nal in the range of frequencies where waveguide losses are considerable. Also, guaranteeing a perfect alignment of the very small chips during assembly, in order to preserve the balanced nature of the multiplier, is practically impossible with the subsequent impact on the multiplier performance.

The design presented here overcomes these difficulties by performing the power-combining directly on-chip. Four E-probes are located at a single input waveguide in order to equally pump four multiplying structures (featuring two diodes each). The produced output power is then recombined at the output using the same concept. The four multiplying structures are physically connected on one chip, so that the align-

ment and symmetry of the circuits can be very well preserved. Contrary to traditional frequency triplers, in this design the input and output waveguides are perpendicular to the waveguide channels where the diodes are located. Therefore, the multiplier block is easier to fabricate with silicon micromachining technology instead of regular machining. The expected conversion efficiency of the tripler is ≈2 to 3% over a ≈20% bandwidth, which is similar to that which is simulated for an equivalent single-chip tripler driven with one fourth the input power.

FPGA Vision Data Architecture

This is an aid to any FPGA vision processing, and can be used by the automotive industry to detect collisions before they occur, and for robotic autonomous navigation for disaster relief.

NASA's Jet Propulsion Laboratory, Pasadena, California

JPL has produced a series of FPGA (field programmable gate array) vision algorithms that were written with custom interfaces to get data in and out of each vision module. Each module has unique requirements on the data interface, and further vision modules are continually being developed, each with their own custom interfaces.

Each memory module had also been designed for direct access to memory or to another memory module. On the development board originally used (an Alpha Data XRC4), there were six independent SSRAM (synchronous static RAM) banks that allowed each module sole access. For a flight mission, there likely would be between one and three memory banks, and arbitration of those banks would need to be supported, interleaving access to individual memory banks between multiple modules.

An FPGA data architecture was required to allow arbitration to onboard DDR (double data rate) and/or SSRAM memory, and to allow up to 10 to 30 independent agents access to that memory. It also required a method of exchanging data directly between modules without reducing the throughput of memory access. This architecture also had to support both low-latency reads and writes, and offer high throughput.

Each FPGA vision module had slightly different input and output re-

quirements. Some required serial access to data, and some were random access. There were 8-bit, 16-bit, and 32-bit input/output widths. Three modules could connect directly together in a series or go directly to memory, depending on runtime configuration options. One of the larger difficulties was posed by the random read access. For industry-standard buses such as AMBA, PLB, or OPB, a single random-read request can take 5 to 10 clock cycles, locking out all other users on the bus until the request was complete. This is far too slow for the vision modules and would effectively reduce performance by 2 to 5 times.

An architecture was created that met the same data throughput as the prior custom interface that had no arbitration. The new architecture also allowed for multiple memory types (DDR, DDRII, SSRAM, NAND memory) without any modification of the FPGA vision modules themselves.

The current Rover Navigation FPGA Vision system contains five vision modules: Rectification, Filtering, Disparity, Feature Detector (via a Harris detector), and Visual Odometry score computation (via a sum of absolute differences operator). Further modules to handle path planning are likely.

Each vision module has an "agent" — an interface to memory for both reads

and writes of different sizes. R32 means a read agent of width 32 bits, and W8 means a write agent of width 8 bits. Each memory bank has a single arbiter that handles all memory requests to its bank. Each agent maps to a single arbiter, but because this mapping will be dependent upon the memory devices used and the number of memory devices available (i.e. two DDR banks vs. six SSRAM banks), there is a large multiplexer called the "vision agent to bank mapping," which assigns agents to appropriate arbiters and memory banks.

Each agent can queue multiple memory requests and queue multiple responses from memory. This allows bursting of data for high throughput, and de-couples the action of requesting memory from the action of receiving data. Many of the vision modules have one part dedicated to computing the location of the next request, and a separate part dedicated to handling the data at that location.