University of Applied Sciences Pforzheim Develops Radar Sensor System Using Vitis™ HLS in AMD Zynq™ SoC

Sensor Can Detect the Range and Speed of Objects for Media Sensors, Road Condition Detection, Water Flow, and Other Applications

PARTNER

HS PF

INDUSTRY

Industrial

CHALLENGES

Pforzheim University wanted to build an embedded radar sensor system to detect the range and speed of objects in multiple applications.

SOLUTION

The solution included a high-frequency frontend with sufficient bandwidth, and a signal-processing backend with ample compute performance, built on the AMD Zynq Z-7030 SoC.

RESULTS

The solution achieved its goals, using only 25% of the Zynq 7000 Z-7030 SoC resources, allowing additional free resources to be allocated for post-processing algorithms. Also, the Vitis™ HLS design tool helped simplify hardware implementation.

AMD TECHNOLOGY AT A GLANCE

AMD Zynq™ 7000 Z-7030 SoC

The research team of the University of Applied Sciences Pforzheim (UAS Pforzheim), in cooperation with other German universities, developed an embedded radar sensor system. The sensor system can detect the range and speed of objects and can be configured to different applications like media sensors, road condition detection, or water flow measurements.

CHALLENGE

In order to adapt the radar system to different applications, a high-frequency (HF) frontend with sufficient bandwidth and dynamic range was needed as well as a signal-processing backend with sufficient computing performance. The HF frontend was developed by University Ulm and contains a 160-GHz-Up-Down-Converter as well as necessary HF signal generation with a broadband VCO and PLL. The signal-processing backend was developed by the research team at the UAS Pforzheim and includes the clock distribution, the analog low-frequency signal processing and the digital signal processing and control of the sensor system as shown in Figure 1 and is based on an AMD Zynq™ 7000 device, specifically the Z-7030 SoC.

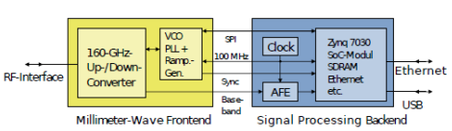

Figure 1: Block diagram of sensor hardware.

In order to configure the system and to visualize the resulting data, a web server runs under Linux on the Zynq device, such that the user can access the system using an internet browser.

The sensor works as an FMCW-radar (Frequency Modulated Continuous Wave Radar). According to the "chirp-sequence" principle, the sensor sends several linear frequency-modulated "ramps." Each ramp is sampled with N samples and a measurement consists of M ramps. The range of objects can be detected through a time-of-flight measurement for each ramp, by comparing the frequency of the reflected wave with the current frequency of the ramp. The requirement was a 100 MS/s sampling rate and a 10 Hz measurement rate. The signal processing consists of a 2-dimensional FFT (Fast Fourier Transform) which generates the radar "image" and some additional post processing.

SOLUTION

Figure 2 shows the system concept for the Zynq SoC. The software runs on the Zynq processor system (PS). The programmable logic (PL) is used for a JESD204 interface which is connected to the analog front end (AFE). Using a control IP core, the data is streamed to the first FFT IP core and written to the SDRAM of the system. The second FFT IP core reads the data generated by the first FFT and calculates, again, an FFT and stores the result back to the SDRAM. The data has to be stored in the

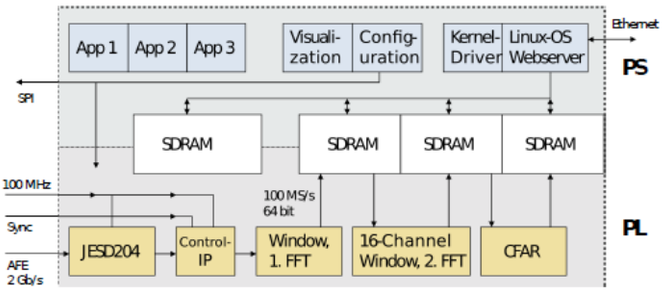

Figure 2: System concept for Zynq SoC.

external SDRAM since the capacity of local Block RAM is not sufficient. For example, with 64 ramps featuring 4,096 samples in each ramp and a bit width of 64 bits per sample, the result is 2 MBytes of data. Finally, the result of the 2D-FFT can be post-processed, e.g. by a Constant False Alarm Rate (CFAR) algorithm.

Figure 3 explains how the data is written to main memory and the relation of the memory address to the data matrix. The addresses are relative addresses to the beginning of the data section in the main memory. The data of the first ramp is written from address 0 to address N-1 and the next ramp of data starts subsequently at address N and so forth, such that all ramps are stored linearly in main memory.

The benefit of the described memory layout is that the FFT1 core can write data linearly to memory, which enables a maximum throughput when processing the incoming ramp data since data can be transferred by "burst transfers" over the bus system. The memory layout on the other hand is not optimal for FFT2. As described in Figure 3 the FFT2 has to be calculated over the columns of the result matrix of FFT1.

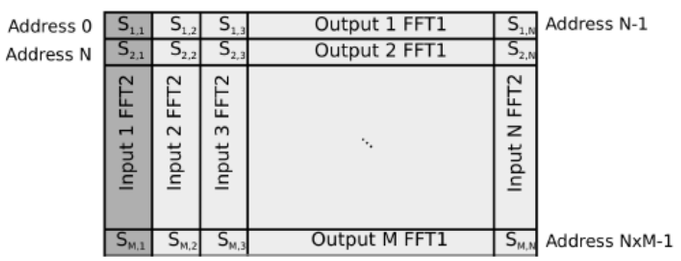

Figure 3: Result matrix after the first FFT and the relation to the memory addresses. Addresses are relative to the beginning of the matrix in memory and are addresses of the samples (= 8 Byte).

This means that for FFT2 the input matrix is in transposed order, so accessing the next sample in a column results in a jump in memory. For example, if we have a size of 4,096 points for FFT1, the jump distance to the next sample is 4,096 x 8 Byte = 32 kByte. If we want to process one column via FFT2, each sample access is a single transfer over the bus and this will result in poor throughput.

In order to optimize the throughput, FFT2 was implemented with 16 parallel channels, which was done by using two FFT cores with 8 channels each. This has two advantages: We can now process 16 columns in parallel, and since we therefore have to read 16 sample values per row, the data can be read by burst transfers with 16 beats per burst transfer, which is already the maximum burst size for the used AXI bus system.

The whole concept of the memory layout for FFT1 and FFT2 and the data transfer mechanisms are depicted in Figure 4.

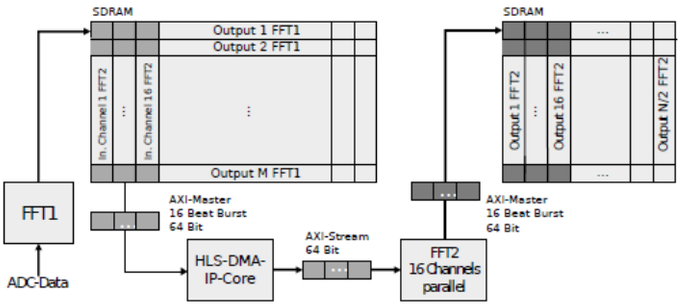

Figure 4: FFT cores, memory layout, and data transfers.

For the memory access for FFT2, a special DMA core was developed that accesses the result data of FFT1 as described above and feeds the 16 channels of FFT2 over an AXI stream interface. The DMA core reads 16 samples of a row and then jumps in memory to the next row of the matrix containing the samples, which belong to the same column in the matrix. When all samples of 16 columns are processed, the DMA core proceeds to the next 16 columns until all columns of the matrix are processed.

The IP cores were developed with the help of the AMD Vitis™ HLS tool. For each IP core, a corresponding C/C++ model was developed. Although it would have been possible to implement the intended functionality with the available AMD IP cores (FFT, DMA), HLS was used for the following reasons:

- The number of FFT cores and their configuration can be easily changed in the C-Code.
- The DMA function (AXI master) to access the SDRAM main memory as well as the AXI-Stream and AXI-Lite interfaces can be simply described with HLS pragmas in the C Code and are generated automatically by Vitis HLS.
- The distribution of the input data, which are in transposed order to the channels of FFT2, can be coded more easily in C than in VHDL. Also, the address arithmetic needed to access the transposed input matrix of FFT2 is easier to code in C.
- Verification of the functionality of the IP cores can be easily done with a C-testbench in Vitis HLS, compared to a testbench being written in VHDL. The input stimuli and the reference data for the verification simulation have been generated with Matlab.

Figure 5 shows the inner structure of the FFT1-IP-Core. The function copy_input basically reads the input data, multiplies them with the FFT window data and sets the imaginary part of the data to zero and finally feeds the FFT. The function copy_output writes the output data of the FFT to the main memory of the system. The output function implements an address arithmetic, such that each FFT output sample (real and imaginary) of a ramp or chirp will be copied sequentially to the memory.

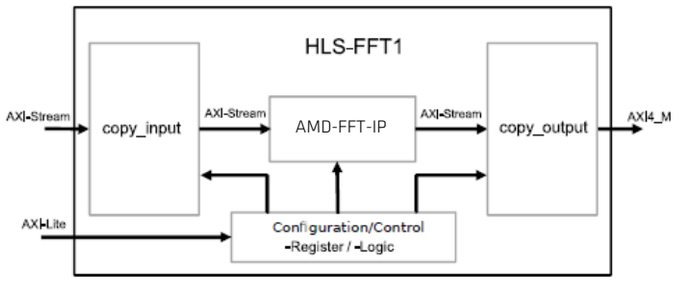

Figure 5: FFT1 IP core.

Also, the ramps/chirps will follow sequentially each other in memory. When all ramps are processed the address pointer will wrap around, such that the next measurement will overwrite the previous one.

RESULT

The measured execution time for the first FFT was 3.4 ms and 5 ms for the second FFT for a configuration of 64 ramps and 4,096 samples per ramp (with 100 MHz clock rate). Considering the maximum configuration of 256 ramps with 4,096 samples per ramp, we can estimate an execution time of 14 ms for FFT1 and 27 ms for FFT2, due to the N × log(N) scaling of the FFT. Therefore it is possible to reach the intended measurement frequency of 10 Hz also for the maximum configuration.

The resource usage of the Zynq PL (flipflops, LUTs, BRAM, DSP) is around 25%, depending on the type of resource, meaning that there are enough free resources for the implementation of additional post-processing algorithms.

The 2D-FFT was implemented by two FFT-IP cores, which calculate two subsequent FFTs over the input data matrix and result in a "Range- Doppler-Map" (radar "image"), which can be used for further radar target detection algorithms, like CFAR. The performance penalties due to the transposed memory layout for FFT2 have been solved by using 16 channels for FFT2, resulting in burst bus transfers and an improved throughput due to the parallel processing in the channels. The developed signal processing backend system was successfully tested together with the 160 GHz frontend in real applications.

Using High-Level Synthesis instead of a VHDL design for the hardware implementation has been proven to be useful. Different architecture variants have been implemented with much less engineering effort, compared to a VHDL implementation. Also the implementation of the special address arithmetic and the slave and master bus interfaces is easier to implement with C/C++ code and HLS and can also be verified easier, compared to a VHDL design.

WANT TO LEARN MORE?

About AMD Zynq SoCs

About Vitis HLS

About Pforzheim University

About Pforzheim University

Pforzheim University with its three schools - the School of Design, the School of Engineering and the Business School - enjoys a first-class reputation. The schools combine creativity with business education and technical precision. This combination also makes the university an attractive science and research partner for the regional and national economy. With around 6,200 students, Pforzheim University - founded in 1877 - is one of the largest universities of applied sciences in the state of Baden-Württemberg. For more information, visit https://www.hs-pforzheim.de/en/.

About AMD

For more than 50 years, AMD has driven innovation in high-performance computing, graphics, and visualization technologies. Billions of people, leading Fortune 500 businesses, and cutting-edge scientific research institutions around the world rely on AMD technology daily to improve how they live, work, and play. AMD employees are focused on building leadership high-performance and adaptive products that push the boundaries of what is possible. For more information about how AMD is enabling today and inspiring tomorrow, visit the AMD (NASDAQ: AMD) website, blog, LinkedIn, and Twitter pages.