

Heterogeneous Processing Meets System Level Needs

Making compute-intensive reconfigurable computing practical today is the concept of heterogeneous processing. Crafting such systems requires a unique blend of silicon, switch fabrics and software.

by Manuel Uhm,
Spectrum Signal Processing

Discussions on reconfigurable computing generally remain within the context of the System-on-Chip (SoC) realm. But while such technology shows great promise for the future, it's still too immature for processing intensive systems. Reconfigurable computing at the system level, on the other hand, has matured quickly over the last couple of years to where it's ready now for deployment in signal processing intensive systems.

Such systems typically use several algorithmic components, each suited for different processing devices, such as field-programmable gate arrays (FPGAs), digital signal processors (DSPs) and general-purpose processors. This approach is known as heterogeneous processing.

System Level Reconfigurable Computing

There are several key reasons why system level reconfigurable computing, or heterogeneous processing, is so important for high-bandwidth, processing-intensive applications. These reasons are driven by efficiency of design and cost effectiveness of the signal processing system. First, consider that many sophisticated real-time applications require several processors to do all the signal processing. In these

instances, the data throughput and processing requirements are too onerous for a single processor to handle.

These applications usually consist of several algorithms with different requirements and data throughputs. As a result, there is no single type of software-programmable processing device that is ideally suited to do all the processing required in such applications. For example, FPGAs are well suited for CDMA chip rate processing due to their parallel processing design, whereas DSPs do not have sufficient clock cycles to perform this task.

On the other hand, DSPs are optimal for symbol rate processing due to their serial processing nature and ability to do complex analysis, while FPGAs could conceivably be used for symbol rate processing, but are more expensive than DSPs and consume significantly more power for the same functionality. Hence, they're inefficient and more costly. All those factors are what drive the necessity for heterogeneous processing platforms, if the processors can be tightly coupled.

The major elements that comprise a heterogeneous processing platform divide into the "three Ss": silicon, switch fabric and software. The silicon part consists of the ability to support multiple configura-

tions of reconfigurable processors, such as FPGAs, DSPs and PowerPCs, as required for the specific application. Switch fabrics and other datapaths provide the communications infrastructure that ties all the heterogeneous processing devices together and ensures data movement throughout the system. And finally, software, specifically a hardware abstraction layer (HAL) or application programming interface (API) is critical. The HAL and API both ease the programming methodology by isolating the programmer from the complexities of the multiprocessor hardware.

Silicon Issues in Heterogeneous Processing

Most sophisticated applications have algorithms that are better suited for some processing devices than others. With that in mind, heterogeneous processing platforms are ideal to support algorithm partitioning in the optimal and most cost effective fashion. As a simple example, a communications gateway based on a heterogeneous processing platform might use four FPGAs for decimation and digital down conversion of high-bandwidth intermediate frequency (IF) data, and four PowerPCs for processing the lower speed baseband data.

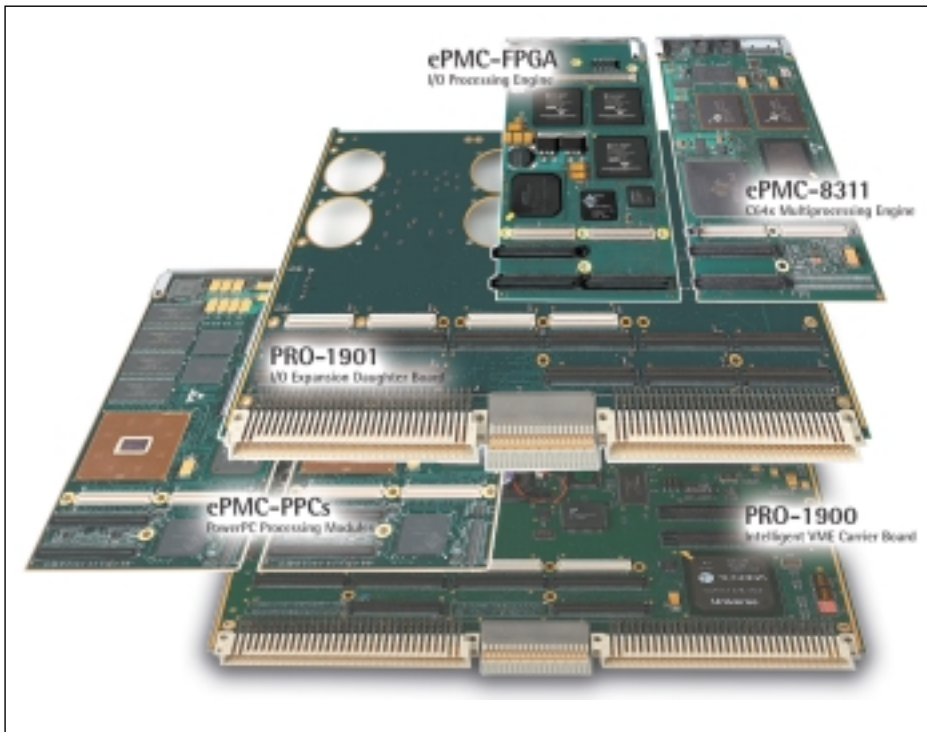


Figure 1 Spectrum's Heterogeneous Processing Platforms Support Modular Mix 'n Match Configurations of FPGAs, DSPs and PowerPCs.

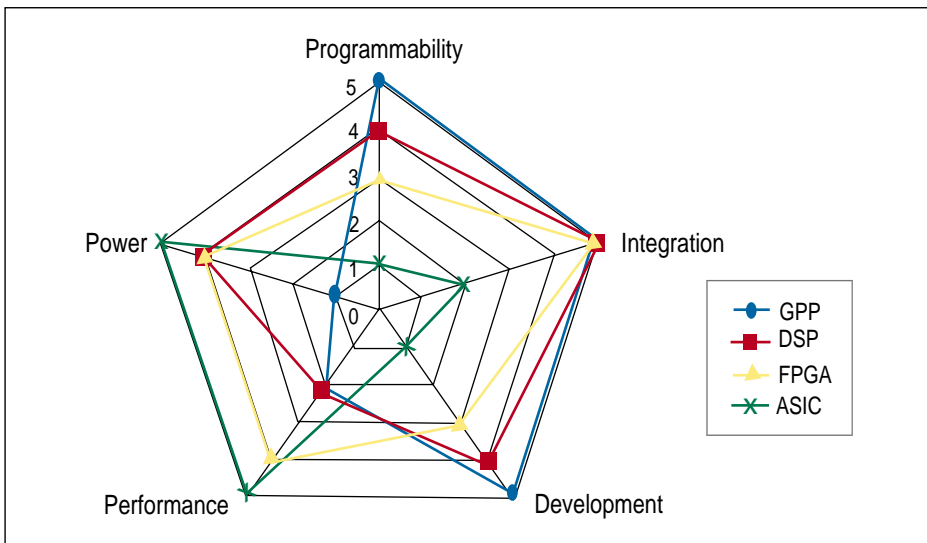


Figure 2 Illustrated here is a comparison of the different device types based on five criteria, where a rating of 5 is better than 1. So for example, ASICs tend to provide the best performance, followed closely by FPGAs, then DSPs and general-purpose processors (GPPs).

On the other hand, a homogeneous processing platform might require twenty or more PowerPCs to distribute the processing, which would be more expensive to produce and consume significantly more power. It's the parallel processing nature of FPGAs that make them a better solution for computationally intensive tasks such as digital down conversion.

A heterogeneous processing platform

should be able to offer the ability to mix and match different types of signal processing devices into the optimal configuration. An example of such a modular approach is illustrated by Spectrum Signal Processing's HCDR-1002 heterogeneous processing platform (Figure 1), which consists of a VME intelligent carrier with an embedded controller, paired with combinations of processing modules that can be

Xilinx FPGA, Motorola G4 PowerPC and/or Texas Instruments DSP-based.

The most common signal processing devices are ASICs, FPGAs, general-purpose processors and DSPs. When choosing between device types for an algorithm, there are five key selection criteria (Figure 2) that designers should consider:

1. **Programmability:** the ability to reconfigure a device to perform the desired functions.
2. **Level of integration:** the ability to integrate several functions into a single device, thus reducing the size and hardware complexity of the signal processing platform.
3. **Development cycle:** the time it takes to develop, implement, and test a function with a specific device.
4. **Performance:** the ability of a device to perform the function within the required time.
5. **Power:** the power utilization of the device when performing the required function.

As a general rule of thumb, ASICs typically offer the best solution for a given function, if they provide an acceptable level of programmability and integration. Meanwhile, FPGAs provide the best programmable solution for high-speed signal processing functions that are highly parallel or involve linear processing. For their part, DSPs provide the best programmable solution for functions that involve complex analysis or decision-making.

Trade-offs in Switch Fabrics

Just as different types of signal processing devices are better suited for discrete algorithm components, different types of switch fabrics or datapaths are usually optimal for feeding data and control information into and out of the various algorithms implemented on each processor to ensure that there are no bottlenecks in the signal processing system. The key issues to consider include bandwidth or data throughput, latency and determinism.

Consider, for example, a hard real-time system like a Software Defined Radio (SDR). SDR systems have four primary data requirements. First, they need a very large, low latency, highly deterministic data pipe to move intermediate frequency (IF) data from A/Ds to FPGAs for IF

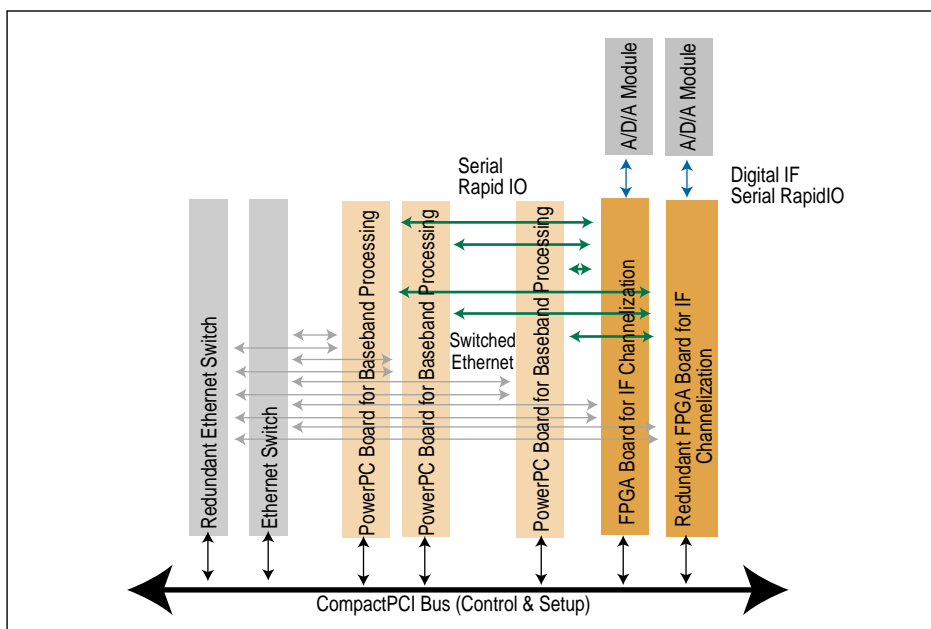


Figure 3 This heterogeneous Software Defined Radio architecture makes use of different datapaths.

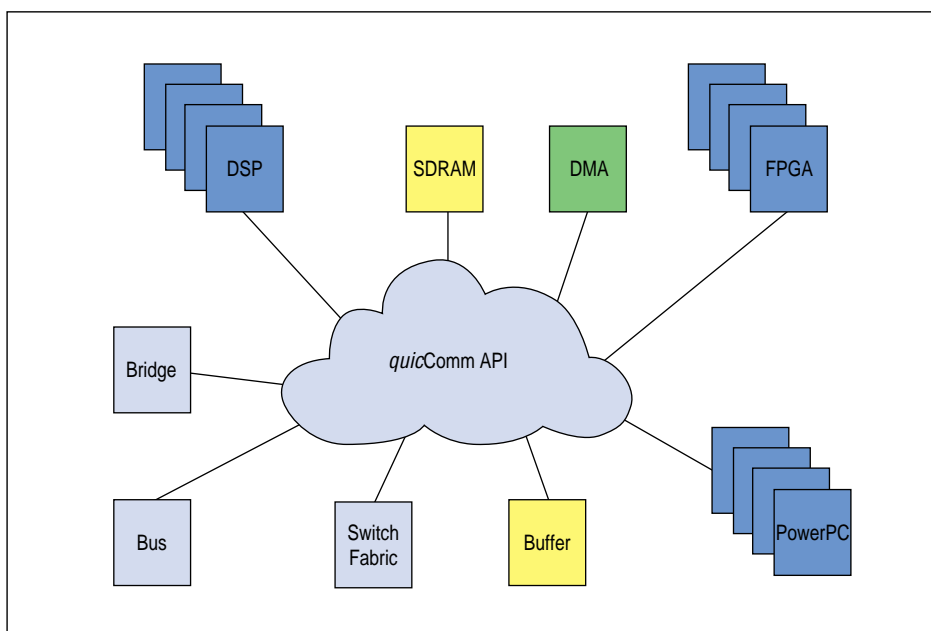


Figure 4 Shown here is a conceptual illustration of an API that binds together all the elements of a heterogeneous processing platform.

channelization. A high-performance switch fabric, such as RapidIO, is optimal for this type of data movement. Second, they require a large, low latency, highly deterministic data pipe to move baseband data from the FPGAs to PowerPCs for baseband processing. A high-performance switch fabric like RapidIO is also optimal for this type of data movement.

The third SDR requirement is a small data pipe that also must be deterministic, to

move payload data to the network. A lower performance switch fabric like Ethernet is a good choice for this. Fourth and finally, control plane data needs to circulate to ensure proper functioning of the system. But that process does not typically need to be deterministic, however. Therefore a parallel bus like the PCI bus, which is inherently non-deterministic, could be used for this purpose. Ethernet or RS-232 would also be suitable.

Such a system could be architected using RapidIO as the large, low latency data pipe, Switched Ethernet as the small data pipe, and the PCI bus running over a CompactPCI backplane as the parallel bus. Shown in Figure 3 is an example along those lines. Such an architecture, with separate datapaths for moving distinct types of data, is critical to the functioning of a heterogeneous processing platform for two primary reasons. First, it's typically undesirable to have low-speed data using up bandwidth in the same pipes moving high-speed data, as that can lead to a bottleneck in the system. Second, it's critical that a non-deterministic control event does not have the capability to interrupt the deterministic data services. It follows then that without these different types of datapaths, a critical bottleneck can occur severely impacting real-time performance.

Software API and HAL are Key

A uniform API is important to enable algorithm partitioning across the various signal processing devices to abstract the programmer from the details of the hardware and operating system. Such an API should enable efficient acquisition and transfer of data through the signal processing system as the data moves through the various datapaths and processors. The same function calls should be used by the programmer to access any resource in the signal processing system, regardless of the type of processor, datapath and operating system.

The API should also provide functionality, such as initializing hardware, downloading algorithms to target processors, controlling data link communications, and generating and responding to interrupts and signals. In addition, to support system level reconfigurable computing, it is important that the API enable dynamic reconfiguration of the target application on a per process basis. All these things significantly ease the programming methodology associated with a complex heterogeneous processing platform. Figure 4 illustrates conceptually how such an API, like Spectrum's quicComm API, can act as the link between the different elements of a heterogeneous processing platform, such as multiple different processors, datapaths and memory.

Another key benefit of a consistent API is code portability, which is critical to

protecting one's software investment. With a unified API, applications can be easily ported between operating systems and hardware, greatly reducing the learning curve and application development time, and extending the life of the software.

Reconfigurable Computing for Today

It's clear that system level reconfigurable computing, or heterogeneous processing, is a rapidly maturing technology that is suitable for processing-intensive applications where single processor designs are not an option. Advances in processing devices such as FPGAs and PowerPCs, as well as high-performance switch fabrics have enabled this trend. A successful heterogeneous processing platform must provide the "three Ss": silicon, switch fabric and software. A modular approach allows the most flexibility in meeting the specific implementation requirements of the application. ■

Spectrum Signal Processing
Burnaby, BC, Canada.
(604) 421-5422.
[www.spectrumsignal.com].