

Re: Microcontroller Project

Source: <http://sci.tech--archive.net/Archive/sci.electronics.design/2006-03/msg03892.html>

- *From:* Keith <krw@xxxxxxxxxxx>
 - *Date:* Mon, 27 Mar 2006 12:46:34 -0500
-

In article <1143447041.276029.172680@xx>, altzone@xxxxxxxx says...

Keith wrote:

On Sun, 26 Mar 2006 17:28:03 -0800, David L. Jones wrote:

akshaychander@xxxxxxxx wrote:

Sorry for the late reply.

Yes, I have to do the project in 8051. I have studied the theory of 8051, but have very little practical experience.

As far as languages are concerned, there is no restriction. I am planning on using C.

In that case a good C compiler will take care of most of the low level 8051 stuff for you.

I disagree! A C compiler will just mask the nonsense underneath. It *will* come back to bite you (think "stack").

Ok, I'm not an 8051 guy, so I'll take your word there is a potential issue here.

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.

Re: Microcontroller Project

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.

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 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?

Re: Microcontroller Project

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.

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.

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.

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,

Re: Microcontroller Project

strange and wonderful hardware, etc.). Hiding it isn't a good idea and may invite disaster (e.g. stacks creaming registers).

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.

On the other hand, if your subject is the 8051 micro and it's architecture then you won't learn much by doing your program in C. I am surprised your teacher is not forcing you to use assembler language.

It's not clear what the issue is here. Perhaps it's to show them what they *don't* know? ;-)

A smart lecturer indeed then!

Re: Microcontroller Project

;~)

--

Keith

.