Re: [OT] Re: writing a dailer in c for a 8051 based system

Source: http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2004−05/3247.html

---

From: Dan Pop (*Dan.Pop_at_cern.ch*)
Date: 05/27/04

```
Date: 27 May 2004 16:46:46 GMT
```

In <c92iql$gr8$1@nntp.webmaster.com> "David Schwartz" <davids@webmaster.com> writes:

>"Dan Pop" <Dan.Pop@cern.ch> wrote in message
>news:c926pr$k03$8@sunnews.cern.ch...
>> In <pan.2004.05.26.09.52.12.812641@sig.now> August Derleth <see@sig.now>
>> writes:
>
>>>[1]A microchip designed to be used in an environment other than a PC,
>>>server, router, or other obviously−computer environment.
>
>> Many older hard disks found in PC's used an 8051 chip, so your definition
>> is rather shaky. Furthermore, Intel had an 8052 chip with an embedded
^^^
>> BASIC interpreter, that could be used as the core of a small computer.
^^
>
> He's saying it's "*designed* to be used in an environment other than a
>PC".

So are plenty of high performance RISC processors, designed to be used in workstations and servers...

>The 8051 was designed to be embedded in a wide variety of devices, and
>not primarily as the central processor of a general−purpose computer.

See the counterexample mentioned above...

>>>They typically
>>>have a place to hardcode instructions (that is, onboard ROM space) and
>>>might support odd word sizes to interface with specialty hardware.
>
>> This is not the case of the general purpose ones, like the 8051, which
>> are best described as single chip computers, containing a CPU, ROM, RAM
>> and various external interfaces on a single chip.
>
> Huh? He said it had unboard ROM space and you said it contained "ROM ...

>on a single chip". So how is it not the case?

I was commenting the whole of his statement, not the on−board ROM bit.
See the rest of the paragraph for full enlightenment.

>> The word sizes are
>> the vanilla ones, so that adding external ROM or RAM can be done using
>> cheap, off the shelf, components (no point in having a cheap
>> microcontroller if it needs expensive external chips).
>
> You may not be familiar with the 8051, but it's addressing modes were
>quite unusual.

Having programmed in 8051 assembly, I am reasonably familiar with it and
its addressing modes. I wouldn't say that a word size of 8 bits is
particularly odd...

>(If you only programmed it in C, you might not have realized
>just how odd it is.) As an example, how many bits do you think you need for
>a general purpose pointer in an 8051? (Keep in mind it has code and data in
>distinct address spaces.) And what about the external MOV instructions that
>take an 8−bit address? Do you know where the other 8 bits of the address
>comes from? (It's truly bizarre.)

I'm failing to see the connection between these questions and the size of
the 8051 word. In your humble opinion, why is it called an 8−bit
microcontroller?

>>>Low
>>>power consumption and extremely low cost−per−chip often counts for more
>>>than clock speed.
>
>> Low power consumption is often not an issue (no point in using a < 1 mW
>> controller in a microwave oven)
>
> His claim was "often". And he didn't say it was the most important
>thing, he said it counted more than clock speed. You aren't seriously going
>to argue that clock speed is important in a microwave oven.

If it's not important, then try to control one using a 1 Hz clock...
But my point was that low power is not that often an overriding concern,
while low cost per chip *is*.

If you'd read my sentences and paragraphs in their entirety instead
of interrupting the process at arbitrary points, you might even be
able to see my points by yourself...

> So he said A is often more important than B and you came up with a case
>where neither is particularly important. How do you think this refutes what
>he said?
>

>> but low cost per chip is an overriding
>> concern, otherwise the designer would opt for a traditional microprocessor
>> + ROM + RAM + I/O intefaces approach, which is often more comfortable to
>> program on (the typical CPU in a microcontroller is less capable than an
>> usual microprocessor, the program and the data often reside in different
>> address spaces, and, for machine code compactness reasons, the program
>> memory is often paged).
>
> Well that's the first thing you've said that's correct.

Everything I said in my previous post was factually correct. Learn to
read and you may eventually realise it.

Dan

--
Dan Pop
DESY Zeuthen, RZ group
Email: Dan.Pop@ifh.de