

Fully Integrated Communication Terminal and Equipment

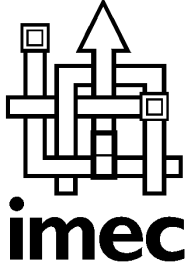
FlexWave II :Executive Summary

Specification : Executive Summary, D36B
Authors : J. Bormans
Document no. : P50314-IM-DL-022
Status : Issue 1
Date : July 2005
ESTEC Contract : 13716/99/NL/FM(SC)

ESTEC Technical Management: R. Weigand

European Space Agency
Contract Report

The work described in this report was done under ESA contract. Responsibility for the contents resides with the authors or organization preparing it.

	FlexWave II: Executive Summary	Do. ref.: P50314-IM- DL-022 Date: 2005/08/01 Issue: 1
Author:	J. Bormans	Page 1 of 11

Contacts:

ESA- ESTEC: R. Weigand – Microelectronics Section TEC-EDM

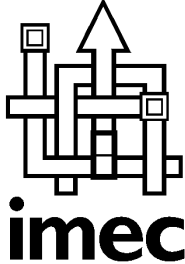
E-Mail: Roland.Weigand[at]esa.int

IMEC : E. Blokken

E-Mail.: Eddy.Blokken[at]imec.be

: J. Roggen

E-Mail: Jean.Roggen[at]imec.be

	FlexWave II: Executive Summary	Do. ref.: P50314-IM-DL-022 Date: 2005/08/01 Issue: 1
Author:	J. Bormans	Page 2 of 11

FlexWave-II overview

The FlexWave-II has been developed as a dedicated image compression component for spaceborne applications, enabling a multitude of application scenarios, including lossless and lossy compression. The FlexWave-II provides scalable compression, allowing gradual enhancement or degradation of the image quality in a programmable way. A wavelet-based compression scheme has been selected because of the intrinsic scalable characteristics. Moreover the compression criteria can be tuned separately for optimal measurement and visual data compression.

The FlexWave-II provides full scalability features and high processing performance. It supports push-broom image processing. The wavelet transform engine is capable of computing up to 5 levels of wavelet transform with 5/3-, 9/3- or 9/7-tap wavelet filters, for image sizes as large as 1k'1k pixels. On an FPGA implementation, clocked on 41 MHz, a processing performance of up to 10 Mpixels/second was measured. The wavelet compression engine allows two compression modes: a fixed compression ratio mode optimised for user-defined criteria and a fixed quantisation mode with user defined quantisation tables.

The FlexWave-II implements a complete wavelet-based image compression engine, including (see Figure 1):

- A programmable, block-based wavelet transform;
- Scaling and thresholding of the wavelet coefficients;
- A DC-coefficients stuffing unit;
- 4 parallel (to achieve high throughput) AC-coefficient compression units containing
 - a Shapiro-based EZT encoder,
 - a 4-symbol arithmetic coder for the dominant symbols,
 - a serial-parallel conversion unit for the subordinate symbols,
 - packet collectors for both the dominant and the subordinate data streams.

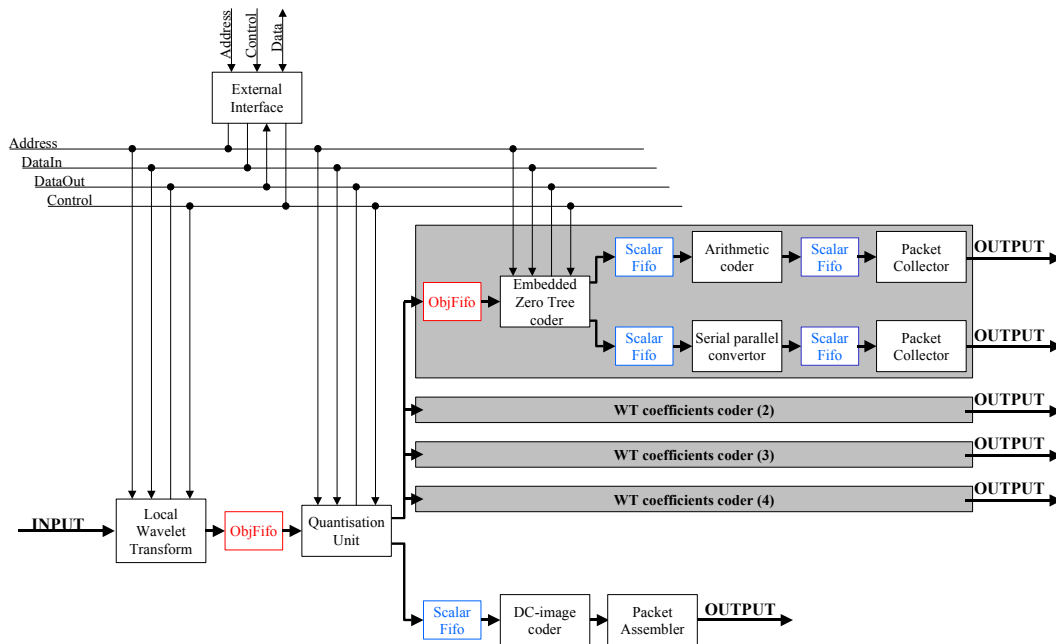


Figure 1: Top-level block diagram of the FlexWave-II

Functional Flexibility

The FlexWave-II was designed with flexibility in mind to be able to deal with a great variety of image compression contexts. The main enabling element is a programmable Local Wavelet Transformation engine (see below) that performs a block-based wavelet computation¹, which is completed by parameterisable functional blocks. The FlexWave-II allows for:

1. A programmable blocksize (4×4 , 8×8 , 16×16 or 32×32)
2. A programmable number of transformations (2, 3, 4 or 5)
3. A programmable wavelet filter (5/3, 9/3 or 9/7) all implemented with the lifting scheme (= lossless transform)
4. Programmable wavelet transform allowing strip-based processing required for push-broom operation.
5. Programmable input pixel accuracy (up to 12 bit)
6. Programmable thresholding and scaling of the wavelet coefficients
7. Programmable quantisation of the wavelet coefficients

¹ Only the computation itself is block-based, the algorithmic behaviour is kept frame-based, i.e. no tiling artefacts are introduced.

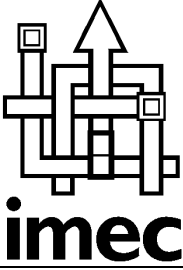
Local Wavelet Transform Engine

FlexWave-II's Local Wavelet Transform (LWT) is a superscalar architecture with dedicated processing modules, interchanging their data along very localised data transfer busses for high-speed processing. The architecture has been tailored to an almost 100% utilisation factor of the DWT filter.

Image Size	Transform	Filter Type	Levels		
			3	4	5
256	LWT	5/3	4.3k	8.3k	19.4k
		9/7	11.1k	20.0k	41.7k
	RC	64k			
1k	LWT	5/3	15k	25k	48k
		9/7	37k	75k	132k
	RC	1M			
4k	LWT	5/3	60k	97k	178k
		9/7	149k	294k	511k
	RC	16M			

Table 1: Memory requirements in function of the image dimensions for the Row Column (RC) and the Local Wavelet Transform (LWT)

It is important to note that this block processing provides functionally/algorithmically identical results as classical wavelet transform techniques, e.g., the Row-Column (RC) wavelet transform. As a consequence, no tiling or blocking artefacts occur, while achieving very significant memory gains, as confirmed by Table 1, comparing the LWT to RC with respect to memory requirements. Because the LWT processes rows of blocks, the memory requirements are only proportional to the image width and are independent of the image height. Consequently, images with an infinite height (= push-broom processing) can be dealt with.

	FlexWave II: Executive Summary	Do. ref.: P50314-IM-DL-022 Date: 2005/08/01 Issue: 1
Author:	J. Bormans	Page 5 of 11

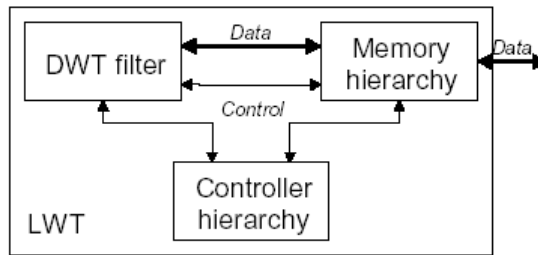


Figure 2: High-Level LWT architecture

The LWT data transfer centric implementation is constructed around three main parts shown in Figure 2: the hierarchical memory architecture, the hierarchical controller architecture and the filter block.

FlexWave-II Implementation Results & Tests

The FlexWave-II is available as a VHDL design kit, including all the necessary test benches and the programming environment for the LWT engine (to generate the appropriate LWT instructions for a given set of parameters).

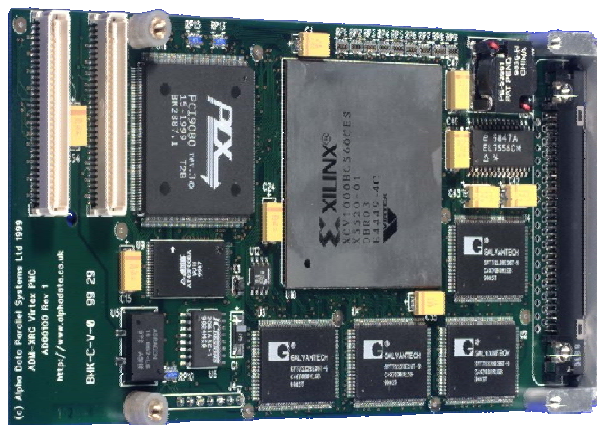
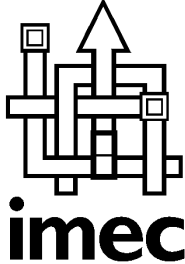


Figure 3: Alpha Data ADM-XRC board

For testing the FlexWave-II implementation an Alpha Data ADM-XRC board has been used (see Figure 3). The features that are present on this board are:

- Xilinx Virtex 2000E FPGA,
- Four external memories of 512K × 36 bits,

	FlexWave II: Executive Summary	Do. ref.: P50314-IM-DL-022 Date: 2005/08/01 Issue: 1
Author:	J. Bormans	Page 6 of 11

- Access to the PCI bus, and
- Two independent and programmable clocks.

Two external memories are used for temporary storage of intermediate results of the LWT block, the two remaining memories are used for data IO between the FPGA and the PC, one containing the input image, the other containing the compressed data.

The mapping results, including the interface logic to the PCI bus and the external memories, are listed in Table 2. Additionally, a study proved the feasibility of an ASIC implementation in the UMC EuroPractice 0.18mm technology.

```

Number of Slices:
    15,906 out of 19,200      82%
Number of Slice Flip Flops:
    11,505 out of 38,400    29%
Total Number 4 input LUTs:
    27,804 out of 38,400    72%
Number used as LUTs:
    26,853
Number used as a route-thru:
    951
Number of Block RAMs:
    128 out of 160          80%
Operating clock frequency:
    41 MHz

```

Table 2: FlexWave-II FPGA implementation report

Table 3 gives an overview of the synthesis results obtained for a clock period of 20 ns (50 MHz clock) and a RAM read delay of 5 ns. Synthesis was performed using worst case operating conditions and estimated RAM delays.

Block	Gate Count	Percentage
LWT	51070	29.2
Quantisation unit	6733	3.9
CoderBlock 0	26890	15.4
CoderBlock 1	26769	15.3
CoderBlock 2	26769	15.3
CoderBlock 3	26769	15.3
DC coder	7508	4.3
Glue	2266	1.3
Total	174776	100

Table 3: Synthesis results for 0.18mm UMC technology. The numbers are obtained for a clock period of 20 ns, the RAM delays were set to 5 ns. The figures listed here are obtained using inaccurate RAM models and do not take layout back annotation into account.

After synthesis, timing analysis has been performed for the available operating conditions (Best Case, Typical and Worst Case). These timing results, summarised in Table 4, were obtained without layout back annotation.

Operating conditions	RAM delay	Critical path	Clock frequency
Best Case	3 ns	8.84 ns	113 MHz
Typical Case	4 ns	11.91 ns	84 MHz
Worst Case	5 ns	19.89 ns	50 MHz

Table 4: Timing analysis results for the available operating conditions.

Figure 4 shows the configuration used for testing the FlexWave-II implementation. The FlexWave-II core together with an interface has been implemented on the board's FPGA. The interface is allowing the software to perform the following actions:

- Access the parameters of the FlexWave-II through the PCI Bus.
- Access to the two of the four memories either from software, through the PCI Bus, or directly from the FlexWave-II. These memories are used as the input and output memories.

The clocks are also controlled from the software and set to the desired frequencies.

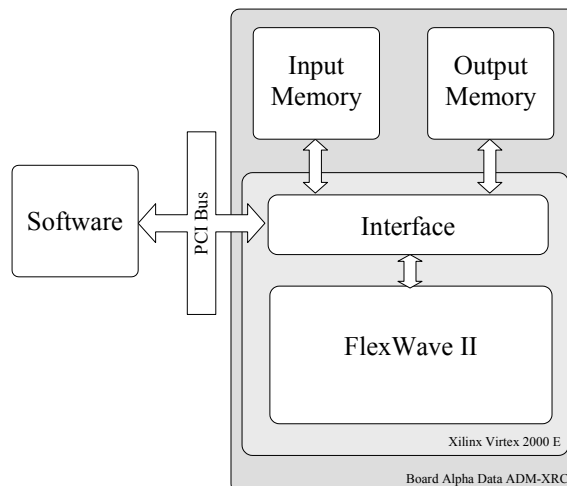
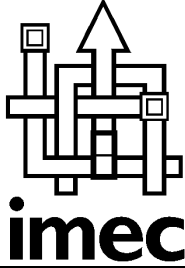


Figure 5 – FlexWave-II Test Configuration

	FlexWave II: Executive Summary	Do. ref.: P50314-IM-DL-022 Date: 2005/08/01 Issue: 1
Author:	J. Bormans	Page 8 of 11

The following procedure for testing is applied in each case:

1. The input image is converted from row wise into block wise (Figure 2).
2. The FPGA is programmed, clocks are set, and the FlexWave-II settings are downloaded.
3. Process Input Image.
 - The blocked image is downloaded into the Input Memory
 - Send to the FlexWave-II component the start signal
 - Wait for the processing to finish. FlexWave-II are written into the Output Memory
 - Retrieve the compressed data and the processing information
4. Decode and Compare processed data.
 - Sort the incoming data from the board.
 - Decode the sorted data.
 - Compare the decoded image with the original image. Both images should be identical.

Figure 6 shows a FlexWave-II demonstration set-up whereby images from a single source are compressed in a scalable way by the FlexWave-II and sent to two terminals with different decoding capabilities. As a consequence, the high-end terminal is able to decode the images at a higher quality than the low-end terminal without the need for the encoding process to be aware of this.

FlexWave-II and CCSDS

IMEC has supported ESA during the standardisation of the Consultative Committee for Space Data Systems (CCSDS) Image Data Compression activity. This initiative aimed at extending the range of available tools to compress images beyond lossless only techniques (such as the Rice algorithm), so as to be suited for a wide range of applications.



FlexWave-II developments have helped convincing CCSDS to adopt a wavelet-based scheme (Figure 7).

Figure 6: FlexWave-II demonstration set-up

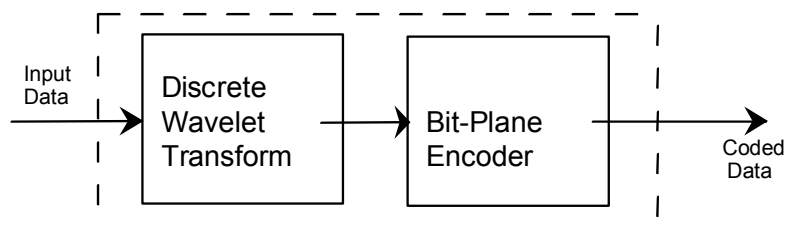
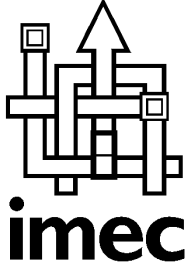


Figure 7: General Schematic of the CCSDS Coder

	FlexWave II: Executive Summary	Do. ref.: P50314-IM- DL-022 Date: 2005/08/01 Issue: 1
Author:	J. Bormans	Page 10 of 11

The entropy coder is not a zero-tree coder as is the case in FlexWave-II, but an NASA originated Bit-Plane Encoder (BPE).

IMEC has tuned the LWT processor so that it can seamlessly interface with the BPE, by producing blocks of 8x8 (or possibly 16x16) wavelet transformed coefficients. The BPE encoder is then able to produce the (scalable) compressed bitstream.