

Flexicamera: Intelligent and Flexible Camera

Team: Universitat Autònoma de Barcelona

Students: A. Portero, O. Feraz, R. Juranteny, M. Monton, J. Tirado

Supervisor: Professor Titular Dr. J. Carrabina

Abstract

This paper presents the design and implementation of a prototype of a digital audio-video wireless camera developed in the framework of the Texas Instruments DSP and Analog Challenge, with a multidisciplinary team of students from the Universitat Autònoma de Barcelona. The system includes many different aspects concerning both system and component design.

In order to decrease throughput requirements for wireless communication, video and audio channels are compressed using MPEG2 format. The camera acquires video from a commercial CMOS sensor with digital output (YUV 4:2:2 format). The image data stream (640x480 pixels maximum with progressive scan) is compressed (producing only I frames) with an FPGA implementation. On the other side, we have a microphone with the analog conditioning circuitry and A/D conversion. This feeds a DSP TMS320C6X those implements MP3 (MPEG2 layer3) compression algorithms. For prototyping we have used a fixed point DSP, though computationally the system requires floating point implementation. These two streams, audio and video, are mixed in a transport layer and sent to a PC through a wireless protocol implemented with an 802.11b protocol PCMCIA card with a maximum throughput of 11Mbps. Finally, the PC will implement the real time MPEG2 decompression.

1. Introduction: ESCam system

ESCam is Electronic Super Camera [1]. The aim of the project is to implement camera that will produce images in electronic format (html) using multimedia standards MPEG with wireless transmission. In following sections we will describe the implementation choices, design strategies and prototyping results of the different subsystems presented in figure 1. Systems Architecture is presented in figure 2.

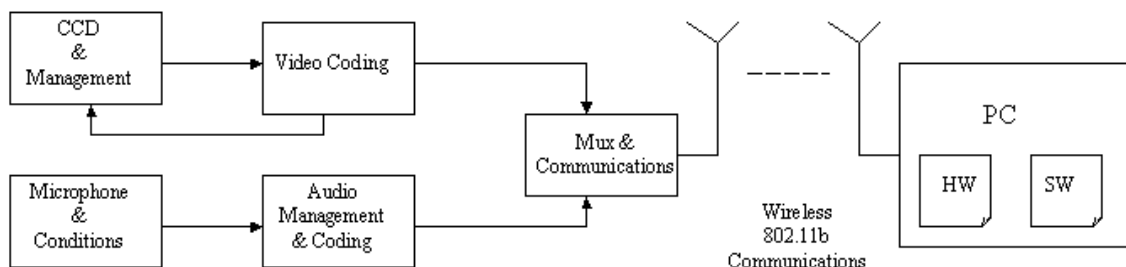


Fig 1. Functional Block Diagram

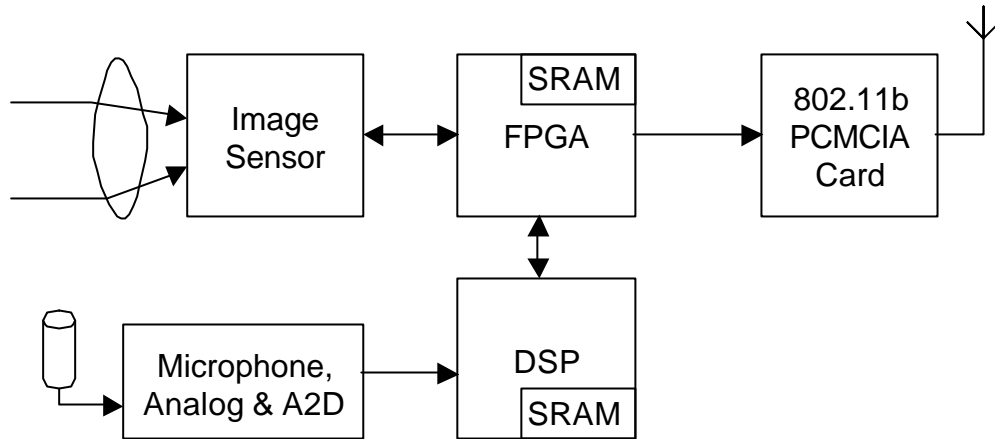


Fig 2. Architectural Block Diagram

2. CCD and Management

The sensor used to implement ESCAM was the OV7620 from Omnivision [2], that permits to program, through a set of configuration registers, several functions such as: gain, windowing, color format, video output format, frames per second etc. The sensor registers configuration is performed through an SCCB bus (I2C-like). The maximum transfer rate is 30 fps with 640x480 pixel images. Color Format that extracts is YUV 4:2:2 although it is possible to program it for obtaining YUV 4:0:0 or RGB 8:8:8.

The configuration and data flow between the microsystem and specific control HW has been tested with the RIC1 prototyping board [3], as shown in figure 3.



Fig 3. RIC (left) and sensor (right) prototyping boards used to prototype image acquisition.

Quality of color conversion has been analyzed with matlab obtaining slightly differences among theoretical and HW simplified RGB2YUV conversion implemented in the sensor [4].

3. Video Coding

The input format coming from the sensor to the compression/coding module is a flow of images of 640x480 pixels using YUV 4:2:2 color code. Data rate is 25 frames per second.

According to the throughput requirements of the system, coming from compression, transport, or transmission, the amount of computation can be slowed down by either changing the image rate or the window size, or simplifying the data format to YUV 4:2:0. The restriction imposed by the compression algorithms say that resizing must be of complete 8*8 pixel windows.

This module implements MPEG2 compression [5,6,7] according to the scheme of figure 4. The structure of the implementation and the complexity of the computation will strongly depend on the following parameters:

- GOP structure: Intra-code (I), Predicted (P), Bidirectional (B).
- Size: range from 1920x1152 to 352x288
- YUV structure: 4:2:2, 4:2:0

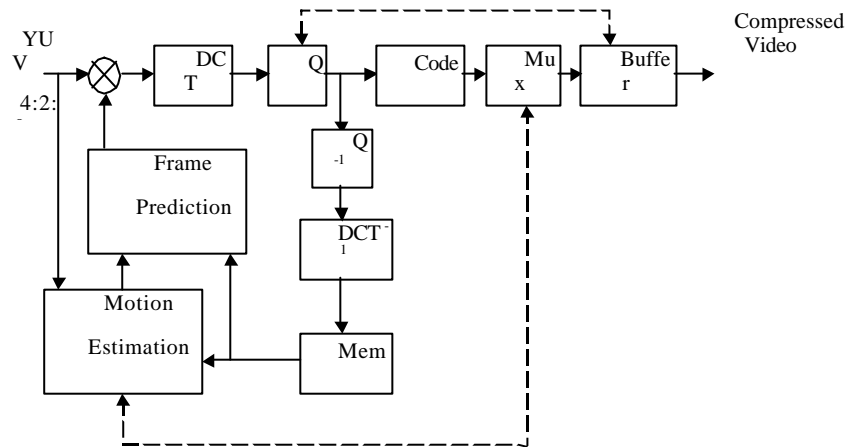


Fig. 4. Bloc diagram of general MPEG

In order to reduce the complexity of our implementation (figure 5), that can be furthermore completed; we have selected to implement the following choices:

- Implement compression producing only I-frames. This allows skipping motion detection algorithms that are rather complex according to the search criteria implemented and area scanned. On the other side, this will increase the bit rate depending on the application. This choice is structurally equivalent to implement JPEG compression.
- Limit the size to the max allowed by our sensor. 640x480.
- Limit the implementation to the usual YUV 4:2:0 standard color codes.

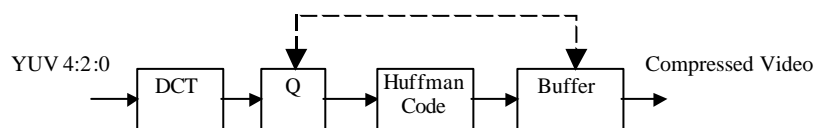


Fig. 5. Bloc diagram of MPEG implemented

In this way the complexity of the implementation is reduced to the implementation of the Direct Cosinus Transform (DCT), quantization (Q) and coding (CODE) modules. Buffer management can be globally considered.

Direct Cosinus Transform Bloc

The computed complexity of the system is estimated as: 640*480 per image, so 4800 blocks of 8x8 pixels (MPEG definition), what means to implement 120000DCT/seg.

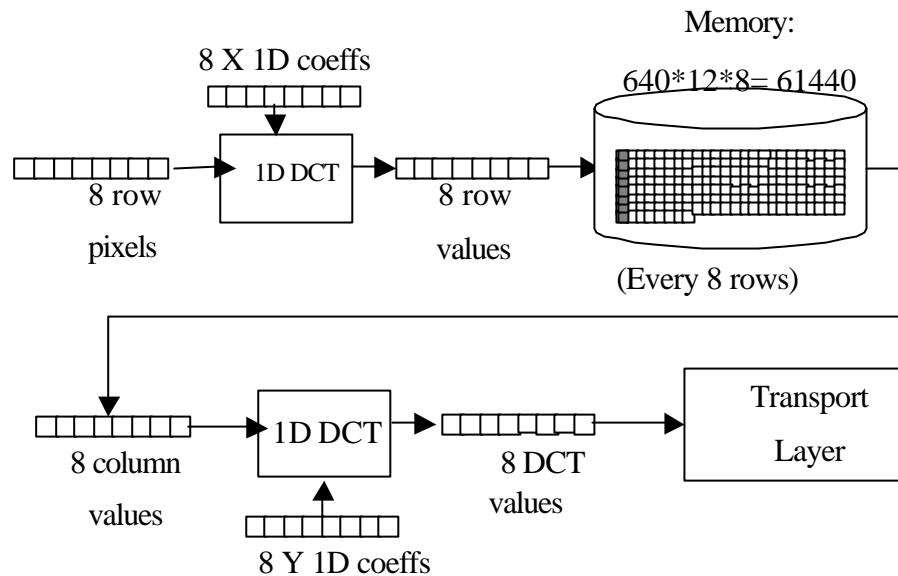


Fig. 6. Data flow for implementation of the DCT

In order to implement the bidimensional DCT transform, we will use an FPGA. This allows us to apply specialized architectures for managing data, and also to implement efficient simplifications in terms of data width (that will affect area costs) and multiplier –accumulate (MAC) implementation with fixed coefficients (that allow to simplify resources). Image data flow using intermediate buffering is shown in figure 6. The implementation of the unidimensional DCT will use a parallel scheme, shown in figure 7, compatible with the 1 pixel per cycle data-flow throughput. MAC units will be synthesized according to the previously mentioned parameterization criteria (bus width & fixed coefficient simplification).

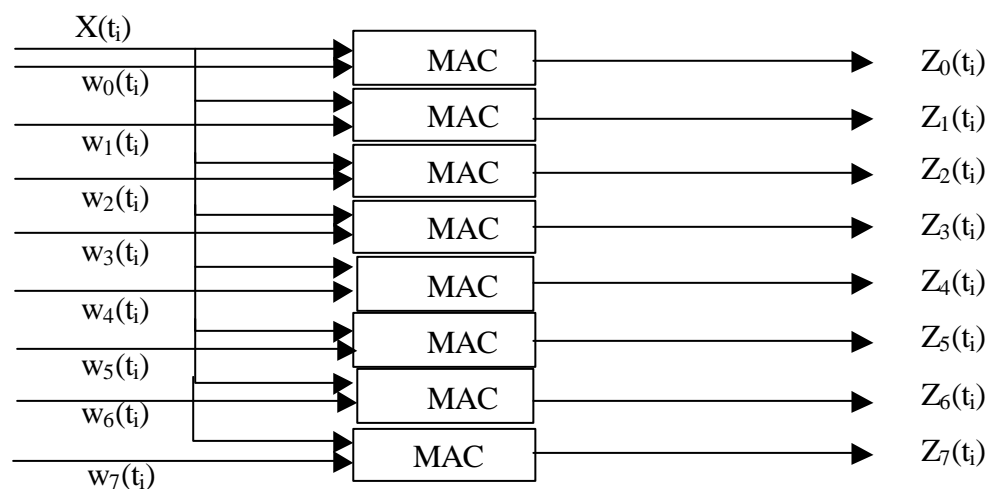


Fig. 7. Bloc diagram of 1D DCT implemented

Simplifications concerning computations of the MPEG compression can be analyzed according to the resolution of the pixels acquired by the image sensor (1/2 LSB on 8 bit pixels). A basic worst-case analysis according to the DCT fixed coefficients for the transform matrix gives us the error level (over 256) summarized in table 1.

<i>N</i>	<i>Error</i>
2	217.309685
4	12.5991416
5	2.41316659
6	0.59105041
7	0.24089893
8	0.0582276

Table 1. Maximum error versus coefficient word length

From this table, we can conclude that coefficients with more than 5 bits can be used for the implementation of the MAC units.

Therefore, we can estimate the upper value for the cost of the MAC units from the estimations of the macro cells implemented by the Altera tools. The area cost from all the 16 MAC cells and the delay per MAC (in parallel), shown in table 2, allows this design to be prototyped into the RIC board.

FLEX10K		MAC cost	
Mult	Add	Area (gate)	Speed (<i>ns</i>)
5 x 8	13	22080	88
6 x 8	14	26880	98
7 x 8	15	30912	103
8 x 8	16	35328	107

Table 2. MAC area and delay costs

Quantization & coding

The process of lossy compression is implemented in the JPEG/MPEG standard in the quantization of the frequency coefficients obtained by the DCT transform. This process takes into account the human visual perception.

In our case this lossy process allows us to implement the previously presented error analysis for computational simplification.

Quantization is implemented dividing the values obtained by the DCT transform by a fixed set of coefficients, allowing an important simplification in the implementation of the divider.

MPEG coding uses Huffman type Variable Length Coder that will be implemented using a tabular method. Coefficients of this coding are fixed by the standard according to statistical results, in such a way that the dictionary does not need to be sent with the image data. The availability of internal memory inside the FPGA allows this implementation with negligible cost.

4. Microphone and Conditioning

This part takes charge of all analogue and A/D conversion circuitry required to transform the acoustic signal to digital data that the audio compressor DSP can process. The structure of the circuit is shown in figure 8 and its prototype implementation in figure 9.

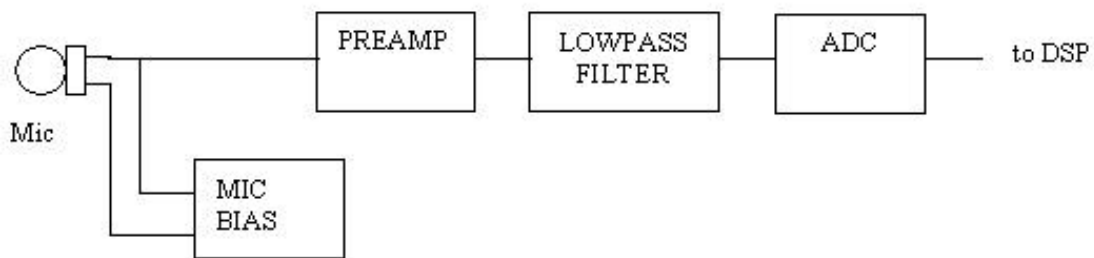


Fig 8. Block diagram of the Audio acquisition circuitry

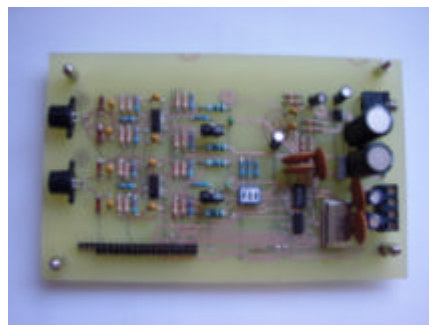


Fig 9. Photography of the prototype

The microphones are two electret-type (the natural choice for low-cost, small-size, little-voltage applications) Panasonic WM-61, with a -35 ± 4 dB re 1V/Pa sensitivity, which corresponds to a gain between 11 and 28mV/Pa, in a bandwidth from 20Hz to 20Khz ($3 \pm$ dB), following the AC'97 standard [9] for PC sound systems.

This standard also specifies a resolution of 16 bits, which means a signal-to-noise ratio $SNR > 96$ dB. The whole design is tuned to achieve such a SNR, with an exception: the only dark point of electret microphones is that they introduce some noise, so the general performance of the audio system is degraded by the $SNR = 62$ dB of the microphones, which means a resolution of just 11 bits, resulting in some perceptible noise. Nevertheless, performance can be enhanced just by switching to professional capacitor microphones.

The next task is biasing the microphones and amplifying the weak signal from them; this is done with Texas Instruments opamps TLV2465 [8], using differential signaling to avoid interferences. For a maximum allowable sound level of 94 dBA=1 Pa, microphones will

deliver an output between 11 and 28 mV_{rms}; considering that (for the sake of low-power) the circuit will be powered at 3.3V, a good gain which would fill this range would be $G=25$.

Filtering must be done to avoid the alias effect in the A/D converter. The analog to digital conversion is performed by a sigma delta converter (TLC320AD58). It produces a continuous data flow with 48ksamples per second, but fortunately it's a sigma-delta converter, so its input sampling rate is much higher, of 6,14 MHz. So, the filter has to pass frequencies below 20 kHz and stop frequencies above 3 MHz; this can be done by just a basic RC circuit with a cut-off frequency of 160KHz.

Stereo samples (containing 16 bits for the right channel and 16 for the left one) are sent by the A/D to the TI DSP through a TDM McBSP port; this communication is governed by a 12.288 MHz oscillator, from which the DSP derives the bit and word clocks.

5. Audio Compression and Coding

The ESCam audio module has a main purpose to compress digital audio data in RAW format to a MP3 standard. This RAW format consists on a continuous data flow with 48k samples per second. Every sample has 16 bits from right Audio channel and 16 from the left channel according to the AC'97 standard. The output format is a MP3 bit stream, totally compatible with all decompressors that are in the market such as: Winamp, Real Jukebox, XMMS, etc. MP3 stands for MPEG1 & MPEG2 layer 3.

Our initial implementation idea, opposite to the choice selected for the image compression, has been to map the algorithms into a digital signal processor. We dispose of the C code from the open source coming from the LAME project [10].

Although floating point arithmetic is specified in the standards, we have mapped this code to the fixed point TMS320C62 DSP from TI. In this way, we can test the code into a PC-based prototyping board TMDS3260A6201'C62X (see figure 10), using the tools coming from TI. Alternative simulations on fixed point TMS320C67 DSPs have been tried with very few success due to the length of simulation time required for the MP3 compression. Later on, implementation will either used fixed point computation (clipping compression coefficients as done for images) or require floating point processing, the code can be directly mapped to a TMS320C67 DSP since they are code compatible.



Fig. 10. TI 'C62 DSP Prototyping board

The choice of this DSP is due to the high performance required for the application. This DSP contains 8 functional units including two multipliers and two adders, though the

compiler does not seem to be able to use them. Complexity analysis can be summarized by the following table 3 that indicates the profiling of usage of different modules, that we have used in order to optimize performance.

Function	%Time
_mdct_long (newmdct.obj)	6,0
_window_subband (newmdct.obj)	51,0
_count_bit_noESC (takehiro.obj)	2,4
_count_bits (takehiro.obj)	2,4
_quantize_xrpow_ISO (quantize-pvt.obj)	1,0
__alloca_probe (chkstk.obj)	2,5
__chkstk (chkstk.obj)	2,5
_calc_xmin (quantize-pvt.obj)	4,2
_iteration_loop (quantize.obj)	2,9
_mdct_sub48 (newmdct.obj)	2,1
_lame_encode_buffer_interleaved (lame.obj)	2,0
_mdct_init48 (newmdct.obj)	3,3

Table 3. Temporal usage of main modules

Main conclusion on this table is that temporal window processing (applying Hamming windows on audio frames) and memory management overhead (strongly dependent on the implementation in the prototyping board) should be optimized manually in order to improve computational requirements.

We have measured on different pieces of compiled code (.asm files) the number of instructions that contain concurrent execution of functional units. The following table 4 summarizes these values respect to the number n of concurrent units, nC . One can verify that the compiler tools never achieve the use more that 6 units in parallel, and the case of 5 and 6 are quite unusual. This is important considering that the algorithms seem to be highly parallel.

Files (.asm)	#Inst.	2C	3C	4C	5C	6C
FFT	3918	214	52	11	3	
Main	386	12	1	1		
Lame	5003	317	71	23	11	2
Newdct	5159	529	91	30	12	2
Psymodel	6395	507	94	27	3	1
Quantize	4030	274	77	25	16	4
Reservoir	213	19	1			
Takehiro	1572	99	51	7	2	
Util	802	42	25	3	4	
Vbrquantize	1756	123	32	1	3	1
Vbrtag	1102	66	13	10		1
Versión	127	6	10			
Formatbit	1426	54	9	3	1	1
l3bits	2476	118	16	10	1	
quantize-pvm	2862	221	69	13	5	

Table 4. Maximum error versus coefficient word length

The original C code was developed to work with concurrent functional units so aspects that can be optimized by the compilers like loop unrolling, etc. have been taken into account.

In order to map this C code to our DSP, we have adapted the input/output structures, fixed the programmable parameters of the code (filter coefficients, quality of compression and data rate), disabled variable bit rate, and fixed some bugs produced by our compiler (dynamic memory allocation, etc). The final C code has 11200 lines, and the coff object code occupies 855Kbytes of memory, including input/output libraries from the emulation system that later on will be out of the code.

6. Mux & Communications

This part mixes unidirectional signals coming from MP3 audio frames and MPEG-2 Video frames, and bidirectional signals for the identification and control of different subsystems (figure 11).

This module implements an MPEG Transport Stream that feeds data to the wireless interface to the PC server. This transport stream was initially placed to a microcontroller, but the high number of ports needed concurrently, and the high throughput required, let us to place this module into the FPGA.

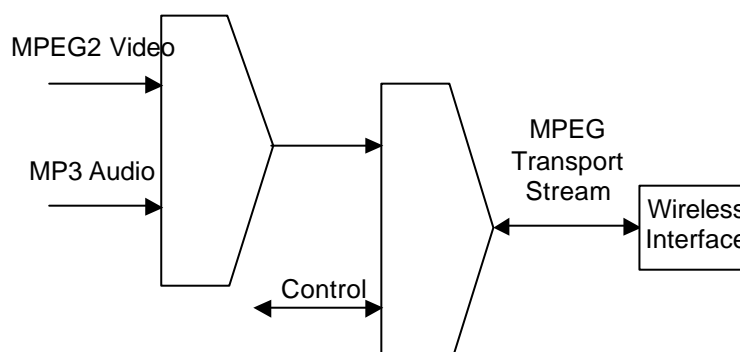


Fig 11. Multiplex and Communications Paths

7. Wireless Protocol

One of the most attractive aspects of our system is its wireless operation capabilities, that allow to place the cameras far away from a central server, what is crucial for many applications like security, home systems, automated plants, visual inspection, etc.

Among the different wireless possibilities we selected implementations of the standard 802.11b since they are the widest available in the market for our range of applications: 11Mbps and 200m rang. IEEE802.11 standard uses Orthogonal Frequency Division Multiplexing (OFDM), targeting a range of data rates from 6 to 54Mbps. Bluetooth was also initially considered but narrowed bandwidth and unavailability of products helped us to discard this option.

The goal of our system is to get high degree of portability and to allow rapid development of different specific applications. So, we have based the wireless communication of our system in the standard socket, PCMCIA [11].

This socket will connect the card that implements the protocol. This allows adapting our system through other protocols (USB, Ethernet, Hyperlan2) by using the corresponding PCMCIA cards. The application performance and cost will be the driving criteria.

From the range of actual wireless products, we decided to use a COMPAQ WL-100[12] PCMCIA card that implements the 802.11b protocol. It allows 11Mbps rate (see Table 5) that seems to be the maximum actual rate for a wireless LAN and enough for our application range. We have verified this capacity in a point-to-point communication between a laptop and a desktop. Figure 12, shows the PCMCIA card and also the PC containing the base station card (implemented as a PCI card) with the corresponding antenna.

Tax Rate	Code length	Technique	Modulation
1Mbps	11(secuencia Barker)	FHSS/DSSS	BPSK
2Mbps	11(secuencia Barker)	FHSS/DSSS	QPSK
5.5Mbps	8(CCK)	DSSS	QPSK
11Mbps	8(CCK)	DSSS	QPSK

Table 5. Tax rate especificationss for standard 802.11b.

Another important factor is the availability of a wide documentation on the internal chipset, PRISM II from Intersil [13]. We dispose also of an open source driver for Linux [14].

Our initial intention have been to implement a direct interface between the Intersil's chipset PRISM2. It is used for digital transmissions using the standard 802.11b that allow 11Mbps tax rates.

Untill now, we have implemented a system that allow to connect the WL100 wireless card with an FPGA. The FPGA must communicate with the Intersil's interface chip Hfa3841. For the implementation of VHDL read and write modules to this chip, we have used its specific communication protocol. To know which are this PRISM2 commands and how to issue to the Hfa3841, we have analyzed the free codes of the C drivers that Absolute Value Systems and Samsung have developed for WL100 wireless card in LINUX platform [15,16]. Starting from the functions and structures of the drivers, we implemented the equivalent VHDL modules. That is to say, we have a hierarchical structure of VHDL modules that will reproduce all the C functions that configure the MAC layer. We expect to develop a good VHDL driver for this wireless card that allow to transfer in AdHoc mode, frames that we receive from the camera.

Moreover, this chip is available in a reasonable availability and price so, we can still foresee a single board camera implementation. Several vendors use this chipset in the systems such as: Nokia, Samsung, 3COM, LynkSys and many others.

Finally, PCMCIA solution is easier that direct chip communication, in order to handle and to improve our system when a higher data rate card will be available in the market. It is only necessary to unplug and plug in the new one. It is clear that new drivers are required.



Fig 12. Wireless subsystem components

Conclusions

This work presents the design of an “intelligent” and flexible” camera, using subsystem specific resources to prototype local data flow and control.

Intelligence is achieved by distributing the control in such a way that an optimal performance between throughput and image and audio quality can be obtained by tuning parameters from the different subsystems. Using a global criterion instead of the existing local ones does this.

The use of standard protocols allows an optimal partition of the design, and a high degree of flexibility, that is improved using package compatible connectors for the critical aspects line FPGAs for the transport layer and PCMCIA cards for communication to the host.

Though this work focuses in system development, we are also using it as an example to model complex HW/SW systems using Codesign Methodologies, virtual components and rapid prototyping platforms.

References

- [1] microelec.uab.es/~escam
- [2] Omnivision www.ovt.com
- [3] F.Lisa, A.Portero, J.Carrabina”Design sample applications for R.I.C. Configurable Platform”. Proc. XV DCIS Montpellier, Francia. Nov. 2000
- [4] R. Juvanteny “Interfície amb sensors òptics per a càmeres intel·ligents”. Eng. Degree project. UAB 2001.

- [5] ISO/IEC JTC1/SC29/WG11 N0702 Rev, "Information Technology – Generic Coding of Moving Pictures and Associated Audio, Recommendation H.262", Draft International Standard, Paris, 25 March 1994..
- [6] Tektronix "A Guide to MPEG Fundamentals and Protocol Analysis (Including DVB and ATSC). September 1997. <http://www.tek.com>
- [7] T.Sikora. "MPEG-1 and MPEG-2 Digital Video Coding Standards" in "Digital Consumer Electronics Handbook " McGraw-Hill .
- [8] <http://www.ti.com>
- [9] INTEL. Audio Codec'97. Revision 2.1. Intel Corporation, 1998.
- [10] LAME project. <http://mp3dev.org>
- [11] <http://www.pcmcia.org>
- [12] www.Compaq.com
- [13] <http://www.intersil.com>
- [14] [http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Linux Wireless Drivers](http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Linux%20Wireless%20Drivers)
- [15] linux-wlan-ng-0.1.7. Driver from Absolute Value Systems located in <http://www.linux-wlan.com>.
- [16] swldpc11_cs.c. Driver from Samsung located in <http://www.MagicLan.com>