Low-cost real-time depth perception with FPGAs

LOW-COST FPGAS CAN TACKLE COMPUTATIONALLY INTENSIVE DISPARITY MAPPING ALGORITHMS IN REAL TIME AT VIDEO RATES, EXPLAINS **BART BOROSKY**

Real-time depth perception is critical for autonomous robot navigation and other machine vision applications. The algorithms currently used to calculate depth via stereo images, such as disparity maps, are computationally intensive, devouring CPU time or requiring expensive components to operate in real time.

An FPGA coprocessor for depth perception via a stereo camera can free up processor time and reduce or eliminate costs for components such as MPUs, DSPs, lasers and expensive lenses. By providing the robot with a real-time disparity

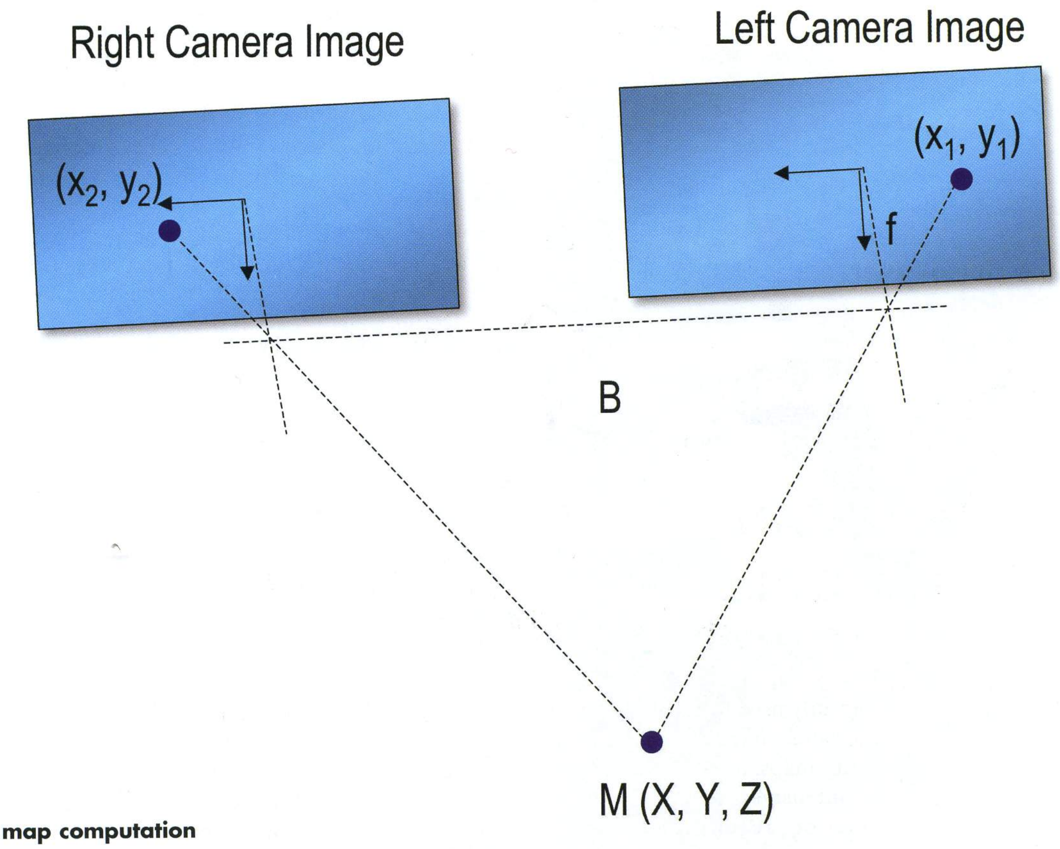

Fig. 1: Disparity map computation

continued from page 29

map of its environment, an FPGA allows the robot's CPU to focus its efforts more on important high-level tasks such as map building and localisation.

Disparity maps

A common technique to add depth perception to a robot is to have two identical cameras mounted horizontally, separated from each other, in parallel. Images from the two cameras are compared using a disparity map algorithm, as shown in Fig. 1.

In simple terms, the disparity is effectively the difference between the right and left images. The larger the disparity between the images, the closer the object is to the cameras.

You can see this by holding a pen horizontally to your face and blinking your right and left eyes. As you move the pen closer to your face, it will "move" more when you blink your eyes. The more movement, the more disparity in the images, and the closer the object is to your eyes. As shown in Fig. 1, the disparity is calculated as the difference in object position between two images: d = x1 - x2. To calculate the real coordinates of point M, captured by the right and left cameras:

$X = Bx1/d$
$Y = By1/d$
$Z = fB/d$

Correspondence problem

Calculating the disparity map relies on corresponding images and matching features from the left image with features from the right image. Calculating the correspondence on a pixel-by-pixel basis is computationally intensive, so other algorithms are used to simplify the problem. One method, called image patch correlation, is to go through the left image in rectangular pixel blocks, called patches, and look for similar patches in the right image. Other popular methods include edge detection and matching. Once the images are aligned and corresponded, disparity calculations can proceed (assuming zero or limited distortion).

Distortion and exposure

If high quality cameras are used, lens distortion can be ignored. To save money, inexpensive cmos or CCD cameras or low-cost wide-angle lenses are often used. These lower quality products may introduce distortion or exposure problems.

Scale and orientation distortion caused by lenses, such as the fish-eye effect where objects nearing the outside of the image are bowed, may be compensated using well known image processing algorithms. One approach to correct fish eye is to remap the pixels in the image using a look-up table (LUT) of coordinates for shifting or other geometric algorithms.

When an environment includes very bright and very dark areas, other algorithms may be required to capture feature detail in the images reliably. For example, determining the optimal exposure settings for different regions and using a composite image of the regions can make up for a lack of consistent background light.

Embedded DSP

The depth perception algorithms described in this article are computationally complex to implement at real time video rates. A system designer must evaluate the tradeoffs among CPU software, asics, ASSPs (such as a DSP) and FPGA, and determine which functions are best served at what price by FPGAs.

Fig. 2 shows a proposed FPGA coprocessor to tackle the computationally intensive distortion, exposure, correspondence and disparity mapping algorithms for depth perception. This frees up valuable CPU time, and allows the DSP to tackle the more serial-intensive tasks.

In general, three circuitry requirements need to be evaluated when selecting FPGAs:

- DSP functionality and performance for coprocessing functions that take advantage of parallelism;
- DDR (double data rate) and LVDS (low voltage differential signalling) support for interfacing to off-chip sdram frame

continued from page 31

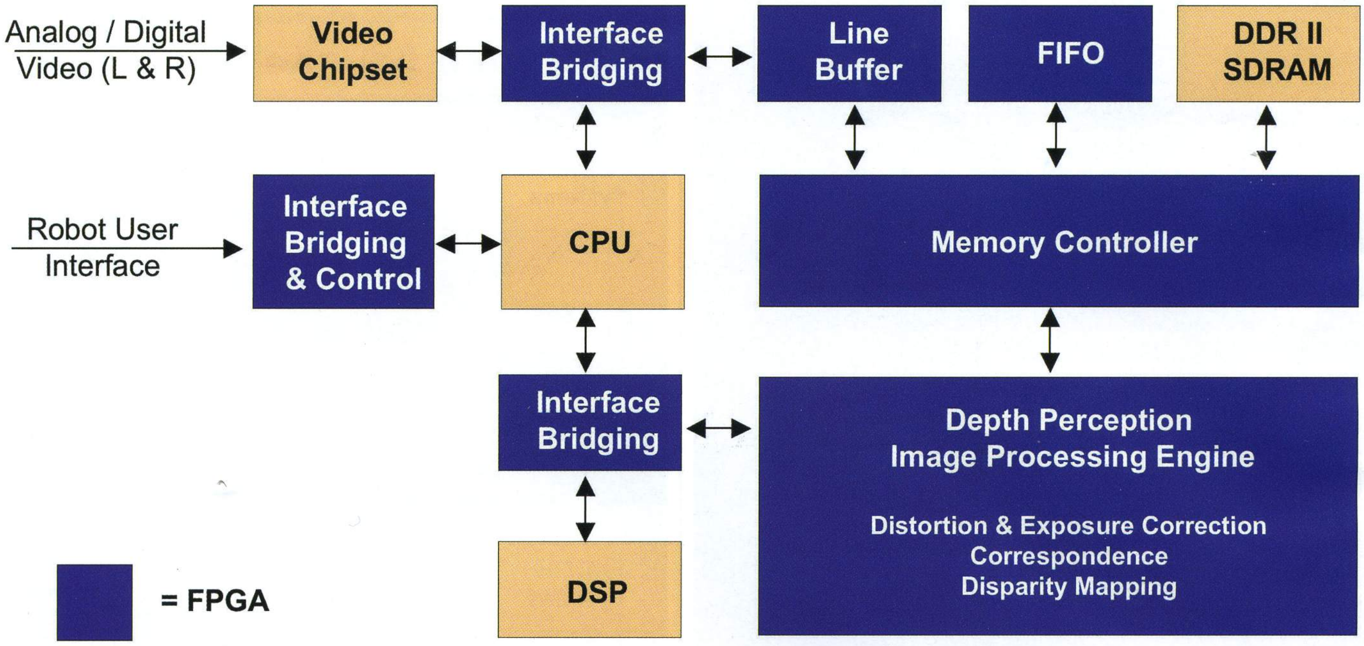

Fig. 2: FPGA supports computationally intensive disparity mapping algorithms

buffer memories and linking directly to image data from the cameras; and

- Security scheme to keep intellectual property from being stolen.

For image processing algorithms, traditional DSPs have limited parallelism, with one to four multipliers typical per DSP chip, as shown on the left side on Fig. 3. As a result, DSP chips compensate with very high clock rates, 1GHz and higher, to achieve high throughput. DSP algorithms that require complex serial tasks to be quickly completed should be implemented in the DSP.

However, many depth perception image processing functions are rich in parallelism, such as linear interpolation techniques, median filters and geometric projections. Compared with a DSP chip, FPGAs can accelerate system performance by executing serial functions in parallel, as shown on the right side of Fig. 3. For example, image correspondence is a very simple algorithm, comparing intensity values of pixels or patches of pixels between two images. Effectively a sum of squared differences (SSD), the image correspondence algorithm must be performed millions of times across the pixels in the image, a difficult task for a standalone DSP chip that executes functions serially.

Typically, FPGAs with embedded DSP functions include several blocks of multipliers, but some FPGAs also have embedded adders, subtractors and accumulators that can significantly increase processing performance. Although low-cost FPGAs typically operate at less than 300MHz system clock frequency, high DSP throughput (3000 MMACs) can be achieved by performing multiple DSP functions in parallel in the multiple DSP blocks on the chip.

Memory and DDR

As with multipliers, the flexibility to take advantage of parallelism in FPGAs is beneficial for memory access. Parallel accesses can easily be made to various memory types. On-chip distributed memory can be used to build small high-performance scratch pads, which could be useful for remapping pixels in the image using an LUT of coordinates for getting rid of fish eye. Larger blocks of on-chip embedded block memories allow larger high-performance memories to be created, useful for fifos and line buffers.

Finally, off-chip memory such as DDR II sdram can be used to provide large, relatively high performance memories. FPGAs can provide memory control and interface for the DDR sdrams, which are often used to store entire frames for image processing.

At lower (100MHz and below) clock speeds, the DDR memory controller interface is straightforward and can be implemented in an FPGA using general-purpose IO and logic capabilities. At higher frequencies, however, FPGAs with dedicated circuits are required to ensure a robust DDR memory interface. Not all FPGA families contain these dedicated circuits, and the cost and complexity of implementing high-speed DDR memory interfaces varies considerably, depending on the specific FPGA family.

LVDS

FPGAs interface the image-processing engine for depth perception with the cameras and video chipset via LVDS interfaces, such as a camera link. LVDS is a low-noise, low-power, low-amplitude differential signalling

continued on page 34

continued from page 32

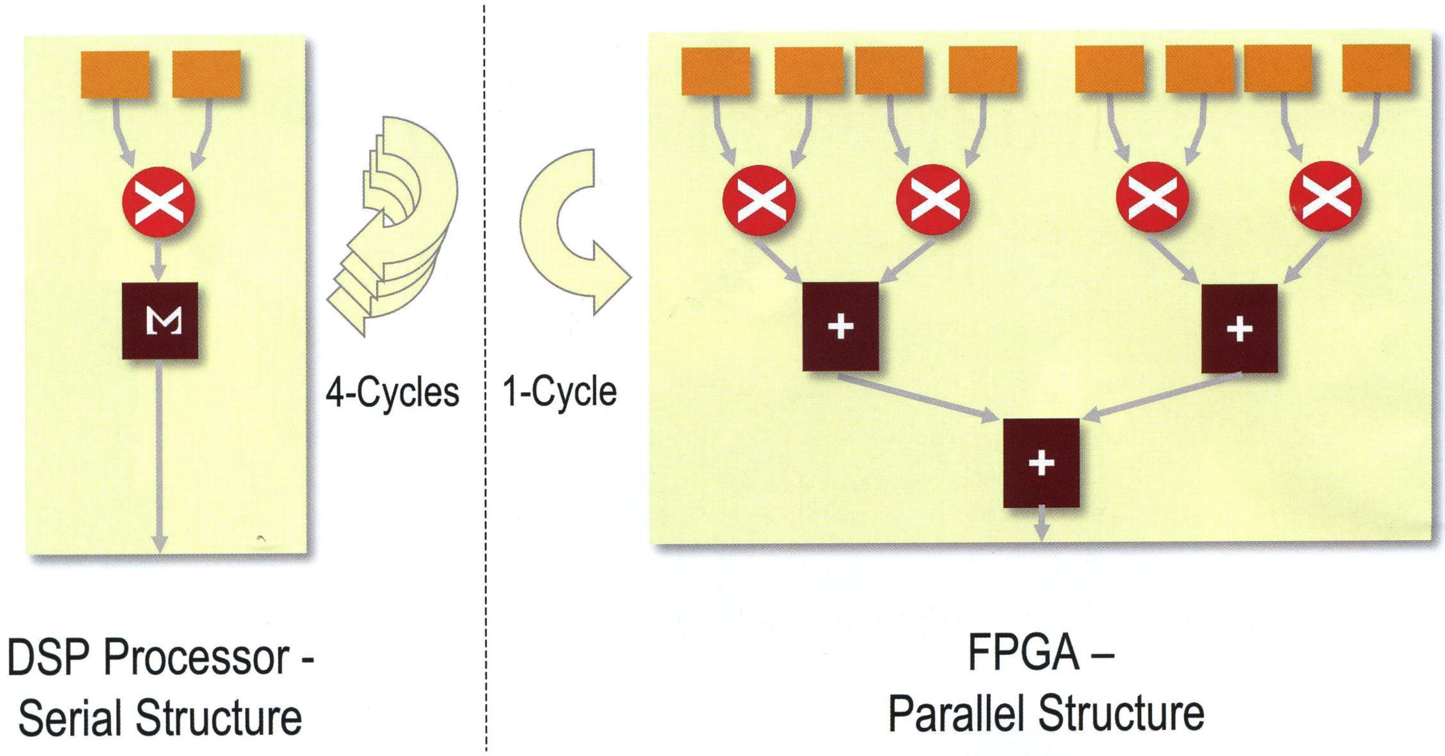

Fig. 3: DSP processor versus FPGA

> “A combination of CPU, DSPs and FPGA coprocessors will yield the optimal balance of performance and price for real-time image processing

method for sending high-speed (gigabits per second) data transmission over copper wire.

An example of an interface used for video is the 7:1 LVDS interface, which has multiple LVDS signal pairs for data and one signal pair for the clock. This is a native IO interface for some FPGAs. The video data enters on three LVDS pairs into the FPGA and the IO structure deserialises the packets and provides parallel data inside the FPGA to the logic array.

Security

Since autonomous robots are sometimes used in military applications, the FPGA should not introduce additional vulnerability to the system during configuration. For sram-based FPGAs, configuration data typically comes from an external non-volatile memory source. Some FPGAs have a built-in 128bit AES decryption engine, which prevents determined hackers from reverse engineering the functions inside the FPGA.

AES configuration bit-stream encryption provides protection as long as the encryption key is unknown; however, non-volatile FPGAs eliminate this security risk entirely. Some non-volatile FPGAs combine reprogrammable flash cells and sram cells on the same chip. The embedded flash memory is used for storing the device configuration securely on the chip. The sram holds the working configuration after power-up. This technology provides high configuration pattern security while delivering all the benefits of infinitely reconfigurable sram.

Summary

An FPGA that supports DSP, differential signalling standards such as LVDS and DDR II memory interfaces makes real-time depth perception in robots a possibility.

For a stereo-vision application in a robot, likely a combination of CPU, DSPs and FPGA coprocessors, will yield the optimal balance of performance and price for real-time image processing implementation. The FPGA would tackle the highly parallel disparity mapping and depth perception algorithms. A DSP would address more complex functions that required less parallelism, high speed and simple coding. The CPU would take the parts requiring the most flexibility and high-level software implementation. ■

Bart Borosky is a product marketing manager at Lattice Semiconductor