

Re: Learning to use PICS

Source: <http://sci.tech-archive.net/Archive/sci.electronics.design/2004-11/7200.html>

From: mike (spamme0_at_netscape.net)

Date: 11/29/04

Date: Mon, 29 Nov 2004 13:46:41 -0800

David Harper wrote:

- > All,
- > I appreciate everyone's suggestions and help so far. Right now I've
- > started diving into understanding the architecture and memory of a
- > typical PIC (16C84), which I figure is the best place to start. After
- > that, I figure the programming will be a lot easier to understand.
- >
- > The program memory I understand, no problem (like the BS2, only it
- > seems only instructions can be written at these locations for PICS,
- > and only during programming).
- >
- > However, with the data memory allocation, I'm having some difficulty
- > based on some of the online datasheets:
- > <http://ww1.microchip.com/downloads/en/DeviceDoc/30445c.pdf>
- >
- > and beginner guides (from piclist.com):
- > <http://www.piclist.com/techref/microchip/intro/pic.htm>
- >
- > From what I've read, there are 2 banks each divided into 128
- > registers. The first 12 registers are SPR, which more or less define
- > the chip's current state. The next 32 registers are GPR (like RAM?).
- > What are the next 88? It's defined as "unimplemented data" according
- > to Fig 4-2 in the 16C84 datasheet.
- >
- > Secondly, figure 4-7 (pg 18) shows 4 banks, not just two... just how
- > many banks are there for this chip? Can it be more than 2 banks for
- > different PICS, which is why they're showing it as 'off limits', so to
- > speak?
- >
- > Lastly, back in figure 4-2, it states the 36 GPR in bank 1 are mapped
- > to bank 0. Does this mean they're connected, and if a GPR in bank X
- > changes, then the same GPR in the other bank will change also? If so,
- > are any of the SPR connected in this fashion?
- >
- > Thanks for the patience if you've made it this far, and I really
- > appreciate the help!
- >
- > Dave

>

> *Secondly*

If your objective is to understand the architecture of a 16F84, you're on the right track to understanding an obsolete and arcane part. There's only one reason to use paged/banked memory...Price! And that advantage has long since been nullified by technological advances.

While it's possible to understand the inner workings of a PIC16 part, the details of actually making it work in practice will drive you nuts. All it takes is one mistake in setting the page registers and your program won't work. And when you edit it, things move around and things that used to work are now broke.

Now, any programmer who does this all day can manage it. Those of us who drag out a PIC once a year to do a quick project tend to get buried by those details. I want to hack out some code and have it work. I don't want to go searching for errors caused by my forgetting some processor quirk.

One option is to let someone else manage all those details. A compiler should be smart enough to keep you out of trouble. I use PicBasic, but there should be lots of others.

Second option is to use a processor that doesn't page memory. I think the PIC18 series fixes up most of this mess, but I haven't tried 'em yet. Think I might go with AVR if I didn't already have investment in tools for PIC.

All boils down to what you're trying to accomplish. Can't think of any reason to want to understand an obsolete part. In sample quantities, all PICs are the same price...free.
mike

--

Return address is VALID.

500MHz Tek DSOscilloscope TDS540 \$2200

<http://nm7u.tripod.com/homepage/te.html>

Wanted, 12.1" LCD for Gateway Solo 5300. Samsung LT121SU-121

Bunch of stuff For Sale and Wanted at the link below.

<http://www.geocities.com/SiliconValley/Monitor/4710/>