FPGA Development for High Altitude Subsonic Parachute Testing

James E.Kowalski, Konstantin G. Gromov and Edward H. Konefat

Jet Propulsion Laboratory,

California Institute of Technology

FPGA Development for High Altitude Subsonic Parachute Testing

James E.Kowalski, Konstantin G. Gromov and Edward H. Konefat
Jet Propulsion Laboratory, California Institute of Technology

Abstract

This paper describes a rapid, top down requirements-driven design of an FPGA used in an Earth qualification test program for a new Mars subsonic parachute. The FPGA is used to process and control storage of telemetry data from multiple sensors throughout launch, ascent, deployment and descent phases of the subsonic parachute test.

The FPGA used was a Xilinx Virtex-E, embedded on a COTS PCMCIA card with dual SRAM buffers. Software on a SBC reads filled SRAM using Direct Memory Access (DMA), and stores the data in a flash disk.

The FPGA design is guided by a spreadsheet of memory partitions based on data rates from each sensor. Accumulators are used to compress some of the high rate data. A prioritized queue is used to control the servicing of received data from multiple sources. The memory transfer rate is high enough to allow single depth buffering.

Section I. covers requirements. Section II details methodology. Section III presents implementation results including functional verification, resource utilization, and timing.

Data post processing and reformatting is in Section IV. Section V describes some of the test results.

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

Introduction

This paper describes a rapid, top down requirements-driven design of an FPGA used in a qualification test program for a next generation Mars subsonic parachute(1). The field programmable gate array (FPGA) processes raw sensor data, and it controls storage of telemetry data from multiple sensors throughout; launch, ascent, deployment and descent phases of the subsonic parachute test. Using an FPGA was instrumental to a quick turn-around in this development, because it enabled modular reusability of heritage Verilog, VHDL source code and design libraries to meet the new system requirements. The system requirements specific to FPGA: development, design methodology, chip organization and verification are covered in this paper. The system was simulated, integrated and ground tested in a comparable environment prior to conducting the actual field tests. Three high altitude subsonic drop tests were conducted in the fall of 2004 from Ft. Sumner, New Mexico successfully recording of deployment, inflation, and inflated performance of the subsonic parachute with this new design.

System Requirements

The instrument set is designed and tested to operate and survive a wide variety of conditions. After ground-level power-on, it ascends to and dwells at high altitude. After release and a period of free fall, the parachute opens and descends to ground. Location and recovery could take as long as a day following impact. Data must be recorded from power-on to impact without loss. The following requirements are specific to the development of the FPGA:

1. Design and development window of four months.
2. Low mass, low power, PCMCIA form factor module.
3. Ability to receive and format data from various sources at various rates compressing by integration the highest rate data.
4. Store it to memory via DMA while continuing to receive and process new data for over 10-hours test duration.
5. Operate in a high altitude, low pressure and extreme thermal environment.

Each field test requires a switch on of the instrumentation 2.5 hours prior to launch, a typical 2-hour ascent and a 1-hour descent. The memory storage is designed for 10-hours test duration. Sufficient margin is included to account for possible delays in launch, or an atypical, prolonged period at the release altitude.

HASP FPGA Design Methodology

The methodology for this design was to use existing Verilog modules wherever possible, adapting as necessary. Heritage IMU data receiver and serial interfaces from MER were used with minor modification. The Annapolis Wildcard, with its VHDL library, was the design platform. Hierarchical design and verification were used, simulating each module before integrating to the next level, then simulating the next higher level.

There are no deep logic levels in this design. The clock cycle, at 25 ns, was fast enough to allow all data sources to be written to memory, even if all their packed 32-bit words became available at the same time. This meant that only a single level of queue depth was needed for each source. A clock cycle was long enough, since there are no complex data operations, to process the data with large margins. The HASP memory interface module is the most complex, and it is at the heart of the design for this new application.

FPGA Chip Organization

The High Altitude Subsonic Parachute FPGA has a VHDL shell, called the HASP PE. VHDL was used to be compatible with the Wildcard development system, which contains a wide library of tested design elements, including bus interfaces and DMA. The HASP PE used these for interfacing to the Wildcard memory chips and via DMA to the host computer. The HASP PE also maps the pins of the HASP_core verilog to the Wildcard PCMCIA I/O pins.

The HASP_core contains:

- the imu_data_rx, interface module for Northrop LN200 Inertial Measurement Unit, designed for MER and modified slightly,
- 6 Accumulators – summing four each of 16-bit delta x, y and z velocity and angle readings from imu_data_rx,
- the ADC interface, which controls two twelve bit analog to digital converters
- serial receiver for GPS data,
- serial receiver for magnetometer data, and
- the memory interface module.

The memory interface module contains:

- the memory interface state machine,
- downlink write state machine which formats and writing downlink packets,
- the downlink interface module,
- 6 18-bit accumulators to reduce the data frequency to downlink,
- buffer registers to build up 32-bit memory data words to be written into the partitions for each of the input data types,
- interface timeout counters for the downlink write state machine,
- time-tag registers,
- counters for indexing ADC data, and
- buffer_request and buffer_active registers.

The design strategy was to write data to memory in partitions which were sized to match the data volume and rates for each of the data sources. This was accomplished by using a spreadsheet (Table 1.) that calculated the partition beginning and end addressees for each data source.

The Wildcard's two 128K x 32 memories were used as ping pong buffers. When one of the data partitions became full, the last addresses for each partition were written into another partition, called last, to serve as pointers to the end of valid data when reconstructing the data stream. The write select then would be switched, so the newly filled chip could offload to the host computer via DMA transfer, while writing continued to the other memory.

The FPGA clock frequency is 40 MHz. The memory interface protocol for the Wildcard, over the LAD (Local Address Data) bus operated at 40 MHz, allowing fast transfers. The Wildcard has a programmable oscillator with a range from 0.5 to 100 MHz. The IMU delivers 13 16-bit words of data at 400 Hz, with a bit rate of 1.0152 Mbps transmissions. We used accumulators to sum the x, y, and z delta velocity and delta angle, decreasing the storage rate to 100 Hz. During a 10 ms interval, 16 12-bit samples are taken from each of two ADC's; 10 bytes of data would be transmitted by the GPS receiver (9600 baud). See Fig.1.

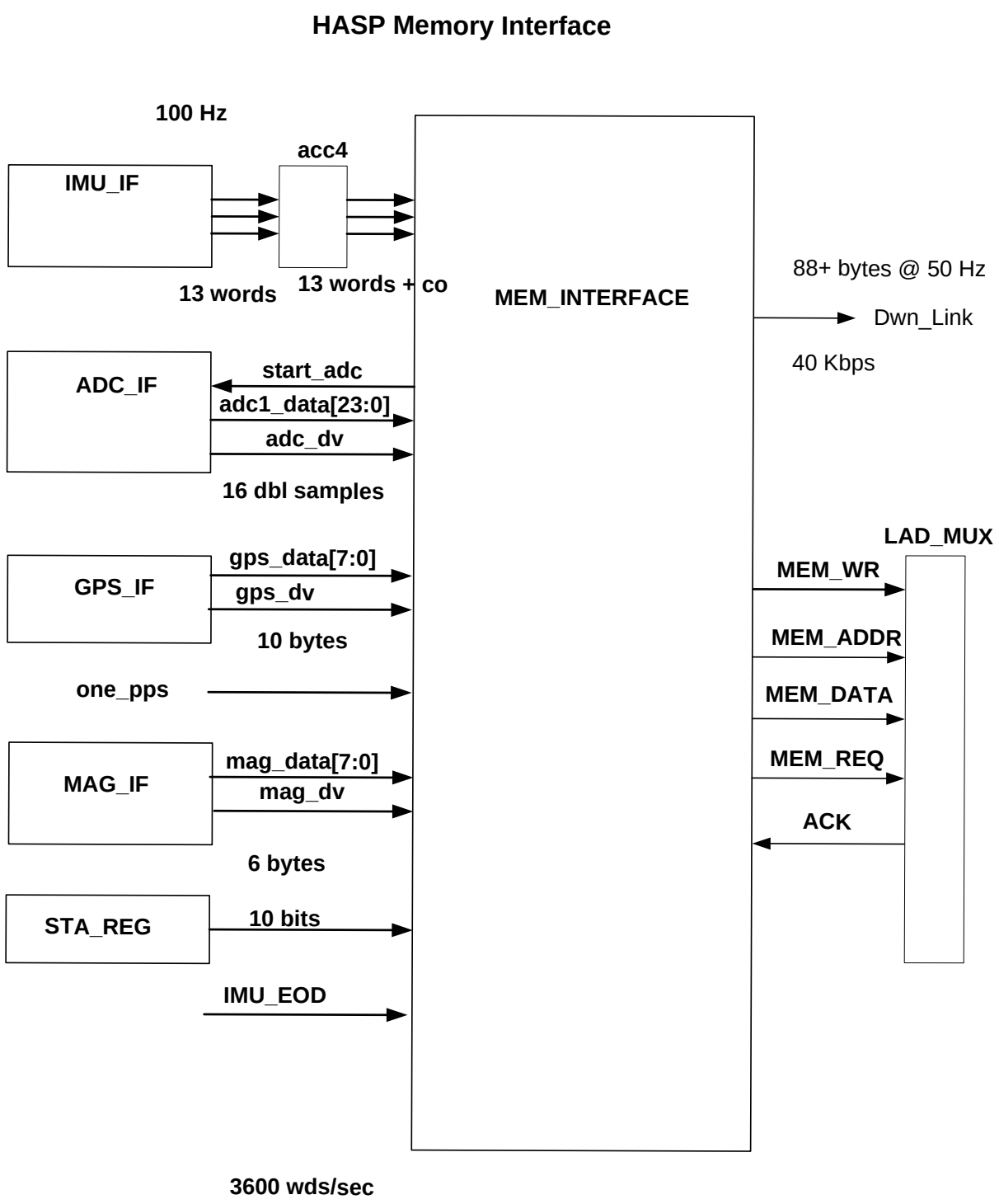

Fig.1 HASP FPGA Interfaces

HASP Memory Partitioning

	pkts 4440	wds/pkt	partition size	**start**_dec	hex	**end**_dec		hex
imu		5	22200	1	1	4	22200	56B8
adc		16	71040	22201	56B9	15AE4	93240	16C38
gps		2.5	11100	93241	16C39	5B0E4	104340	19794
mag		4	17760	104341	19795	65E54	122100	1DCF4
sta		2	8880	122101	1DCF5	773D4	130980	1FFA4
last_wr			6	130981	1FFA5	7FE94	130986	1FFAA
		total	130986					

J. Kowalski
9/9/2004

Table 1. HASP Memory Partitions

Note: The packets (pkt) referenced in the spreadsheet are the amount of data from each source during a 10 ms interval. The IMU outputs its raw data at 400 Hz. Four consecutive outputs were summed in 6 accumulators in the memory interface module. The integrated data rate of 100 Hz became the standard unit for the spreadsheet.

Implementation and Verification

Simulation

The simulation approach used was hierarchical verification. Testbenches were constructed at low level for functional verification of the design. When a module had been thoroughly simulated, the design was incorporated into the next level, and re-simulated. A preliminary version of the HASP memory module was written and simulated using input test data before the other data source modules were ready. This simulation was sufficient for the accumulators to pass on data, write to memory and exercise the downlink function. The memory buffer switchover was verified by starting a simulation with the addresses near their maximum.

Hardware Implementation

As each interface module was developed, after thorough simulation at block and chip level, synthesis was performed with Synplify Pro. FPGA physical design processing was done with the Xilinx Webpack place and route software. See Table 2.

Timing Verification

The unit was designed with a clock sufficiently fast to meet the requirements, but slow enough to allow generous slack in the data path to conserve power. The 19-bit accumulator adders were the longest logic paths. The Xilinx place and route software was given aggressive target of 10 ns cycle time to meet, while the actual cycle need was 25 ns. Only one net failed to meet the constraint, and it achieved a cycle time of 17.6 ns. As a sanity check, limited back annotated simulation was performed.

Environmental Testing

The field test operating conditions include high altitude Earth atmosphere, low pressure environment to replicate a Mars entry conditions at subsonic speeds. The FPGA has to survive and operate in this near vacuum. Some modifications were made to enhance the host SBC's conductive thermal path allowing heat to be dissipated in the absence of air that would cool the device. A full thermal vacuum test was conducted to verify FPGA operation in near vacuum for an extended duration.

Device Utilization

Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops:	3,214	6,144	52%	
Number of 4 input LUTs:	3,510	6,144	57%	
Logic Distribution:				
Number of occupied Slices:	2,824	3,072	91%	
Number of Slices containing only related logic:	2,824	2,824	100%	
Number of Slices containing unrelated logic:	0	2,824	0%	
Total Number 4 input LUTs:	3,808	6,144	61%	
Number used as logic:	3,510			
Number used as a route-thru:	264			
Number used as Shift registers:	34			
Number of bonded IOBs:	200	260	76%	
Number of Tbufs:	1	3,200	1%	
Number of Block RAMs:	10	32	31%	
Number of GCLKs:	3	4	75%	
Number of GCLKIOBs:	3	4	75%	
Number of DLLs:	2	8	25%	
Number of Startups:	1	1	100%	
Number of RPM macros:	2			

Routing Performance Summary

Property	Value
Final Timing Score:	1205701
Number of Unrouted Signals:	All signals are completely routed.
Number of Failing Constraints:	0

Failing Timing Constraints (total failing = 1)

Constraint(s)	Requested	Actual	Logic Levels
* NET "clocks_in.m_clk" PERIOD = 10 ns HIGH50%	10.000ns	17.066ns	5

Table 2. Xilinx Place and Route Results

FPGA and Sensor Interface Test

The FPGA interfaces were connected to each sensor: IMU, GPS and ADC units. The tests ran for ten hours, showing that the data could be stored and retrieved using post processing software.

Post Processing

An executable file was compiled using C++ to aid in data reconstruction, because data was stored to flash disk in a packed format. The post processing code used the partition start addresses and the last address pointers that were saved for every buffer switchover. It unpacks, formats, and writes each of the data types into its own separate file, reconstructing the original data stream.

Field Test Results

Three high altitude subsonic drop tests were conducted in the fall of 2004 from Ft. Sumner, New Mexico successfully recording of deployment, inflation, and inflated performance of the subsonic parachute with this new FPGA design. The data was retrieved for post flight analysis and reconstruction of events. The rapid development of this FPGA was vital in obtaining the data for understanding the physical environment of deployment, inflation, and inflated performance of the parachute (1,2,3).

Conclusion

The use of an FPGA was shown to be instrumental to a quick turn-around in this development. The memory partition spreadsheet enabled quick design modifications as data sizes and rate requirements evolved. Heritage Verilog and VHDL augmented original design to successfully meet the new system requirements**.**

References

1. **High Altitude Test Program for a Mars Subsonic Parachute**
   R. Mitcheltree,PhD., R. Bruno, E. Slimko, C. Baffes, and E. Konefat, NASA Jet Propulsion Laboratory; and A. Witkowski, Pioneer Aerospace Corporation, South Windsor, CT
   AIAA-2005-1659
   18th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar, Munich, Germany, May 23-26, 2005

2. **Opportunities and limitations in low earth subsonic testing for qualification of extraterrestrial supersonic parachute designs**
   A. Steltzner, J. Cruz, R. Bruno, Dr. R. Mitcheltree
   NASA Jet Propulsion Laboratory
   AIAA Aerodynamic Decelerators Conference , May 20-22, 2003
   Parachutes for Mars and other planetary missions often need to operate at supersonic speeds in very low density atmospheres. Flight testing of such parachutes at appropriate conditions in the Earth's atmosphere is possible at high altitudes.
   http://techreports.jpl.nasa.gov/2003/03-1289.pdf
   Updated/Added to NTRS: 2004-08-20

3. **Mars Science Laboratory: entry, descent and landing system overview**
   J. W. Umland, A. Chen, E. Wong, T. Rivellini, Dr. B. Mitcheltree, A. Johnson, B. Pollard, M. Lockwood, C. Graves, E. Venkataphy
   NASA Jet Propulsion Laboratory
   2004 IEEE Aerospace Conference , March 6-13, 2004

   http://techreports.jpl.nasa.gov/2003/03-2088.pdf
   Updated/Added to NTRS: 2004-09-03