A COMPARISON OF EMBEDDED RECONFIGURABLE VIDEO-PROCESSING ARCHITECTURES

C. Claus, W. Stechele

Institute for Integrated Systems
Technische Universität München
Theresienstrasse 90, München, Germany
Christopher.Claus@tum.de,
Walter.Stechele@tum.de

M. Kovatsch, J. Angermeier, J. Teich

Department of Computer Science 12
University of Erlangen-Nuremberg
Am Weichselgarten 3, Erlangen, Germany
Matthias.Kovatsch@informatik.uni-erlangen.de,
Josef.Angermeier@informatik.uni-erlangen.de,
Juergen.Teich@informatik.uni-erlangen.de

ABSTRACT

Using Field Programmable Gate Arrays (FPGAs) as accelerators for image or video processing operations and algorithms has gained increasing attention over the last few years. One reason for that is FPGAs are able to exploit both temporal and spatial parallelism. In this paper two platforms for FPGA-based real-time image and video processing are presented and compared against each other. With both of these platforms it is possible to update the physical resources during run-time by exploiting the dynamic partial reconfiguration capabilities of Xilinx Virtex FPGAs. The analysis of both platforms with respect to their benefits and drawbacks has led to the concept of an optimal FPGA-based dynamically and partially reconfigurable platform for real-time video and image processing.

1. INTRODUCTION AND RELATED WORK

To exploit the benefits of FPGAs in the image processing domain is an ongoing field of research. The authors in [1] analyzed the requirements of video processing systems with respect to the available embedded memory resources (RAM Blocks) on an FPGA. The usage of large RAMs might result in unused memory in the allocated Block RAM resources which can be avoided using an time-multiplexed architecture. In [2] the author describes the implementation of sophisticated imaging modules on low-cost devices such as Spartan3. The benefits are mentioned when using FPGAs in the context of image enhancement technology for display applications. The FlexFilm architecture [3] is used for computationally intensive digital film processing. It is possible to process images with a resolution of 2048x2048 pixels in real-time. This architecture is an extensible FPGA based system. Each processing element consists of four Xilinx Virtex-II Pro FPGAs each equipped with 4 gigabit on-board DDR SDRAM. Several processing elements can be interconnected via PCI-Express. There are many publications which describe how dynamic partial reconfiguration can be exploited in fine-grained FPGA-based image- and video-processing systems. Most of them present rather concepts than an actual implementation of the proposed approach.

Sometimes it is meaningful and necessary to divide the algorithms for image or video processing into a hardware and software part. This is especially beneficial if the algorithms are not standardized and quick changes have to be made quite often. In that case only the performance intense parts, that are not meant to be changed, are implemented in hardware using a fine-grained processor array. An example for such computationally intense parts is the extraction of feature points (corners or lightspots) or image segmenting. The remainder of the algorithms can be implemented in software on general purpose CPUs to allow quick updates without re-synthesizing the whole system as required by pure hardware implementations. Thus a combination of processors for post-processing and a fine-grained processor array for pre-processing is favourable. In this paper two platforms and architectures for real-time video processing on reconfigurable hardware are presented. In order to compare these two platforms, an algorithm from the AutoVision project [4] to detect cars in dark environments, which is described in detail in [5], was partitioned into a hardware and software part and implemented on the Erlangen Slot Machine (ESM) [6] and on an XUP board from Xilinx.

The paper is organized in the following manner: In sections 2 and 3 the HW acceleration on the ESM and the dataflow through the system are described. In sections 4 and 5 the HW acceleration and the dataflow on the XUP board are explained respectively. The benefits of both platforms are combined and an optimal platform is proposed in section 6. Finally, section 7 concludes this paper with an outlook on future research activities.

978-1-4244-1961-6/08/$25.00 ©2008 IEEE.

2. HARDWARE ACCELERATION ON ESM

The main idea of the ESM [7] is to accelerate the application development as well as the research in the area of partially reconfigurable hardware (see Figure 1).

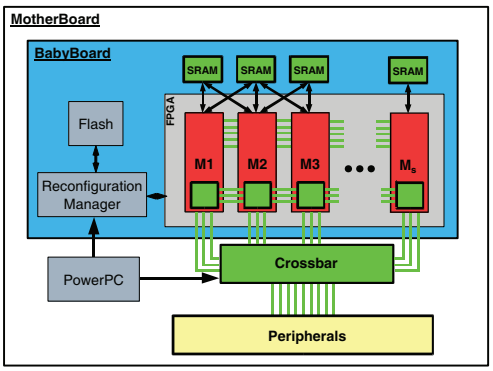

Fig. 1. ESM architecture overview.

The hardware implementation on the *Main FPGA* consists of a deinterlacer, filter specific logic, a module for hardware-software-communication with the PowerPC, and output logic. The convolution part inside the filter specific logic can be compared to the hardware accelerator of the AutoVision project. As a result, the used resources on the Virtex-II 6000, given in table 1, exceed those of the implementation on the XUP board.

Number of Slices:	8051	out of	33792	23%
Number of Slice Flip Flops:	2979	out of	67584	4%
Number of 4 input LUTs:	14142	out of	67584	20%
Number of BRAMs:	55	out of	144	38%
Minimum period: 7.116ns (Maximum Frequency: 60.520MHz)				

Table 1. Device Utilization on an xc2v6000 and timing summary of the hardware module

Additionally, SRAM memory directly attached to the top of the Main FPGA is engaged for deinterlacing the camera pixels and as buffer for the video engine modules. In case of the taillight detection module a complete, but down-sampled frame is stored, due to the fact that the number of BRAMs available on the Virtex-II 6000 limits the image size. However, this drawback can be eliminated by the use of convolution FIFOs which has already been tested in other video filters on the ESM. Here, the number of buffered lines can be freely configured. With the same utilization of BRAMs and 8bit color depth for instance, 37 lines with 2048 pixels each could be stored. Moreover, it should be mentioned that the ESM is capable to run other hardware modules in parallel since the video processing only occupies a bit more than half of the chip area. They can be partially reconfigured without interrupting the video filter. The same applies the other way around. Here, only the engine specific logic has to be replaced. The partial reconfiguration is handled by the *Reconfiguration Manager FPGA* which can directly load bitfiles from an attached flash memory (see Figure 1).

3. DATAFLOW ON THE ESM

The arriving interlaced frames from the video input processor are buffered in the SRAM memory and deinterlaced on the Main FPGA, before they are forwarded to the video filter. Since the deinterlacer passes RGB data to support any kind of filter, a conversion to grayscale is done and a copy of the image is down- sampled to 320x240 pixels to be stored in the BRAMs. The displayed video itself remains at 640x480 pixels. To do the matrix convolution, 16 parallel, dual-ported BRAMs are used which allow to access 32 pixels simultaneously. One pixel per clock cycle is processed and buffered in the external SRAM afterwards. In the next step the feature points from the convoluted image in the SRAM are extracted and sent to the PowerPC over the hardware-software-communication.

After all points were received and evaluated the result is sent back to the *Main FPGA* as a parameterized list of objects which must be incorporated into the video stream. This is done by the visualization routine, which draws the results of the feature point extraction into the image. Here, the processing of the video engine ends and an output logic relays the video stream and control signals to the framebuffer back through the Crossbar. As framebuffer a Spartan-IIE400 is used, which operates two SDRAMs, that are used to buffer the image for the RAMDAC. After a complete processed frame is received it can finally be displayed.

Image Resolution	color depth	theoretical proc. time (50Mhz)	measured proc. time (50Mhz)
320x240	8	1.536 ms	1.55 ms
384x288	8	2.211 ms	2.23 ms

Table 2. Theoretical and measured performance on the ESM

The performance measurements in table 2 are related to the matrix convolution of the filter module. As more processing is done in hardware and no bus is involved, the system can be run at a lower frequency and still meet real-time conditions. The theoretical processing time is calculated from the number of pixels times the cycle time (20 ns in this case).

The big FPGA on the ESM offers enough room to execute other hardware tasks on it. The reconfigurable resource may be used more efficiently because of the more flexible placement. Due to the crossbar connector, the video engines can be placed at different positions and dynamically route the IO peripherals to the the corresponding hardware module locations.

4. HARDWARE ACCELERATION ON XUP

Instead of storing complete images in the precious on-chip memory (BRAMs), in the AutoVision architecture a local memory is used to store up to sixteen complete image lines for operations on pixels and their neighborhood. This decreases the number of utilized BRAMs significantly as can be seen in table 4.

Number of Slices:	2884	out of	13696	21%
Number of Slice Flip Flops:	2803	out of	27392	10%
Number of 4 input LUTs:	4932	out of	27392	18%
Number of BRAMs:	36	out of	136	26%
Minimum period: 8.752ns (Maximum Frequency: 114.266MHz)				

Table 3. Device Utilization on a XC2VP30 and timing summary of the HWaccelerator without PLB interface

This values are related to a HW accelerator for taillight detection that is able to process 8 bit pixels. It is configurable to support also 16, 32 or 64 bit pixel, which is necessary if along with the pixel data some additional data has to be transferred. In its local memory 16 complete image lines (each up to 1024 pixels) can be stored which is sufficient for a 15x15 pixel neighborhood. In addition the HW accelerator is attached to the PLB via a Master interface with DMA capabilities. The DMA transfers greatly offload the CPUs. This architecture with one or more central busses and the on-chip CPU enables the possibility to access the pixel data from both HW accelerator and CPU. This is especially beneficial, if the software part of the algorithm has to access pixel data again, e.g. when a license plate between two taillights has to be found. In addition it is possible to draw detected features or objects directly into the result image from the software. One drawback is that no separate framebuffer is available. The video input sends 25 frames per second to the main memory. If an output rate of 60 Hz is considered, this means that every frame has to be transferred 2.4 times over the PLB to be displayed on a monitor. This drastically increases the busload, which could be avoided using an off-chip framebuffer. In that case the image has to be transferred over the PLB only once.

5. DATAFLOW ON THE XUP BOARD

First, the grayscale images taken by a camera are converted from analog to digital and transferred into the DDR SDRAM using burst transfers. Image sizes with a resolution up to 1024x1024 pixels are supported. In Figure 2 a block diagram of the system is shown.

After a complete image was transferred to the DDR SDRAM a hardware accelerator engine can access the corresponding pixels, process them and write them back.

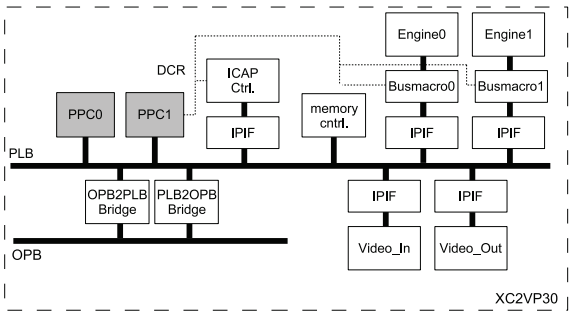

Fig. 2. Blockdiagram of the AutoVision architecture on an XC2VP30 FPGA

The accelerator engine can be seen as a wrapper around a user logic. The engine is responsible for read and write transfers of pixel data from and to the DDR SDRAM. The user logic determines if the hardware accelerator is responsible for e.g. taillight detection. Compared to the remainder of the engine, the user logic can be seen as separate module, because it includes the actual pixel processing operation. Due to inherent parallelism it possible to process a single pixel together with its neighborhood in one clock cycle. Some of the performance measurements are shown in table 4.

Image Resolution	color depth	theoretical proc. time (100Mhz)	measured proc. time (100Mhz)
320x240	8	0.768 ms	0.801 ms
640x480	8	3.072 ms	3.145 ms

Table 4. Theoretical and measured performance on the XUP board

The theoretical processing time is calculated from the number of pixels times the cycle time (10 ns in this case). As can be seen in table 4 the measured time to process one image is just slightly higher than the theoretical processing time. This is due to the fact that the input local memory has to be filled before processing the first pixel and the some additional time is required to write the processed pixel back into the DDR SDRAM.

The DDR SDRAM is also the desired place to store the configuration information namely the bitstream data. Using the concept described above it is also possible to modify the bitstream data via software if desired. One PPC is responsible for the configuration management. But instead of managing the transfer data from the memory to the configuration port, the PPC just sends the start address of the bitstream in the memory along with the number of 128 byte bursts to be performed to the ICAP controller, an IP core responsible for the reconfiguration. The ICAP controller is a bus master on the PLB. After it has obtained the necessary data from the PPC it fetches the bitstream data using direct memory access.

6. AN OPTIMAL PLATFORM FOR RECONFIGURABLE EMBEDDED VIDEO PROCESSING

The examination of both platforms along with their benefits and drawbacks has led to the idea of an optimal platform for reconfigurable embedded video processing which combines the advantages while avoiding the disadvantages. First, of all dedicated hardware managing the video input data and the deinterlacing process should be spent. If the video input data arrives in analog format the hardware should digitize all incoming video signals and provide the data to the main board. In addition if two half frames (fields) arrive the dedicated hardware should be capable to deinterlace the video data in a way that "tearing" effects (motion artefacts) are reduced to a minimum. Another solution would be to use full frames in already digitized data as provided e.g. by a standard webcam. In that case the dedicated hardware should include a JPEG decompressor so that the pixel data can be stored in the on-board memory in raw format. The results for embedded video processing have shown that it is beneficial to use fast RAM (on-chip block RAM) for convolution. Instead of loading the same pixel various times from the main memory, storing a few pixel lines inside a local memory attached to the hardware accelerator provides a resource efficient alternative, which is almost as fast as storing the complete image inside the precious on-chip block RAM. Therefore the optimal platform should comprehend enough block RAM distributed over the whole FPGA to place the hardware accelerators at any desired location. To enable the access to the image data from both, hardware and software, a on-chip CPU is beneficial. A central bus or crossbar which connects the main memory, the CPU and the accelerator is one possible solution.

One necessary and important design consideration is to use a random access framebuffer to display image data on a monitor using standard frequencies. This is necessary in order to cope with the problem that the image data has to be transferred multiple times over a central bus or crossbar if the output frequency (60 Hz, 75 Hz etc.) is higher than the input frequency (25 Hz, 30 Hz etc.). As soon as an image has been processed it can be written to the framebuffer and the data can be sent to the output at any desired frequency. By using dedicated hardware for the video input and such a frame buffer, the designer does not have to take care about how the image data is coming onto the board and how the processed data is displayed. Hence, the designer could concentrate on the development of the accelerators.

Independent if an on-chip CPU or a separate FPGA for reconfiguration management is used, one has to make sure that the reconfiguration is as fast as possible so that during this process no image frame is dropped. This requires that either the Internal Configuration Access Port or the Select MAP interface has to be fed with the maximum number of bytes of configuration data each clock cycle. Thus, the latency to access the memory, where the configuration data is stored, should be minimized.

7. CONCLUSION AND FURTHER WORK

In this paper two different platforms for video-processing using dynamic partial reconfiguration were presented. The results show that both platform are useful but dependant on the application to be performed. On the ESM the reconfiguration of modules is more easy. By contrast, on the XUP board a mixed HW/SW infrastructure with higher clock frequencies is easier to implement on the XUP board, due to on-chip CPU hardcores. As both platforms have their advantages and disadvantages the authors have proposed an optimal platform for embedded reconfigurable video processing that combines the advantages of both platforms to easy the development of hardware accelerators.

In future it is planned to further improve the sharing of hardware and software sources of the implementations on both platforms. Increased modularization into platform dependent and independent parts, respectively, will help to easily port new video engine modules between the different platforms.

8. REFERENCES

[1] N. Lawal and M. ONils, "Embedded FPGA memory requirements for real-time video processing applications," *NORCHIP Conference, 2005. 23rd*, pp. 206–209, Nov. 2005.

[2] M. Tusch, "High-Performance Image Processing on FPGAs," *Xcell Journal*, vol. 57, no. 2, pp. 42–44, Apr. 2006.

[3] A. do Carmo Lucas, S. Heithecker, P. Ruffer, R. Ernst, H. Ruckert, G. Wischermann, K. Gebel, R. Fach, W. Huther, S. Eichner, and G. Scheller, "A reconfigurable HW/SW platform for computation intensive high-resolution real-time digital film applications," in *Proceedings of DATE'06, Munich, Germany*, Mar. 2006, pp. 1–6.

[4] C. Claus, W. Stechele, and A. Herkersdorf, "Autovision - A Run-time Reconfigurable MPSoC Architecture for Future Driver Assistance Systems," *Information Technology*, vol. 49, no. 3, pp. 181–186, June 2007.

[5] N. Alt, C. Claus, and W. Stechele, "Hardware/software architecture of an algorithm for vision-based real-time vehicle detection in dark environments," in *Proceedings of DATE'08, Munich, Germany*, Mar. 2008, pp. 176–181.

[6] J. Angermeier, U. Batzer, M. Majer, J. Teich, C. Claus, and W. Stechele, "Reconfigurable HW/SW Architecture of a Real-Time Driver Assistance System," in *Proceedings of ARC'08, London, U.K.*, Mar. 2008.

[7] J. Angermeier, D. Göhringer, M. Majer, J. Teich, S. P. Fekete, and J. V. der Veen, "The Erlangen Slot Machine - A Platform for Interdisciplinary Research in Dynamically Reconfigurable Computing," *Information Technology*, vol. 49, pp. 143–149, June 2007.