

# Re: C3088 CMOS Imaging Sensor Questions

---

*Source:* <http://coding.derkeiler.com/Archive/General/comp.arch.embedded/2006-07/msg00503.html>

---

- *From:* [paul\\$@pcserviceselectronics.co.uk](mailto:paul$@pcserviceselectronics.co.uk) (Paul Carpenter)
  - *Date:* Fri, 14 Jul 2006 14:32:00 +0100 (BST)
- 

On 14 Jul, in article

<1152878792.710362.210770@xx>

weg22@xxxxxxxxxxx wrote:

Assuming the PIC can keep up with normally 8.68MHz \* 2 bytes transfers during an image capture, what else do you want it to do? Because it will be almost permanently locked in that interrupt routine doing NOTHING else.

I'd like to (a) take spatial (x & y), and temporal derivatives and (b) incorporate those derivatives into another equation.

Which require LOTS of computation time and memory to process. Especially as spatial (x & y) in image processing can mean a lot of things depending on how many points to be found and how those points are to be recognised.

Are you sure your PIC has enough RAM for your program variables and at least a buffer 356 x 292 x 2 bytes EACH! Your PIC does have 256kB of RAM at least  
I take it. Normally you need at least two buffers for image processing on whole images so double that memory requirement to 0.5MB!

Can you please explain why you need two buffers when doing image processing (I'm new at this)?

To maintain a reasonable speed on frame update or just for processing. Often applications have dedicated hardware to grab one frame, whilst working on a copy of a previous frame. Many algorithms need two buffers for part or result of process, having an input buffer and an output (or temporary buffer), the input buffer may well be used more than once depending on output buffer contents.

## Re: C3088 CMOS Imaging Sensor Questions

My PIC has 3.94 KB of RAM and I probably won't be processing the entire image...more like a 100 x 292 image.

The basic problem is you will NOT be able to use the PIC directly to probably get more than two pixels per video line. The video rate is too fast for the PIC and the many instructions required to do the transfer to keep up. Even in machine code!

Some people use algorithms that ASSUME that if they take several pictures and grab at different times they can build up an image over many frames. This method can SOMETIMES work to a limited extent as lighting variations let alone movement in the image over time greatly reduces the accuracy and resolution of data.

You will NOT be able to process on the fly.

I didn't think I'd be able to get an algorithm working at 30 fps, but I was hoping for at least 5. Do you still not think this is feasible?

You will be lucky to get ONE per second. The PIC is the wrong beast to do this, image processing digitally needs lots of dataspace and processors that can do 8/16/32 bit operations for quite a few of the operations. Multibyte arithmetic will not cut it for any form of control algorithm as the overhead of doing one 16bit add many thousands of times slows the process to almost glacial speed.

Why a PIC?

I'm use to working with them...besides is the AVR that much different than the PIC I'm working with (PIC18F8722)?

Having larger memory areas and other attributes for the processing and I believe things like DMA make it a lot easier. Video is not like serial data there is a lot of it, in short spaces of time and repeatedly. Miss one pixel and everything afterwards is meaning less.

I have done some highly compute intensive applications on H8 which was grabbing one frame then doing full frame processing, with speeds by lookup tables to produce enhancement maps, that involved all sorts of complicated maths. This device had PLD controlled frame grab (monochrome) into dual ported RAM, and running on 20MHz clock took about 2 seconds to run! H8 was using its 16 and 32bit integer operations and avoiding floats and other nasties. In that application the 2 second wait was faster than what the operator was doing at the time.

Re: C3088 CMOS Imaging Sensor Questions

Finding spatial information and object recognition to find the spatial points is a similar amount of computations if not more on a worse architecture, is asking for trouble.

Work out what image capture size and update rate you need first.

Work out what algorithms you need to run and how much memory and time they are likely to use, then when they need to complete by.

When you have worked all that out, THEN and ONLY THEN think about how you are going to capture it, into what memory, THEN see what is the best micro match to perform the function.

Great suggestions...thanks!

You will need a high end 16bit or more probably 32bit processor like ARM with fast memory access to get anywhere near your required update rate of 5 fps. Even then a dedicated piece of hardware to grab the image into a dual-ported buffer will speed things up immensely.

—  
Paul Carpenter | paul@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx  
<<http://www.pcserviceselectronics.co.uk/>> PC Services  
<<http://www.gnuh8.org.uk/>> GNU H8 & mailing list info  
<<http://www.badweb.org.uk/>> For those web sites you hate