



Your support network

Productivity is everything nowadays. How that productivity is defined varies – current favourites include time to market, time to volume or even time to money – but the bottom line is that designers are under pressure to get the job done as quickly as possible.

These pressures have existed for some time in the consumer electronics world, where shelf lives dictate a rapid development cycle for products. But those pressures are now becoming common in the embedded systems sector. So it's no surprise to see companies developing ways to help embedded systems designers maintain their productivity.

A key activity within the embedded systems world is developing board support packages (BSP). These packages have a seemingly simple role in life: to help the designer's embedded software application to communicate with the hardware resources available on the host processor.

Paul Tingey is a systems architect with Wind River. He gave his view of what a BSP is. "The best way to describe a BSP," he believed, "is as an abstraction layer between the operating system and hardware. When you want to run a generic operating system on a specific piece of hardware, you need a layer between the two and that's a BSP."

So what's in a BSP. Tingey noted: "A BSP will include analysis and configuration tools and may also include device drivers over and above those which come with the operating system. What a BSP does is to take you away from the operating system and the hardware. And you need a BSP whenever you're going to run a real time operating system or something like it – Linux, for example."

Working with a real time operating system (rtos) increases the need for a

Board support packages are adapting to cope with Linux, programmable logic and multicore processors. By **Graham Pitcher**.

BSP. "If you're running any kind of rtos on any kind of board, you'll need that layer of abstraction. But the other thing you'll need is specific hardware or architectural support for that board."

Tingey used Wind River's VxWorks rtos as an example. "We have one large source," he noted, "and recompile this source for a specific architecture – ARM, PowerPC and so on. These lumps of code run on that particular processor, but what that code doesn't know is where memory is or what the port

address may be or how wide the flash memory is; and that's what you get with a BSP."

With such a wide range of target processors and operating systems to deal with, it's no surprise to find the BSP world is complex. "There's a huge amount of variation," Tingey observed. "A VxWorks BSP may appear simple, because there are a limited number of files. In fact, there may only be a couple of dozen files and when you look at them, they play distinct roles and are easy to read."

The situation changes, however, when

Elly Walton





"Every sector is asking about multicore. This makes BSPs more complex because of the interaction."

Paul Tingey, **Wind River**

Linux is considered. "Linux BSPs don't exist," said Tingey. "The relevant files are scattered around the Linux tree and it's more complex."

Tingey noted another point to bear in mind. "A lot of BSPs don't need many more drivers than are available in standard VxWorks. For instance, if you choose a Freescale 8260 processor, that will include devices supported through the standard VxWorks source. That's fairly easy to do and you don't have to build any more drivers. However, if you include a device for which there isn't a driver, you'd have to surround that and include it in the BSP because it's something the operating system doesn't know about."

All this is fine if you're using a reasonably popular microcontroller. But fpgas are becoming more popular in embedded systems and this complicates things somewhat because fpgas can con-

tain embedded processors.

Tingey said working with fpgas can become a bit more complex. "Take Xilinx' Virtex-4 range for example. These have a PowerPC405 'floating' in a sea of programmable logic and there's no direct access to the processor. So this is supported by a Xilinx embedded development kit (EDK) that allows you to generate what's essentially a BSP."

Within this, tools ask the designer to provide information about what's needed in the fpga fabric and how it is to be connected. Behind the scenes, a bit stream is built such that the PowerPC can 'talk' to the outside world. "It's an extra level of complexity," Tingey observed. Nevertheless, he said the work can generally be done in half a day and this gives the customer something to move forward with, adding 'tools do a lot of the work'.

Giving the Xilinx perspective is Rich Moler, Xilinx' manager of software IP. He said designing for a Xilinx processor involves a hardware platform assembly flow and an embedded software development flow. "Both are handled within Xilinx Platform Studio (XPS), part of the EDK."

The design typically starts by assembling and configuring the processor and its connected components in XPS. Once the platform is defined, the software parameters can be configured.

"A key feature of XPS is its ability to produce a BSP that is customised, based on the users selection and configuration of processor, peripherals and embedded operating system."



The process, Moler claims, creates a BSP automatically that matches the hardware design, eliminates BSP design bugs by using precertified components and jump starts application software development.

XPS can produce customised VxWorks BSPs for the PowerPC405 in Virtex-II and Virtex-4 devices. The BSP contains all the necessary support software, including boot code, device drivers and VxWorks initialisation.

The only difference between fpga and microcontroller BSPs is placement of driver code. Because fpgas are reprogrammable, device driver configuration can change. So driver files are implemented in C and distributed amongst several source files.

Moving to multicore

Looking to the future, Tingey believes the immediate challenge for BSPs is dealing with multicore processors. "Every sector is asking about multicore because just about every processor is moving that way. This makes BSPs more complex because of the interaction between the cores."

Wind River recently unveiled a multicore solution optimised for Freescale's MPC8641D dual core processor. It uses the ability of the processor to support symmetric and asymmetric multiprocessing. Running a copy of VxWorks on each core, processes running on each core communicate using the Transparent Interprocess Communication protocol (TIPC), with shared memory to coordinate tasks and resources. TIPC is designed for intracluster communication – essentially for any type of distributed multiprocessor system.

But, whilst Linux BSPs aren't that structured today, Tingey says that will change. "Everyone is talking about Linux and there's a huge amount of interest. People are getting better at analysing whether it should be Linux or an rtos. In many cases, they are going with an rtos, but Linux is becoming more viable." In response, Wind River is building a template system that will make it easier to create Linux BSPs. 