port to your 8051. Instead of writing to the screen you would turn on a LED on a port of the 8051, so only a small part of your program should change between a PC version and an 8051 version.

...and watch the stack explode. Good idea!

Once again, what are you saying exactly? Are you saying that no C compiler for the 8051 can anticipate a stack overflow and give you a warning?

No, because stack overflows are run-time events. How is a compiler going to anticipate what the program is going to do at run-time.

It's called static allocation and knowing what your program does. In most programs it's not hard to know the maximum number of stack calls you can expect, and good C compilers can detect this and warn you. For instance, if you have 10 nested function calls and your stack can only support say 8 (after the static variable allocation), the compiler will warn you.

If you are using dynamic allocation for stuff then you are asking for trouble on any micro.

The advantage of using a high level language like C is that you shouldn't have to care what micro platform it is used on.

A naive statement. "shouldn't?" There's a comforting statement. Very few people can write truly platform independent C. ...I'm certainly not one, and my bet is that given your statements here that you aren't either.

Ok, I should have said "shouldn't have to care too much" especially for the OP's case of a simple hangman game. No C program is ever truly platform independent, but you can go a long way to making it so.

## Re: Microcontroller Project

They can be made so, or at least C programmers on the comp. groups think they do it. ;-) ...but it's usually not worth the bother.

Again, the 8051 is a strange beast. Much of its utility comes from this strangeness (memory mapped registers, bit addressable memory, strange and wonderful hardware, etc.). Hiding it isn't a good idea and may invite disaster (e.g. stacks creaming registers).

Check out the PIC some day, some said it was impossible to write an ANSI C compiler for it, yet many have done it now.

In many cases only minimal changes are required to the code that accesses the hardware, and depending on the program and application that can be a tiny percentage of the whole code indeed.

You'd loose your bet BTW :->

You just said it can't be done a few lines up; "No C program is ever truly platform independant"

It ain't that hard if you try, really.

Which is it? ;-)

If you need help with the C code then the 8051 is the least of your problems.

Granted, but workign C isn't likely to help him either.

Maybe not, he'd probably have oodles of problems with the programmer settings alone :->

There's that too. I wonder what he's going to use as a development system. ...edit->compile/link->program eprom->plug->repeat is going to be a bitch.

He'll be playing with that for weeks!

Dave :)

Re: Microcontroller Project