PIC n' Mix

Mike Hibbett

Our periodic column for PIC programming enlightenment

Introducing The Propeller Processor

IN February's issue, Robert Penfold gave an interesting introduction to the Propeller processor demo board from Parallax, and starting this month we thought we would take an in-depth look at the processor, and build a development circuit of our own.

We've been searching for a number of months for a solution to the problem of generating colour video on a small processor, and it was quite by coincidence that Robert's article should appear just days after we discovered the Propeller chip and its on-chip hardware support for generating colour video. The demo board is a nice piece of engineering, with some interesting features, but as Robert pointed out, the number of free I/O pins is limited.

What is it?

So, what exactly is the Propeller Chip? Conceived in the late 1990s by engineers at Parallax Inc, it came to the market in 2006. Frustrated by what they saw as inadequacies in existing microcontroller products, it was originally intended to be a small single-core processor, but through an iterative process of design, test, write software and evaluate, and with falling silicon costs, they ended up at an eight-core controller on a single die.

While we may be becoming used to multiple processing cores on our PCs, courtesy or Intel and AMD, multicore processors are very unusual in the small microcontroller market (in fact we have not seen another – let us know if you have!) This processor is also readily available to the hobbyist, at a very reasonable cost – 5US$ from Digikey, £9 from Farnell. Supplied in both a standard 40-pin DIL and a QFP surface mount format.

Multiple cores

Having eight 'cores' on the chip does not mean that you have eight completely

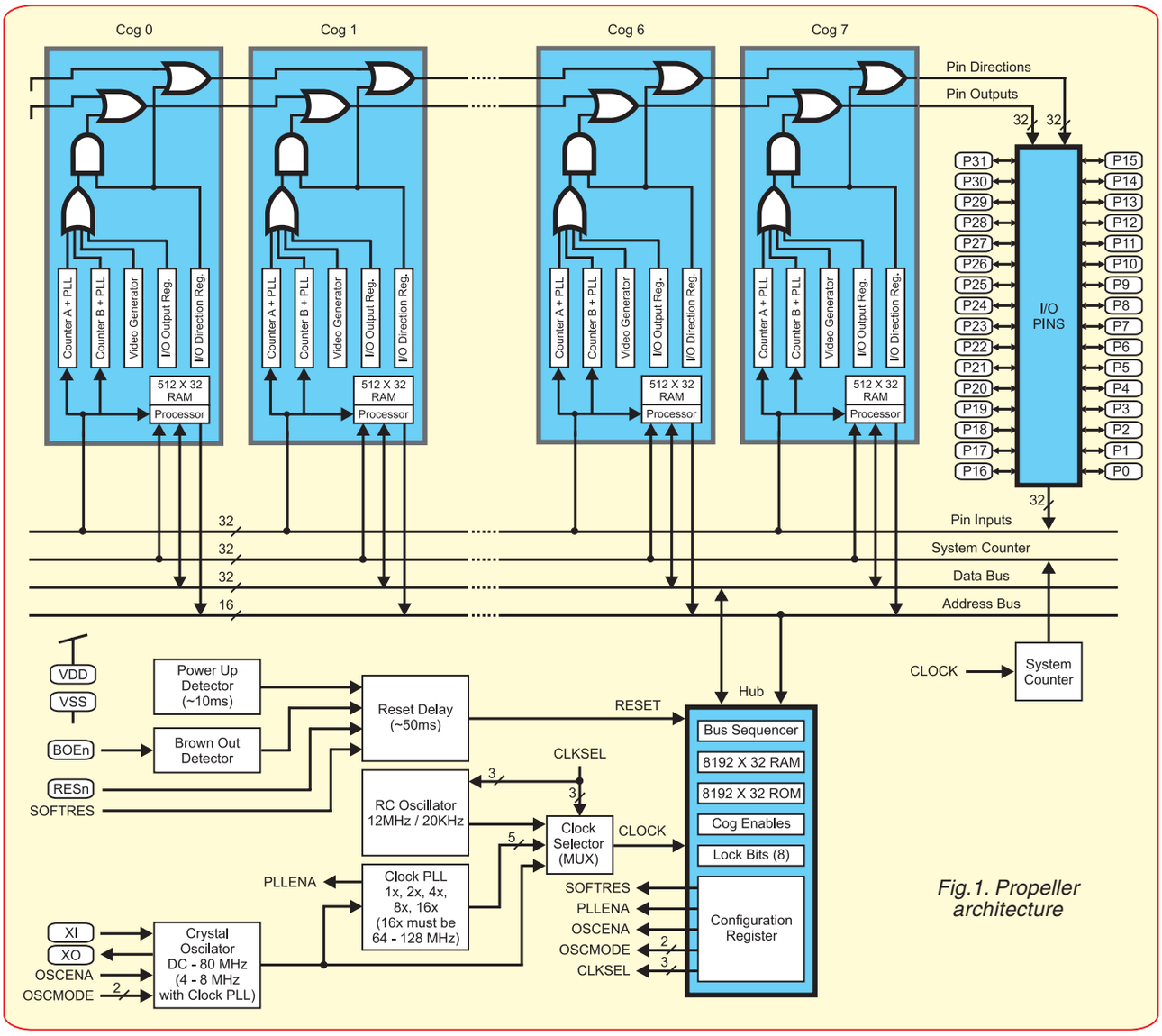

Fig.1. Propeller architecture

independent processors sitting inside a single package. There are eight 32-bit processing units with their own independent program and data area, but sharing access to common peripherals. The setup is shown in Fig.1.

Within each processor is 2KB of RAM that serves as both program and data storage. Each processor also has two counters, video generation hardware and an I/O peripheral to drive the single 32-bit-wide I/O port. As you can see, every processor can drive all 32 I/O pins. It's a flexible arrangement designed so that the individual processors, called 'Cogs', can drive individual pins, or even share control of a single pin if desired.

Interestingly, there are no interrupt capabilities on this device, to which some will give a sigh of relief. The intention is that tasks which would normally require an interrupt, such as an RS232 interface, will be handled by a dedicated Cog. This would poll the receive pin and bit-bash the transmit pin, and write received data into a buffer in common memory, to be picked up and processed by another Cog.

Although 2KB of data and program space sounds small, this is an allocation per Cog – so there is a total of 16KB for the chip as a whole. In addition, there is a single block of 32KB RAM that is accessible by all Cogs. Access to this memory is coordinated through the 'Hub', a synchronisation mechanism that ensures only one Cog can be reading or writing to it at any one time. While all Cogs run in parallel (at up to 20 million instructions per second each, for a maximum performance of 160MIPS!) access to the shared memory has to wait until the Hub allocates a processor a time slot. The Hub 'spins round' at a rate of half the system clock speed, and so a Cog can gain access to the shared memory once every 16 clock cycles.

Every Cog is allocated a slot (even if that Cog is not use) and so the timing of access to shared memory is deterministic, allowing the software developer to optimise their software to account for the delay when waiting for the next time slot. The Hub arrangement is shown in Fig.2

In addition to the 32KB block of shared RAM, the Propeller chip is also equipped with a 32KB block of read only memory (ROM.) The ROM holds a number of useful data tables, including a character set for the video generator and mathematical tables to enable fast log, antilog and trigonometric functions, useful when generating graphical display data.

Spin interpreter

Also held within 8KB of the ROM is a custom language interpreter. Called 'Spin'; it's a very efficient language, not unlike BASIC. A ten-lesson tutorial is presented in the freely available development environment, and there is a wide range of Spin modules available for free on the Parallax website. It's a highly tuned language, even the video generation software examples are written in Spin, which makes writing your own applications much simpler – just copy in the relevant Spin modules to your project and build upon them. Programs can be written in assembler too, but with a very efficient interpreted language built in there is seldom need to work at such a low level. We will be covering Spin next month when we look at the video generation capabilities.

A C compiler is being developed which is also freely available. The aim is to provide a speed improvement over the Spin interpreter and enable existing C programs to be (relatively) easy to port over to the Propeller chip. Whether this will be more efficient in terms of code space is yet to be determined; for us, however, Spin looks like the appropriate route to follow for now.

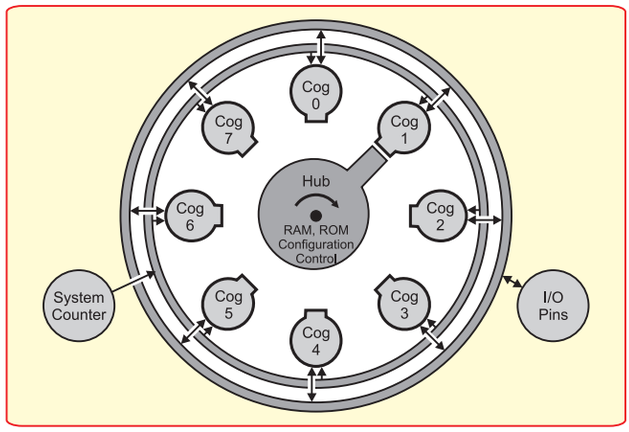

Fig. 2. Hub and Cog interaction

Packages

There is a single version of the Propeller chip at present – the P8X32A, available in three package types: LQFP, a surface mount package with 0.8mm pin spacings, QFN, a surface mount package with no pins at all (almost impossible to hand solder,) and the 40-pin DIP. The DIP package can fit in breadboards or matrix boards, and sockets are cheap, which will make it a favorite with hobbyists. A close inspection of the datasheet suggests that a device with two 32-bit I/O ports may be a future option. And who knows, maybe 16 Cogs? That would be impressive, but is just speculation on our part. The pinout for the DIP package is shown in Fig. 3.

Documentation

Although there isn't a vast amount of information on the Propeller chip available on the web (we PIC enthusiasts have become spoilt over the years,) the available document is clear and well presented. A datasheet, user manual and example demo schematic provide sufficient information to get you started, and there is an online database of free software that can be downloaded and incorporated in your own designs.

A book on the processor has just been released, details of which can be found on the Parallax website. And there is, as you might expect, a very active and friendly user forum.

Development environment

Parallax supply a complete, free, development environment, not unlike MPLAB. It's a lot more straightforward than the Microchip offering, but then this is hardly surprising as they have only a single device to deal with, while MPLAB has hundreds. The Propeller Tool combines editing, compiling and downloading in one simple application. There is no debugging or simulation support, however.

Getting your code into the Propeller is again very different to the more common microcontrollers. There is no non-volatile writable memory on the chip; instead, a built-in bootloader looks to see if a program is being downloaded to it from a serial interface. If not, then it attempts to load an image from an externally attached 32KB or 64KB EEPROM.

The Propeller Tool implements the serial communications interface, and provides a facility to program the attached EEPROM chip through the Propeller. So, from the Propeller tool you can write code and quickly download it to the device. It does mean that any design based on the Propeller must have an EEPROM device too, but such chips are cheap and readily available.

Hardware setup

An external crystal typically in the range of 4MHz to 8MHz is multiplied internally to provide a system clock of up to 80MHz. An internal RC oscillator can also be used, if accurate timings are not required. A standalone system will require a 24LC256 32KB EEPROM for user code storage. The processor runs at between 2.7V and 3.3V and consume only 80mA at full speed, with all Cogs operational.

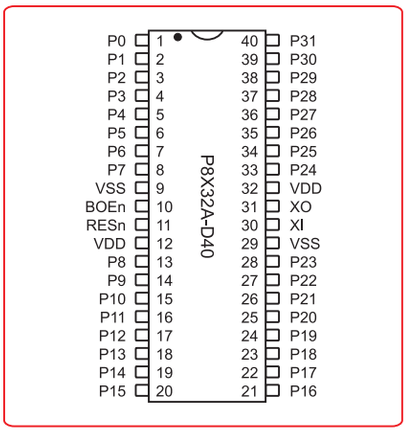

Fig.3. DIP package pinout

Video support

As mentioned earlier, one of the main attractions of this device is its on-chip hardware support for generating video signals. This is a relatively simple circuit that takes care of the most time consuming aspects of bit-bashing composite and SVGA colour signals. This still leaves the processor with a lot to do, but off-loads enough of the processor intensive work to make multiple resolution colour display generation affordable, and it requires very little power consumption.

Next month, we will start with the design of a very simple circuit to enable us to work with the video capabilities of the chip, and if successful, we will then look at a practical application circuit.

References

C Compiler: http://forums.parallax.com/forums/default.aspx?f=25&m=388930

Parallax website: http://www.parallax.com/