FEUP Universidade do Porto
Faculdade de Engenharia

Licenciatura em Engenharia Electrotécnica e de Computadores

Projecto, Seminário ou Trabalho Final de Curso

Ano lectivo 2005/2006

CÂMARA DE VIDEO VIGILÂNCIA

Financiamento
União Europeia
FCT
Governo da República Portuguesa
Ciência.Inovação 2010

Licenciatura em Engenharia Electrotécnica e de Computadores

Projecto, Seminário ou Trabalho Final de Curso

Ano lectivo 2005/2006

CÂMARA DE VIDEO VIGILÂNCIA

Orientador:
Prof. João Canas Ferreira

Autoras:
Carla Manuela Neves
Gisela Mota e Costa

Página disponível em:
http://gnomo.fe.up.pt/~ee01162
ou http://gnomo.fe.up.pt/~ee01211

1 Agradecimentos

Gostávamos de agradecer o financiamento do programa POCI2010.

Gostávamos de agradecer ao nosso orientador Professor João Canas Ferreira que nos animou nos momentos mais difíceis e nos incentivou a ultrapassar alguns dos obstáculos encontrados na realização deste projecto.

Gostávamos de agradecer ao nosso colega Miguel L. Silva pela sua paciência e disponibilidade para esclarecer as nossas dúvidas, sem a sua orientação não teríamos conseguido alcançar alguns dos nossos objectivos.

Gostávamos de agradecer ao Professor José Carlos Alves que apesar da sua chegada tardia sempre se mostrou interessado pelo trabalho desenvolvido.

Gostávamos de agradecer ao Engenheiro António Almeida e ao Engenheiro Paulo Magalhães, da Mactek, pelo apoio prestado durante a realização deste projecto.

Gostava de agradecer aos meus pais e avó, em especial ao meu pai que sempre me apoiou nos momentos mais difíceis e que acreditou sempre em mim. Queria também agradecer ao meu marido, pelo apoio, dedicação e ajuda que sempre demonstrou e pela paciência que teve comigo, sem ele não teria chegado até aqui. Ass. Carla Neves.

Aproveito estas linhas para agradecer à família e amigos que me permitiram estabilidade e tranquilidade para atingir todos os meus objectivos. Agradeço também aos docentes e a todos os colegas que fizeram destes cinco anos de curso, a experiência mais enriquecedora e produtiva até ao momento. Ass. Gisela Costa.

2 Índice

3 Índice de ilustrações, tabelas e fluxogramas

4 Lista de abreviaturas e símbolos

ISE	Integrated Software Environment
EDK	Embedded Development Kit
XPS	Xilinx Platform Studio
VHDL	Verilog Hardware Development Language
IP	Intelectual Property
Bmp	Bitmap File
XMD	Xilinx Microprocessor Debugger
GCC	GNU compiler
IP	Internet Protocol
TCP	Transmission Control Protocol
HTTP	Hyper Text Transfer Protocol
SCCB	Serial Camera Control Bus
PROM	Programmable ROM
BRAM	Block RAM
DDR-SDRAM	Dual Port Synchronous RAM
DCT	Discrete Cosine Transform
SDA	Serial Data

SCLK	Serial Clock
GIF	Graphics Interchange Format
PNG	Portable Network Graphics
JPEG	Joint Photographic Expert Group

Tabela 1 - Abreviaturas e símbolos.

5 Sumário

Este sistema associa um módulo de uma câmara com saída digital a um kit Virtex-4, aqui é feita a aquisição da imagem transmitida pela câmara, a sua codificação em JPEG e a sua apresentação numa página de Internet, que é visualizada apenas ligando o computador à FPGA através de um cabo de rede e abrindo um "browser" na página no endereço do Servidor de Internet implementado na FPGA.

O sistema final permite visualizar uma imagem de vídeo de vigilância, para que desta forma se efectue a detecção de movimento numa área abrangida pela câmara.

O desenvolvimento deste projecto implicou um elevado grau de dedicação e muito empenho para que fossem atingidos os requisitos mínimos do sistema que foram definidos no inicio do projecto. A implementação de sistemas em hardware demonstrou ser um desafio constante para se ultrapassarem as contrariedades que surgem. No entanto a satisfação de ver o trabalho a produzir resultados que não são apenas teóricos mas também visíveis compensa toda a ansiedade sentida nos momentos em que parece não existir soluções.

6 Summary

This system associates a digital output camera module to a Virtex-4 kit. Here the image transmitted by the camera is captured, encoded in JPEG format and presented on a Web page, witch is visualised only requiring the connection to a computer with an Ethernet cable and opening a browser on the web page address of the internet web server implemented on the FPGA.

The final system allows visualising a video surveillance image, so it can detect de movement on a specific area covered by the camera.

The development of this project implied a high level of commitment and a hard work so that the minimal requirements of the system that were defined on the beginning of the project were to be achieved. The implementation of hardware systems proved to be a constant challenge to overcome all the difficulties that appeared. However the satisfaction to see a working project with results that aren't just theory but also visible compensates all the anxiety felt on the hardest moments when appears to be no solutions.

7 Introdução

Neste projecto o principal objectivo foi desenvolver um Sistema de Vídeo Vigilância, recorrendo ao uso de Sistemas Reconfiguráveis, uma câmara CMOS com saída digital e um PC com placa de rede. Devido à grande dimensão e complexidade do projecto os objectivos a atingir foram divididos por fases, pois para resolver um problema de grandes dimensões a única solução é dividi-lo em partes de menor dimensão e resolver passo a passo cada uma das importantes etapas deste projecto.

A implementação do código de aquisição dos dados da câmara é considerada a etapa mais importante deste projecto, uma vez que este código é o ponto de partida para qualquer implementação futura que trabalhe com os dados enviados pela câmara. Para atingir este objectivo é necessário confirmar que está a ser enviada uma imagem sincronizada, para este efeito considera-se de vital importância o desenvolvimento de uma interface gráfica capaz de receber os dados recebidos na porta série do PC. Para além de efectuar a recepção dos dados é necessário fazer a apresentação dos mesmos de uma forma perceptível, para tal deve criado um ficheiro bmp. Deve ser criado um ficheiro de mapa de bits uma vez que estes ficheiros possibilitam a escrita de todos os dados após o cabeçalho, de uma forma progressiva.

Após concluída a fase de hardware pretende-se implementar um servidor de Internet, usando o protocolo HTTP1.1. A principal razão que nos leva a querer enviar os dados via cabo de rede é aumentar a taxa de transmissão de imagens para desta forma atingir a transmissão de imagens em tempo real. Outro factor importante de querer apresentar os dados numa página de Internet é o facto de podermos usar um "router" e ter vários clientes ligados a este servidor.

A configuração do Kit Virtex-4 final deverá ser efectuada programando a PROM, desta forma o sistema poderá ser usado por qualquer utilizador sem ser reprogramada.

8 Material utilizado

A câmara digital escolhida para efectuar a aquisição da imagem foi a câmara C3188A em módulo com sensor de imagem CMOS OV7620 da OmniVision®. Apesar de ser um módulo de 1/3" a cores, optou-se por se fazer a sua configuração para o modo de funcionamento a preto e branco. Esta opção foi devida ao facto de se considerar a imagem a preto e branco suficiente para o desenvolvimento de um sistema de Vídeo Vigilância, além de que esta opção facilitará o processo de aquisição, codificação e transmissão da imagem.

Optou-se pela tecnologia do Kit Virtex-4 da Xilinx® para efectuar a aquisição dos dados transmitidos pela câmara e posterior processamento dos mesmos. A transmissão das imagens é efectuada através de uma ligação de cabo de rede, usando o protocolo TCP/IP.

8.1 Software utilizado

- Windows XP® "home edition";
- Xilinx® ISE 8.1.03i, com o "service pack" 3;
- Xilinx® EDK 8.1.02i, com o "service pack" 2;
- ModelSim XE III 6.0d "starter edition";
- Macromedia® Dreamweaver® 8;
- OrCAD® 10.5;
- Borland Delphi® 7;
- Avnet LX25/LX60/SX35 Evaluation Board código de servidor de Internet com grandes alterações feitas ao longo do projecto;
- SerialWatcher;
- Ethereal.

8.2 Hardware utilizado

- Computador com porta série, porta paralela e placa de rede;
- Kit Virtex-4 XC4VLX60 da Avnet com AvBus breakout module;
- Câmara digital C3188A em módulo com sensor de imagem CMOS OV7620 da OmniVision®;
- Cabo RS-232;
- Cabo rede cruzado;
- Cabo JTAG.

8.3 Preparação básica do kit

- Instalar "jumpers" em:
  - J20, pinos 2 e 3;
  - JP18, pinos 3 e 4 (M1);
  - JP10;
  - JP23;
  - JP3;
  - JP12;
  - JP13, pinos 1 e 2;
  - JP2, pinos 1 e 2;
  - JP16 (o teste da DDR pode falhar se este "jumper" não estiver instalado);
- Ligar o cabo JTAG do conector JP17 à porta paralela do PC;
- Ligar o cabo de rede do conector J2 ao PC.

9 Considerações sobre o material utilizado

Nesta secção são feitas considerações acerca do material utilizado, qual a sua função e objectivos, assim como outras informações consideradas relevantes.

9.1 Software utilizado

9.1.1 ISE 8.1.03i

Utilizaram-se as versões mais recentes do ISE e do EDK da Xilinx®, não só por serem mais eficientes mas também por serem a versão na qual o código de servidor de Internet da Avnet® foi desenvolvido. Uma vez que estas versões apresentam diferenças significativas em relação às anteriores (EDK 7.1i e ISE 7.1i), foi necessária uma prévia familiarização e estudo aprofundado destes softwares. Estas ferramentas de desenvolvimento e implementação de projectos são produzidas pela Xilinx®, também fabricante da FPGA utilizada neste trabalho.

A ferramenta ISE integra um Project Navigator de modo a organizar a hierarquia do projecto, e a englobar todos os ficheiros utilizados, sendo possível incluir no mesmo projecto ficheiros na linguagem Verilog e ficheiros em VHDL. Inclui também um sintetizador lógico XST, que permite definir "constraints" para o circuito, para além de ferramentas de "translate", "map", "place & route" e para a geração do "bitmap" propriamente dito.
Para além destes componentes indispensáveis ao fluxo de projecto, o ISE inclui ainda softwares que realizam funções diversas tais como a programação efectiva dos dispositivos através de JTAG ou outras interfaces previstas (iMPACT). É disponibilizado também um IP Core Generator que permite gerar código parameterizado de soluções lógicas já desenvolvidas pela Xilinx® para os seus dispositivos, tais como FIFOS, memórias RAM, DCTs e processadores em Soft Core como é o caso do MicroBlaze.

9.1.2 EDK 8.1.02i

Para trabalhar com este software é necessário o ISE para implementar os projectos gerados na

ferramenta XPS do EDK. As ferramentas disponibilizadas pelo software Xilinx® EDK são: XPS, XMD, GCC, Libgen, Simgen e Platgen.

Os "drivers" deste componente têm uma arquitectura em camada, esta arquitectura engloba os múltiplos usos dos "drivers" dos componentes, que proporciona uma máxima portabilidade entre sistemas, ferramentas e processadores. A implementação dos "drivers" é em código "ANCI C", para aumentar esta característica de portabilidade entre processadores e ferramentas de desenvolvimento. Uma vez que as FPGA podem ser programadas para várias funcionalidades, a configuração dos "drivers" do componente suporta tempos de compilação e execução que podem ser diferentes, estas características providenciam a flexibilidade necessária no desenvolvimento de sistemas reconfiguráveis dinâmicos. Estes "drivers" suportam também múltiplas instanciações do mesmo componente sem efectuar a duplicação do código para cada uma delas, mas no entanto trata as características únicas de cada instanciação.

Este software permite a criação de um OPB IPIF, este módulo é capaz de implementar soluções lógicas (IP) em componentes periféricos integrados. O OPB IPIF está posicionado entre o OPB e o IP como se pode confirmar na (Ilustração 1).

O OPB é um elemento da arquitectura IBM CoreConnect, este barramento síncrono de fácil implementação em componentes periféricos integrados, permite a comunicação destes com o MicroBlaze.

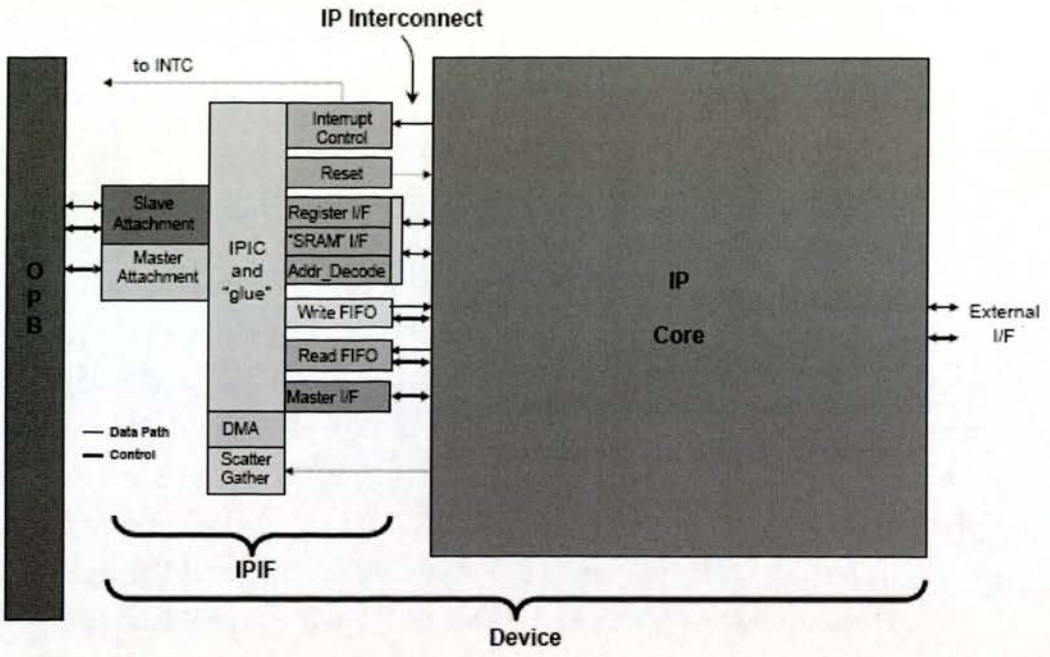


Ilustração 1- Componente OPB com todas as funcionalidades do IPIF

9.1.3 ModelSim XE III 6.0d

Para a realização de testes de toda a implementação de hardware, foi utilizada a ferramenta ModelSim XE III 6.0d "starter edition" da Mentor Graphics® fornecida pela Xilinx®. Este simulador inclui informação temporal no que diz respeito a atrasos provocados pelas diferentes camadas de lógica existente nas FPGAs da Xilinx®, sendo assim muito útil para as simulações após síntese, após "map", e após "place & route".

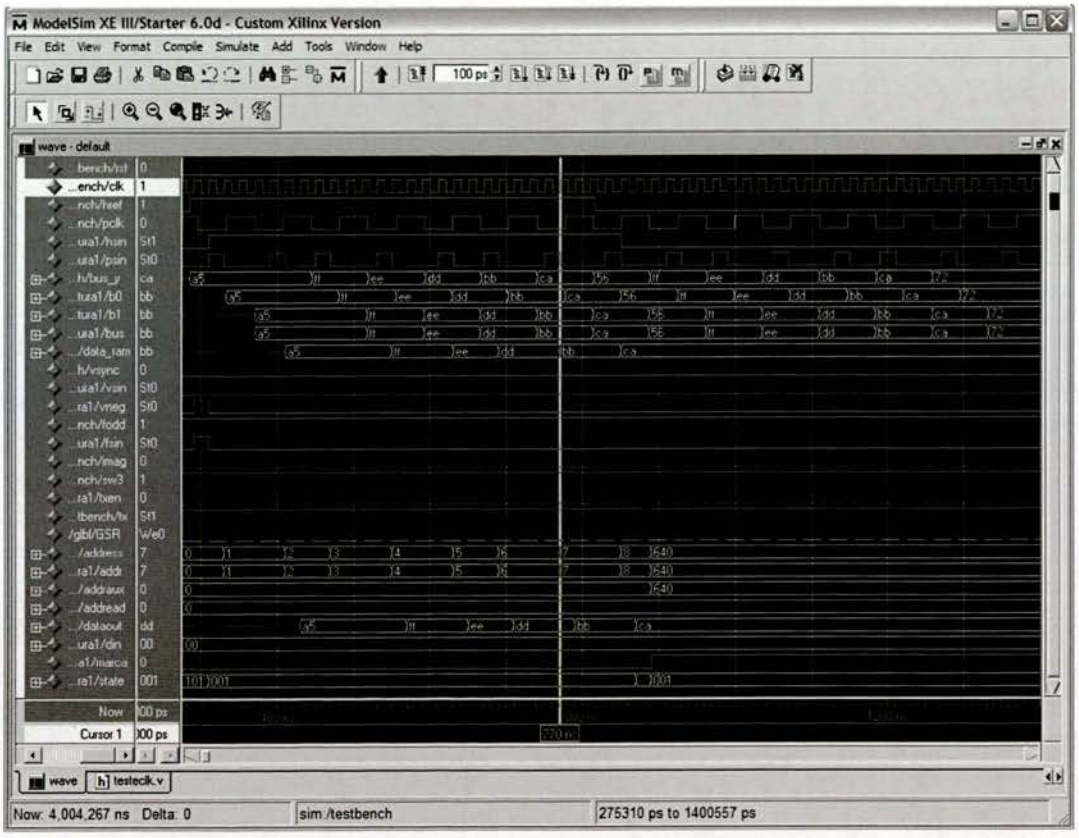


Ilustração 2 - Modelsim Wave.

9.1.4 Macromedia® Dreamweaver® 8

Este programa foi utilizado para produzir a página, em html, do projecto. Macromedia® Dreamweaver® 8 é um programa profissional para a construção de páginas de Internet e aplicações. Fornece uma combinação poderosa de instrumentos de "layout" visual, aplicações de desenvolvimento, e suporte de edição de código permitindo aos programadores e projectistas de vários níveis a criação rápida de aplicações e de páginas Internet visualmente atractivas e de acordo com as normas. Dreamweaver® fornece ferramentas profissionais necessárias para criar um ambiente integrado, moderno e aerodinâmico suportando desde projectos baseados em CSS até programação livre. Permite ao programador a escolha do Internet browser da sua preferência, a implementação de aplicações de Internet em que utilizadores acedem a uma base de dados e serviços de Internet.

9.1.5 OrCAD®

Este software da Cadence Design Systems® é composto por diversas ferramentas, o Capture CIS, o Layout e o Gerbtool estão entre as mais importantes. O Capture CIS é utilizado para desenhar o esquemático e criar uma lista de ligações (NETLIST) com as características físicas dos componentes. O Layout trata dos "templates" e organização das camadas, da importação da lista de ligações, da colocação dos componentes e do tratamento das pistas (Nets). O programa Gerber Tool é utilizado para preparar os ficheiros ou o fotolito para enviar para o fabricante de placas de circuito impresso.

9.1.6 Borland Delphi® 7

Este software permite desenvolver base de dados de cliente/servidor, servidores de Internet e componentes feitos à medida das necessidades do utilizador. Permite também desenvolver aplicações, tais com: SOAP, TCP/IP, COM+ e ActiveX. Utilizou-se a versão 7 deste software pois muitas das funcionalidades que suportam desenvolvimento na Internet, tecnologia XML e desenvolvimento de base de dados, necessitam de componentes e assistentes integrados no software que não estão disponíveis nas versões anteriores.

9.1.7 Código de servidor de Internet

Foi adaptado o código de um servidor de Internet para a Virtex-2. Esta aplicação usa o protocolo http 1.1 e envia informação entre clientes conectados via cabo de rede.

Devido a limitações nas livrarias da Xilinx®, apenas é possível enviar ficheiros até 1500 bytes. Outra importante limitação deste servidor é o facto de não ser possível efectuar múltiplos acessos ao servidor em pequenos intervalos de tempo.

As livrarias utilizadas são:

- "Xilinx Memory File System library (LibXil MFS)v1.00a", esta livraria trata parte da memória como ficheiros de sistema.
- "Xilinx Net v1.00a", esta livraria usa o controlador de rede para promover uma aplicação em camadas de alto nível designada Socket API.

- Os dados recebidos do Socket "s" são guardados em "xilsock_sockets[s].recvbuf.buf".
- O tipo de pacote é identificado por "xilsock_status_flag". O bit XILSOCK_TCP_DATA desta "flag" indica se o pacote contém dados. O bit XILSOCK_TCP_ACK indica se é um pacote de confirmação.
- Ao usar "xilsock_send", é necessário guardar os primeiros bytes no início do "buffer", o espaço a reservar para esta operação deve ter as seguintes dimensões:

  LINK_HDR_LEN + (IP_HDR_LEN*4) + (TCP_HDR_LEN*4)
- Esta função não consegue transferir informação superior a 1500 bytes e não espera pela confirmação do servidor antes de transmitir novos pacotes.

Hardware usado: EMAC, INTC, GPIO, ZBT.

O código fonte deste servidor foi modificado, a partir do exemplo extraído de: http://www.xilinx.com/products/boards/multimedia/docs/examples/MicroBlaze

9.1.8 SerialWatcher

Esta aplicação desenvolvida em Delphi®, foi fundamental no decorrer deste projecto para o processo de "debuging" do sistema.

O SerialWatcher permite visualizar os dados recebidos na porta série. É possível seleccionar a porta série que se pretende utilizar e configurar as suas definições para receber os dados enviados pela FPGA no formato desejado. Permite seleccionar qual o modo de apresentação da informação que se pretende visualizar, sendo as opções: hexadecimal, decimal e ascii. Esta interface permite também a criação de um ficheiro onde será armazenada toda a informação recebida.

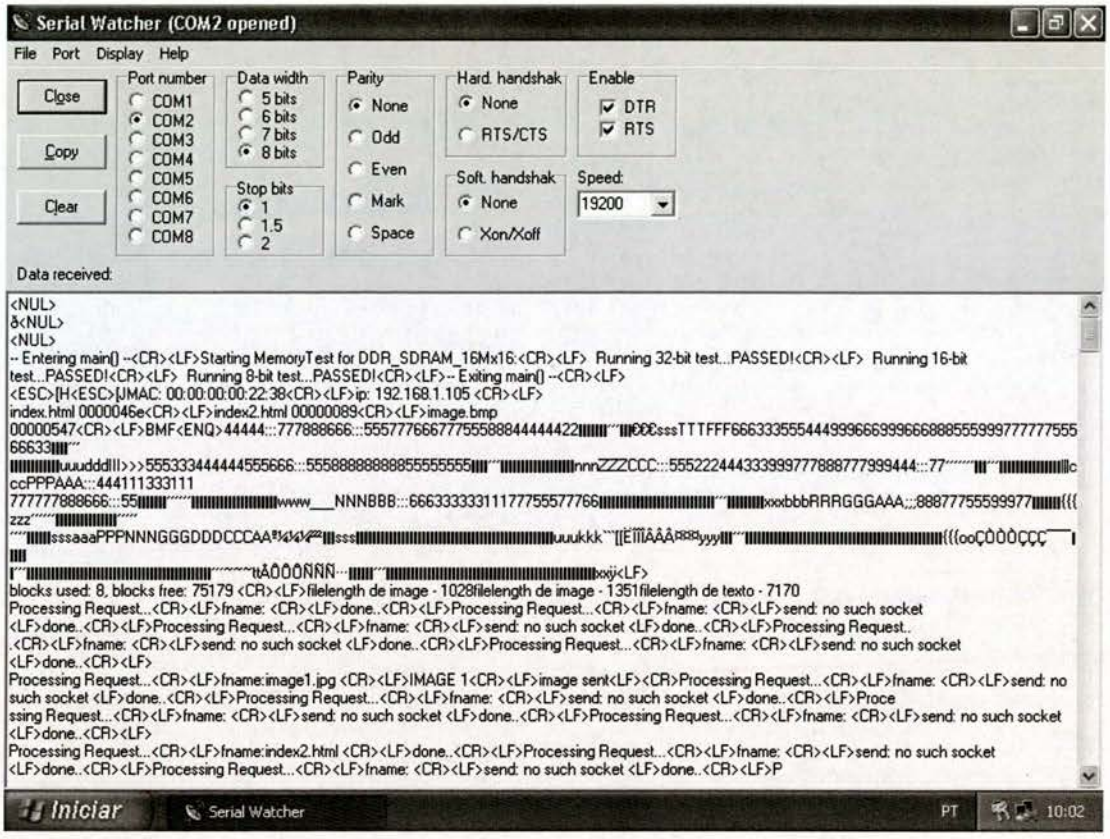


Ilustração 3 - SerialWatcher

9.1.9 Ethereal

O Ethereal permite a visualização dos pacotes transferidos pelo terminal da placa de rede. Para trabalhar com este software existem alguns passos fundamentais que deve seguir:

- Deve-se colocar um filtro com o objectivo de se observar apenas a informação considerada útil e de seguida premir "apply".
- Para iniciar a captura dos dados no filtro selecciona-se Capture> Interfaces> Prepare (da Interface que pretende visualizar) e coloca-se um visto nas opções:
  - Update List of packets in real time.
  - Automatique scrolling in live capture.
  - Hide capture info dialogue.

E retiram-se os vistos das opções:

- Enable MAC name resolution.
- Enable Transport name resolution.

Após seguir estes passos deve-se abrir o browser e a paritr desse momento tem-se acesso ao conteúdo de todos os pacotes transmitidos entre o cliente e o servidor. Este software possibilita o acesso a informação ao nível do "frame", "ethernet II", IP, TCP e HTTP.

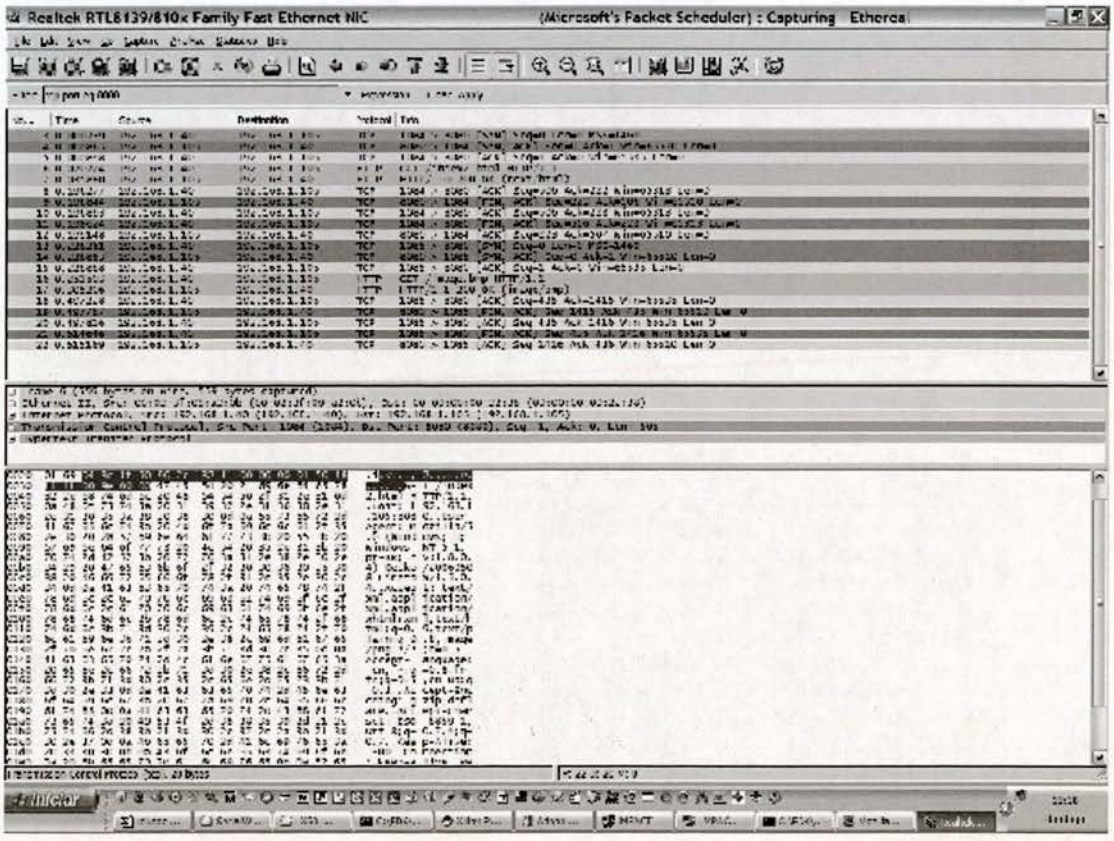


Ilustração 4 - Ethereal

9.2 Hardware utilizado:

9.2.1 Computador utilizado

Foi necessário que o computador utilizado tivesse certas características para ser possível a realização deste projecto. Para diversos testes e para a implementação da recepção de uma imagem no PC foi fundamental o uso da porta série. O programa iMPACT que trata a programação da FPGA, usa como meio de transmissão de informação o cabo JTAG que se liga à porta paralela. A placa de rede é necessária para a interligação via cabo de rede com o kit da FPGA e assim aceder ao servidor de Internet lá implementado.

9.2.2 Kit Virtex-4

O Kit Virtex-4 vem equipado com a placa FPGA Virtex-4 da Xilinx®, este kit está equipado com o hardware necessário para responder às funcionalidades desta placa. Simultaneamente também possibilita a implementação de aplicações mais complexas sem necessitar de recorrer a mais recursos de hardware.

Ilustração 5 - Kit Virtex-4 XC4VLX60.

Este kit foi desenvolvido para a FPGA Virtex-4, que é um dispositivo lógico reprogramável, esta

FPGA tem 668 pinos e um "flip-chip BGA package (FF668) ". O dispositivo FF668 tem 448 portas de entrada e saída, separadas em blocos de 10 portas de entrada e saída, que pode observar na seguinte ilustração.

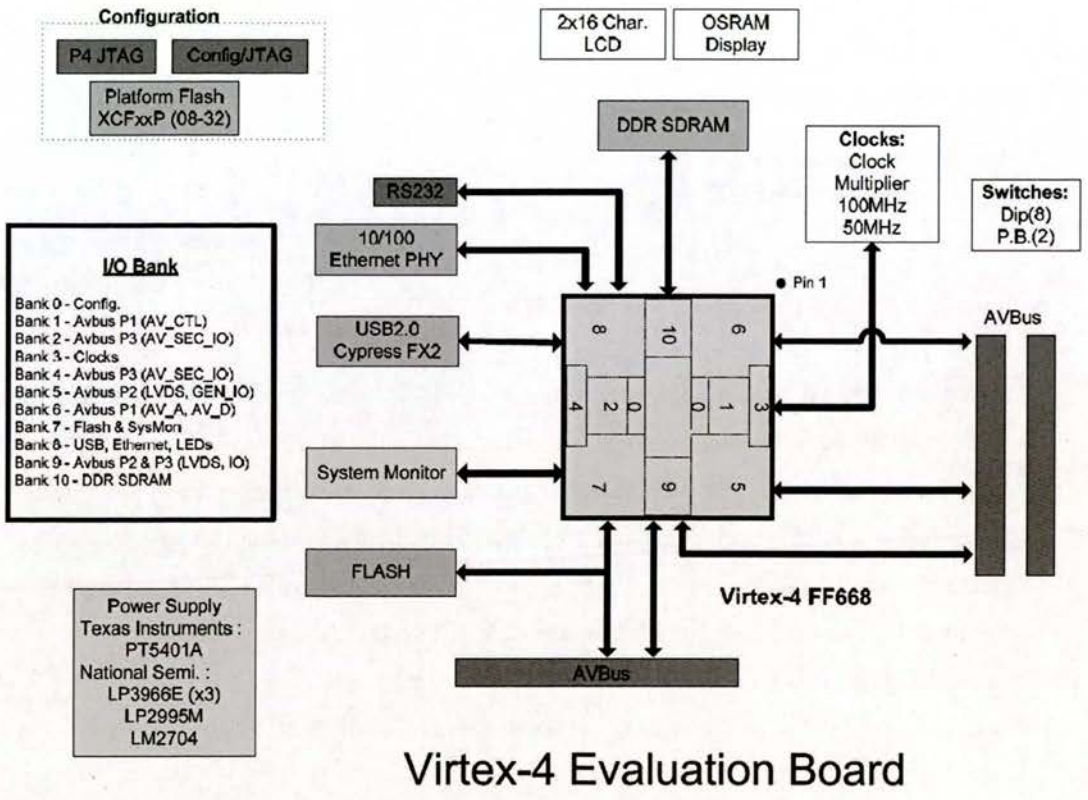


Ilustração 6 - Componentes de hardware da placa.

Esta placa suporta vários métodos de configuração, tais como: "Boundary-Scan", "Master Serial" e "Master/Slave Parallel". Durante a realização deste projecto a programação da placa foi efectuada no modo "Boundary-Scan", para este efeito é necessário o cabo JTAG, se for usado o cabo paralelo IV deve-se colocar um "jumper" no pino JP17.

Esta placa permite ao utilizador adicionar ou remover componentes da cadeia JTAG. O modo "default" da placa é o "standalone mode", em que é adicionado à cadeia JTAG o componente PROM. Para seleccionar este modo deve-se instalar um "jumper" nos pinos 2 e 3 do pino JP20, para seleccionar o modo "Boundary-Scan" deve instaslar dois "jumpers" nos pinos 2, 3, 5 e 6 do

pino JP18, tal como apresentado na ilustração.

Ilustração 7 - Configuração modo de boundary scan.

Ilustração 8 - Configuração modo de Standalone.

Utilizando este kit é possível implementar circuitos integrados bastante complexos, devido aos sucessivos avanços desta tecnologia e ao facto da velocidade de funcionamento ser bastante elevada, é possível efectuar um exaustivo estudo durante o desenvolvimento da aplicação a criar. Desta forma quando é detectado um erro no desenvolvimento da aplicação, apenas é necessário efectuar a correcção do mesmo e reprogramar a placa. Assim que a sua aplicação apresente a máxima eficiência deve ser implementada em ASIC, desta forma não é necessário a implementação em ASIC para a realização dos testes intermédios.

Para além do kit foi utilizado um "breakout module" para fazer a conexão entre o kit e o "flat cable" que faz a ligação com a câmara.

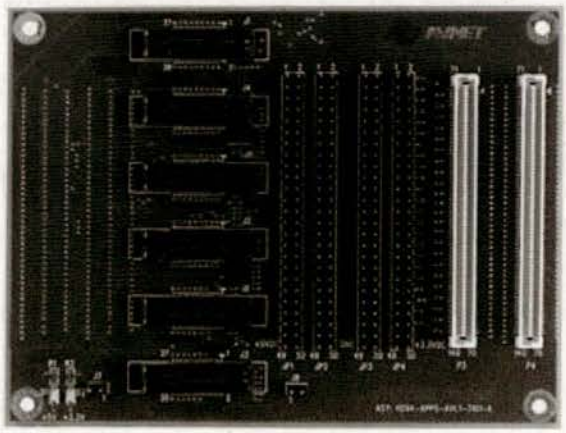


Ilustração 9 - AvBus breakout module.

9.2.3 Câmara digital C3188A

Este módulo contém um sensor CMOS OV7620 da Omnivision®, capaz de fornecer imagens até 664x486 pixels, sendo o valor da configuração por defeito uma resolução de 640x480 podendo também ser configurável para 320x240 a uma taxa máxima de 60 quadros por segundo (em formato QVGA).

Ilustração 10 - Módulo da câmara completo.

O módulo contém todos os componentes necessários para seu funcionamento do sensor, tais como condensadores e resistências de desacoplagem e rectificação, além de possuir também um cristal que fornece um sinal de relógio ao sensor à frequência de 27MHz.

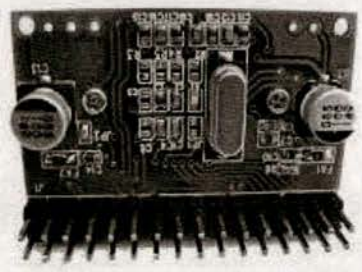


Ilustração 11 - Módulo da câmara visto de baixo.

Uma grande vantagem no uso deste módulo é o facto de o seu custo ser mais baixo que outros no mercado, isto porque a tecnologia de fabrico é praticamente a mesma dos dispositivos CMOS

habituais. Outra das vantagens é a possibilidade de integrar no mesmo "chip" do sensor circuitos para efectuar processamento analógico e digital, que podem mesmo ser efectuados ao nível do "pixel".

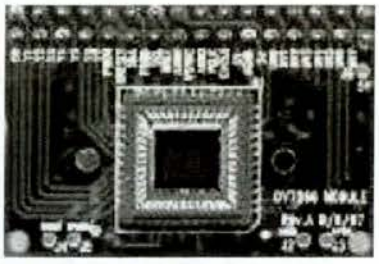
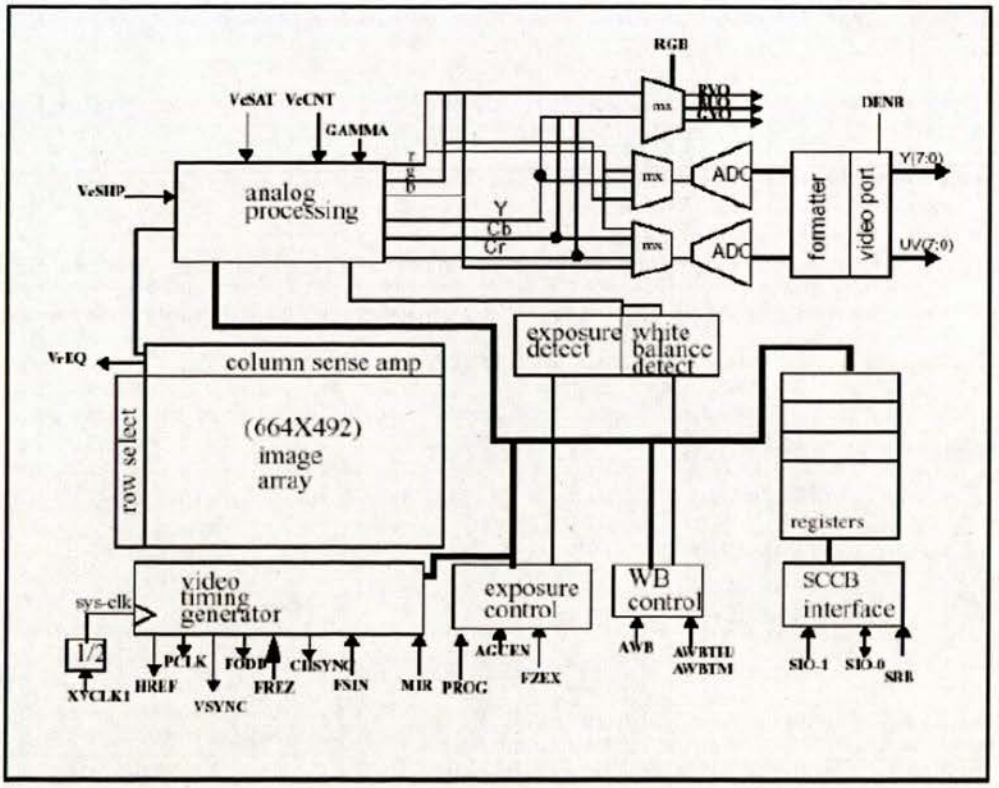


Ilustração 12 - Sensor CMOS OV7620.

De seguida referem-se algumas das suas características mais importantes, transcrevendo os dados da sua folha de características.



Fluxograma 1- Sensor OV7620.

Como se pode observar através do fluxograma, este componente tem potencialidades para além das de um mero sensor de imagem. No diagrama são diferenciáveis oito componentes principais:

- O "array" de imagem, onde a imagem é capturada e posteriormente enviada para o bloco de processamento analógico.
- O gerador de sincronismo de vídeo é o bloco responsável por todos os sincronismos da câmara, recebe do pino 27 um sinal de relógio, com uma frequência máxima de 30Mhz, e gera todos os sinais de sincronismo necessários: sincronismo horizontal e vertical, pixel clock, "horizontal reference", entre outros. Estes sincronismos são gerados de acordo com os parâmetros de configuração existentes nos registos, pois a velocidade do pixel clock dobra quando a saída é de 8 bits em relação a 16 bits, e há diferentes requisitos temporais conforme o formato seleccionado (progressivo ou entrelaçado por exemplo). É também possível por alteração no valor dos registos, mudar a polaridade dos sinais de sincronismo.
- O bloco de processamento analógico realiza funções tais como o controlo automático de ganho e de balanceamento de brancos e o controlo da qualidade de imagem. Estes controlos de qualidade incluem "sharpness", "hue", ganho, controlo "gamma" e "antiblooming". Efectua também conversões do sinal para diferentes formatos normalizados de saída.
- Os dois conversores analógicos para digital contêm 10 bits cada. As entradas de ambos provêm de "multiplexers" de modo a ser possível seleccionar a origem dos dados a serem convertidos.
- O bloco de formatação de saída, permite formatar a saída segundo diversas normas. A selecção do formato é feita através dos registos de configuração do OV7620. A tabela seguinte representa com um "Y" os formatos de saída suportados:

		Interlaced		Progressive Scan	
Resolution		640x480	320x240	640x480	320x240
YUV 4:2:2	16Bit	Y	Y	Y	Y
	8Bit	Y	Y	Y	Y
	CCIR656	Y	Y	Y	Y
RGB	16Bit	Y	Y	Y	Y
	8Bit	Y	Y	Y	Y
	CCIR656[1]	Y	Y	Y	Y
Y/UV swap[2]	16Bit	-	-	-	-
	8Bit	Y	Y	Y	Y
U/V swap	YUV[3]	Y	Y	Y	Y
	RGB[4]	Y	Y	Y	Y
YG	16Bit	Y	Y	Y	Y
	8Bit	-	-	-	-
One Line	16Bit	-	-	Y	-
	8Bit	-	-	-	-
MSB/LSB swap[5]		Y	Y	Y	Y

Tabela 2 - Formatos de saída suportados pela OV7620.

- A porta de vídeo digital, consiste em dois barramentos de 8 bits cada. A taxa de transferência máxima é de quase 7 mega bytes por segundo. As correntes de saída são ajustadas automaticamente em função da carga.
- A interface SCCB é a opção de configuração dos registos internos que a câmara apresenta. As especificações desta interface obedecem às da mais conhecida interface I2C, sendo a taxa de transferência apresentada na folha de características a de 400 kb/s. O método de escrita num registo é efectuado transmitindo 3 Bytes, sendo o primeiro o endereço do dispositivo, o segundo indica o registo a escrever, e no terceiro situa-se o valor a colocar no registo. Algumas especificações do protocolo I2C são apresentadas no capítulo Protocolos

Implementados deste relatório.

- Os registos internos da OV7620 são onde estão armazenadas as configurações da câmara. Podem ser carregados valores nos registos através de configurações de pinos no "Power-up", ou então serem acedidos através da interface SCCB. A câmara possui 125 registos, estando 70 dos quais reservados para uso interno do componente. Uma explicação mais detalhada da função de cada registo pode ser observada na folha de características do componente.

10 Trabalho desenvolvido

Nesta secção do relatório serão descritas em pormenor as diferentes partes constituintes do projecto, assim como os problemas que surgiram no decorrer do mesmo e as soluções encontradas. Embora uma grande parte do trabalho tenha resultado no desenvolvimento de módulos na linguagem de descrição de hardware Verilog, foi também desenvolvido código em Delphi e em C++. Podem ser encontrados na secção de anexos, com os comentários necessários para uma boa compreensão e avaliação do mesmo, o código implementado considerado relevante.
Nesta parte do relatório estão ainda as explicações dos motivos que levaram a uma determinada implementação dum módulo e uma descrição do comportamento do mesmo, auxiliada por uma representação fluxograma, sempre que tal ajudar na sua compreensão.

10.1 Protocolos implementados

10.1.1 RS-232

O estudo do protocolo RS-232 foi fundamental durante o processo de envio dos dados. Para a visualização dos dados recebidos na porta série pode-se usar a interface SerialWatcher ou o HyperTerminal do PC.
Iniciar> Acessórios> Comunicações> HyperTerminal
A informação pode ser enviada a uma cadência compreendida entre 110 e 921600 bits por segundo, podendo enviar-se entre 5 a 8 bits de dados. Este protocolo permite também o envio da informação com diferentes tipos de paridade, tais como: par, impar, marcar, espaço e sem paridade. Permite o envio de 1, 1.5 ou até 2 bits de paragem, pode efectuar controlo de fluxo por hardware, "Xon/Xoff", ou mesmo não efectuar controlo de fluxo.

10.1.2 Ficheiro de mapa de bits

Os ficheiros bmp são constituídos por um cabeçalho de 54 bytes, que tem os seguintes campos: tamanho da imagem em bits, offset, largura e altura em pixeis, número de bits por pixeis, tipo de compressão, resolução da imagem, entre outros campos reservados para uso futuro e campos com valor previamente estabelecido. O cabeçalho da imagem ocupa os primeiros 54 bytes e os restantes

bytes são reservados para o armazenamento dos dados da imagem. Cada pixel é constituído por 3 bytes, cada byte indica o valor de R, G e B, desta forma se se quiser guardar uma imagem monocromática deve-se replicar o mesmo byte três vezes. Os ficheiros bmp têm ainda uma particularidade, os dados da imagem são escritos de forma inversa, portanto após guardar a sua imagem deve-se ter o cuidado de a inverter.

10.1.3 I2C

O protocolo I2C foi desenvolvido pela Philips® no início dos anos 80 e tem como principal objectivo simplificar a comunicação digital entre dispositivos. Este protocolo utiliza apenas duas linhas de comunicação, o Serial Data (SDA) e o Serial Clock (SCL). Estabelece que para a transmissão de dados é necessário um sinal de Start, um conjunto de parâmetros para a transmissão de dados e a existência de um sinal de Stop. Devido à sua simplicidade o protocolo I2C é um standard em comunicação, sendo o mais usado até ao momento. Neste protocolo é referido que enquanto a linha de SCL se mantém no nível lógico válido, não pode haver mudança na linha SDA. Como excepções a esta regra, existem os sinais de Start e Stop. É detectado um sinal de Start quando a linha SDA efectua uma transição do nível alto para o nível baixo enquanto a linha SCL se mantém no nível lógico alto. Por sua vez, o sinal de Stop é detectado quando a linha de SDA passar do nível baixo para o nível alto, enquanto a linha SCL se mantém no nível lógico alto.

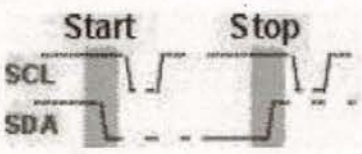


Ilustração 13 - Sinais de Start e Stop em I2C.

O barramento I2C é um barramento que permite que a ligação de vários dispositivos em simultâneo. Cada um deles é identificado por um ou dois endereços (um para escrita e um para leitura). De modo a permitir isso as linhas SDA e SCL são linhas "abertas", ou seja, quando não estão a ser utilizadas, estão ao nível lógico alto. Isto é garantido pela utilização de resistências de "Pull Up" nas linhas. Os dispositivos I2C ligados ao barramento podem ser de dois tipos: "Master" ou "Slave". O "Master" é o dispositivo que dá início à transferência e que gera o sinal de SCL

sendo possível a existência de vários "Masters". Como as linhas são abertas, o "Master" pode detectar se o barramento está a ser utilizado ou não, de modo a poder iniciar uma transferência. Cabe ao "Slave" gerar sinais de "Acknowledge" de modo a confirmar a recepção dos dados.

Uma das vantagens deste protocolo é o facto de a velocidade de transmissão não ser fixa, pois é determinada em cada transmissão através do SCL. Há no entanto alguns valores padrão mais utilizados, tais como 100kb/s e 400kb/s para Fast I2C.

Para a câmara digital C3188A é apresentado na sua folha de caracterísicas o diagrama temporal para a comunicação com o dispositivo, assim como os valores temporais. Onde o SIO-O é a linha SDA e o SIO-1 é a linha SCL.

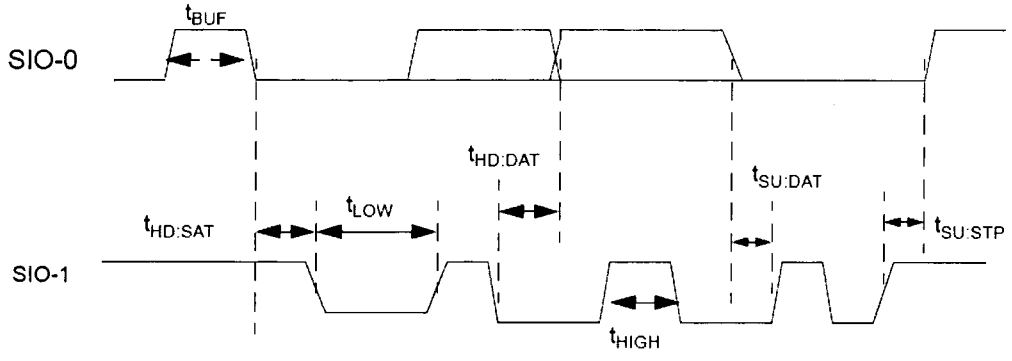
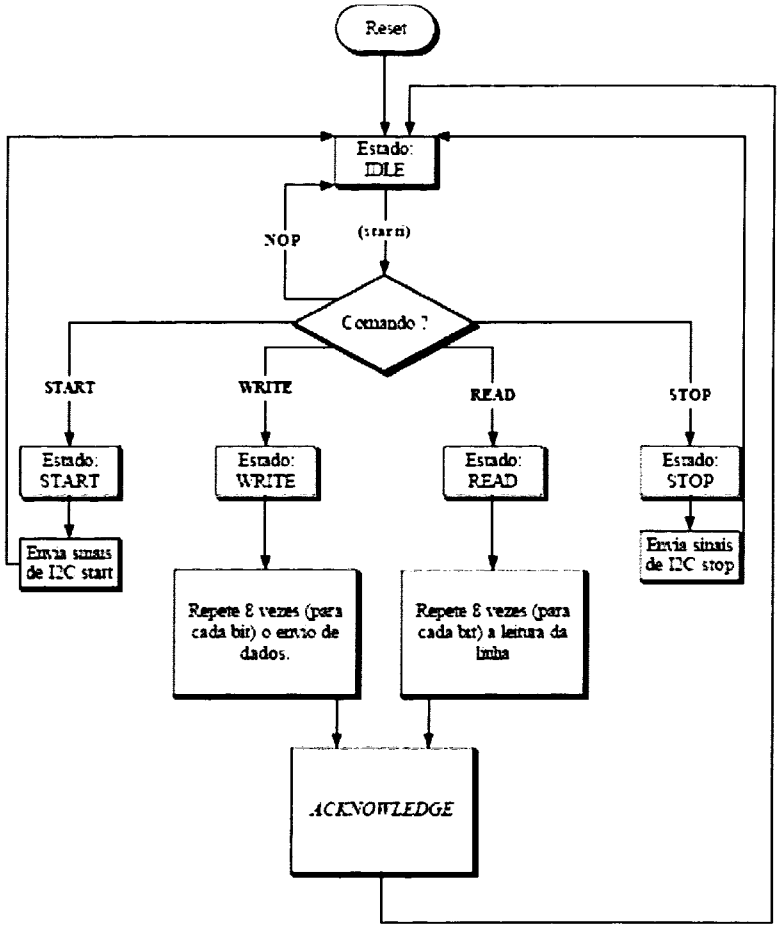


Ilustração 14 - Diagrama temporal para comunicação I2C com a câmara.

Symbol	Descriptions	Min	Max	Typ	Units
SCCB timing(400kbit/s)					
t_{BUF}	Bus free time between STOP & START	1.3	-	-	us
$t_{HD:SAT}$	SIO-1 change after START status	0.6	-	-	us
t_{LOW}	SIO-1 low period	1.3	-	-	us
t_{HIGH}	SIO-1 high period	0.6	-	-	us
$t_{HD:DAT}$	Data hold time	0	-	-	us
$t_{SU:DAT}$	Data set-up time	0.1	-	-	us
$t_{SU:STP}$	Set-up time for STOP status	0.6	-	-	us

Tabela 3 - Valores temporais para a comunicação I2C da câmara.

Este protocolo foi implementado no projecto, para ser possível transmitir uma imagem a preto e branco em modo entrelaçado sendo necessário alterar o conteúdo do registo 28 do OV7620. Os módulos responsáveis pela comunicação I2C com a câmara, comand_i2c, i2c e clkdiv50, encontram-se na secção de Anexos deste relatório. O módulo i2c, desenvolvido por Paulo Serra, recebe comandos através de um registo, e um sinal de start. Quando detecta um flanco positivo no sinal de start, efectua a operação indicada pelo registo, que poderá ser enviar por I2C um sinal de start, stop, ou então enviar ou receber dados. O registo que envia os comandos ou dados para o módulo I2C está ligado ao módulo comand_i2c. A implementação do módulo propriamente dito segue uma arquitectura do tipo máquina de estados. Cujo funcionamento pode ser melhor compreendido observando o diagrama de fluxo seguinte:



Fluxograma 2 - Funcionamento do módulo i2c.

No teste da implementação do protocolo I2C, verificou-se que apesar de o módulo estar validado para funcionamento a 400kb/s e na datasheet da câmara ser indicado que o componente suporta a comunicação a essa taxa, tal não acontece. E a comunicação com a OV7620 só consegue ser efectuada se for utilizada uma taxa de cerca de 100kb/s, sendo portanto necessária a existência de um relógio que forneça essa temporização, o módulo clkdiv50 é responsável por criar esse relógio a partir do relógio do sistema. Esta discrepância poderá ser devida a qualquer elemento da placa C3188A que interfira com a taxa de comunicação, ou então ao facto de a datasheet apresentada ser uma versão preliminar, e a comunicação a 400kb/s poderia estar prevista pela Omnivision, mas não ter chegado a ser implementada.

10.1.4 Codificação JPEG da imagem

Foi necessário efectuar compressão de imagem para permitir uma maior taxa de transmissão. Esta fase do trabalho exigiu bastante esforço e dedicação pois foi necessário efectuar uma pesquisa em relação a cada uma das fases deste processo. No entanto devido aos atrasos que decorreram da fase de transmissão de imagem via Ethernet, não foi possível, até à data, testar plenamente a codificação da imagem.

Nos testes iniciais efectuados, usou-se "Lossy Image Compression", pois esta compressão proporciona taxas de compressão mais elevadas do que "Lossless Image Compression", por esta razão eliminou-se a possibilidade de usar o formato GIF, ou o formato PNG e optou-se pelo formato JPEG. Apesar de se perder algum conteúdo da imagem ao usar "Lossy Compression", concluiu-se que este facto não seria perceptível ao utilizador, pois para o olho humano as diferenças entre a imagem antes da compressão e depois da compressão não são distinguíveis.

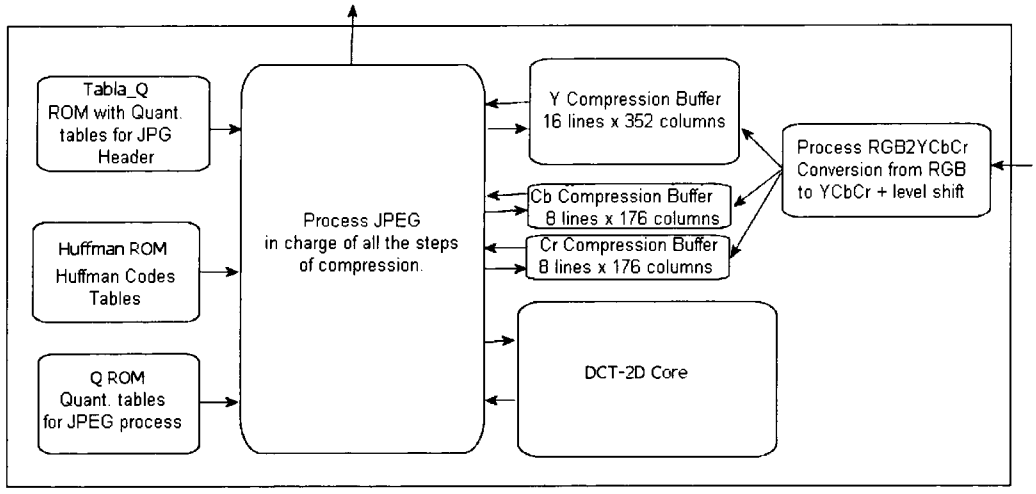
O olho humano é muito mais sensível ao detalhe da forma do que à informação contida na cor, apesar da taxa de compressão de uma imagem a cores ser mais elevada do que a de uma imagem a preto e branco, optou-se por comprimir uma imagem a preto e branco, pois como já foi referido a finalidade deste projecto seria o uso em operações de vigilância. Ao mesmo tempo, a aquisição da imagem a cores obrigaria à compressão de 921600 bytes (640*480*24/8). Uma vez que fazendo a aquisição da imagem a preto e branco apenas é necessário comprimir 307200 bytes (640*480), concluiu-se que apesar de a taxa de compressão da imagem a cores ser mais elevada, a aquisição de uma imagem a preto e branco proporciona a taxa de transmissão mais elevada, pois é necessário transmitir menos informação.

A compressão JPEG começa por converter o "array" dos dados original de duas dimensões em três "arrays" de 2 dimensões, um "array" contém informação de luminância e os restantes dois têm informação de crominância. Cada um destes "arrays" é dividido numa grelha de duas dimensões de 8 linhas e 8 pixeis, obtendo um total de 64 valores por bloco. Estes "blocos" de 8*8 são a chave da codificação JPEG. Cada um dos valores destes blocos deve ser convertido à sua forma espectral, ou seja deve passar do domínio dos tempos para o domínio da frequência, esta conversão é efectuada via DCT ("Discrete Cosine Transforme"). O próximo passo é o processo de compressão denominado "Quantização", neste processo é eliminada alguma da informação contida na imagem.

Como o olho humano é menos sensível a "high harmonics" ("fine details") do que aos "low harmonics"("coarse details"), logo estes podem ser representados com menos bits do que os "low harmonics". Neste caso estamos a trabalhar com valores de 8 bits, os "lowest harmonics" são representados por 8 bits e os "highest harmonics" são convertidos em valores de 4 bits. Para realizar esta operação é apenas necessário dividir o valor original (de 8 bits) por 16, para tal basta realizar um shift do valor original. Para efectuar a descompressão é necessário multiplicar o valor obtido por 16, obtendo-se assim uma razoável aproximação do valor original. O "factor de quantização" pode variar entre 1 (não há perda de informação) e 128 (compressão do valor para 1 bit). Este factor de quantização é diferente para cada um dos 64 coeficientes do bloco. É usada uma tabela de factores de quantização de 64 elementos, para atribuir os factores de quantização, esta tabela é aplicada a todos os blocos. Após a conversão do sinal no domínio das frequências e sua respectiva quantização, o algoritmo de compressão JPEG, converte o resultado num "array" linear de uma dimensão (vector) de 64 valores. É efectuado um "scan" em "zig-zag", este processo coloca os valores por ordem crescente de frequência, como os valores de altas-frequências têm uma maior probabilidade de serem zero após a quantização, os valores nulos são agrupados nas últimas posições do vector. Desta forma podem-se eliminar estes últimos valores dos vectores pois sabemos que têm valor zero (lossless compression algorithm). Por fim é usada a tabela de Huffman para finalizar a compressão da imagem.

Neste caso efectuamos a compressão de uma imagem a preto e branco, que é equivalente a comprimir apenas a componente "luminância". Nesta fase foi usado um compressor JPEG implementado em hardware, retirado do repositório OPENCORES em www.opencores.org, que o seu uso não teria sido possível sem um estudo pormenorizado. Como o compressor foi implementado numa placa virtex-2, utilizando IP Cores para essa FPGA, foi necessário gerar novamente todas as IP Cores no ISE para que este compressor funcionasse com o hardware da placa virtex-4.

O código está transcrito na secção de Anexos deste relatório. Para uma melhor percepção do seu funcionamento apresenta-se o seguinte fluxograma que implementa o processo de compressão acima descrito.



Fluxograma 3 - Compressor JPEG implementado em hardware.

10.2 Aquisição da imagem

10.2.1 Placa Orcad

Para a interligação do módulo da câmara com o "breakout module" foi necessário criar uma placa de circuito impresso. Para o seu desenvolvimento recorreu-se à ferramenta ORCAD, tendo como guia de trabalho o texto Formação de Orcad de 2005, de Manuel Silva, aluno da FEUP com quem foi possível esclarecer algumas dúvidas. O layout da placa desenvolvida pode ser observado na figura seguinte:

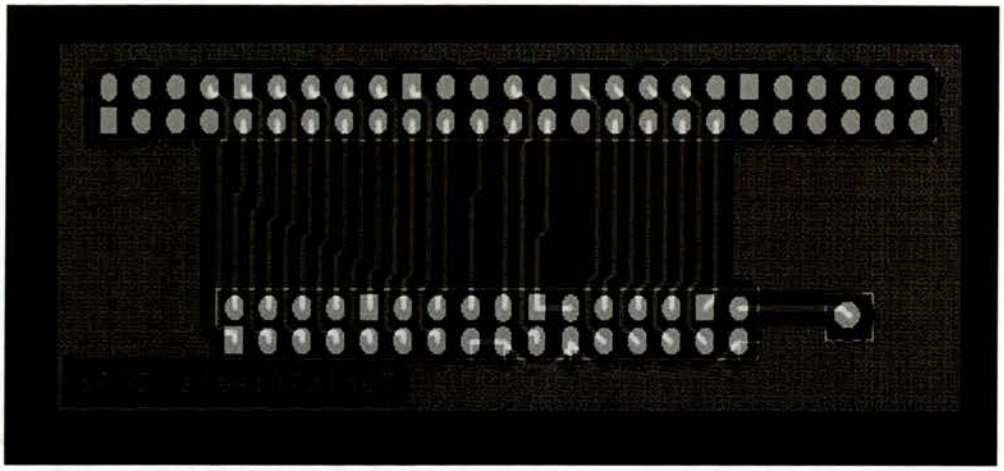


Ilustração 15 - Layout da placa.

Após o seu fabrico e soldadura dos componentes, vista inferior:

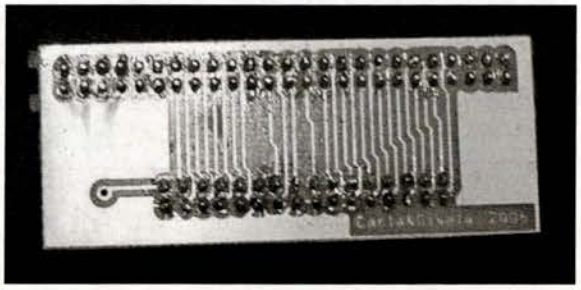


Ilustração 16 - Placa vista pela parte das soldaduras.

Uma vez concluída a placa foi possível fazer a atribuição dos pinos da câmara aos pinos do conector do "breakout module" e deste ao da FPGA. Segue-se uma tabela com essa atribuição.

P2	FPGA	JP3	Câmara	Câmara
2		1		
3		2		
5		3		
6		4		
8		5		
9		6		
11		7		
12	F3	8	1	Y0
14	F4	9	2	Y1
15	E1	10	3	Y2
17	E2	11	4	Y3
18	C2	12	5	Y4
20	D3	13	6	Y5
21	A3	14	7	Y6
23	C4	15	8	Y7
24	D4	16	9	PWDN
26	D5	17	10	RST
27	E5	18	11	SDA
29	D6	19	12	FODD
30	E6	20	13	SCL
32	A7	21	14	HREF
33		22		
35	C7	23	16	VSYN
36		24		
38	F7	25	18	PCLK
39	F8	26	19	EXCLK
41		27		
42		28		
44		29		
45	E10	30	23	UV0
47	A11	31	24	UV1
48	D11	32	25	UV2
50	G8	33	26	UV3
51	F12	34	27	UV4
53	G9	35	28	UV5
54	D13	36	29	UV6
56	E13	37	30	UV7
57		38		
59		39		
60		40		
62		41		
63		42		
65		43		
66		44		
68		45		
69		46		

Tabela 4 - Atribuição dos pinos da câmara aos da FPGA.

10.2.2 Captura e envio de imagem

Após a familiarização com o Kit fpga, realizaram-se os primeiros testes.

Foi efectuada a recepção de uma imagem com a tampa da lente na câmara, no entanto quando se tirou a tampa da lente e se captou uma nova imagem verificou-se a existência de falta de sincronismo além de perda de informação. Para detectar a causa das faltas, guardou-se na BRAM uma sequência de bytes formando um padrão, uma vez que a imagem recebida no PC foi a

esperada, verificou-se que a falha do sistema encontrava-se na aquisição dos dados transmitidos pela câmara e ao mesmo tempo foi possível confirmar o bom funcionamento do módulo de envio via RS232. A falta de sincronismo foi resolvida fazendo passar os sinais transmitidos pela câmara, usados para controlar a aquisição dos dados, por dois flips-flops ou mesmo três quando necessário, desta forma foi corrigido o problema de metaestabilidade e sincronismo. Em relação à perda de bits ao longo de uma linha a solução encontrada foi predeterminar o endereço no qual se iria escrever no início de cada linha. O módulo captura é o responsável pela captação e escrita de cada pixel/byte na BRAM, o módulo read lê os dados da BRAM e envia-os para o módulo suart encarregue da transmissão RS232, da autoria do Professor José Carlos Alves. Além destes módulos foi necessário criar um IP Core de uma BRAM, Read/Write, WidthxDepth = 8 bits x 307200 e um módulo toplevel que instancia todos os módulos e encarrega-se da sua interligação. Estes módulos podem ser encontrados na secção de Anexos.

Para ser possível a verificação deste sistema optou-se pelo desenvolvimento de uma interface gráfica capaz de efectuar a recepção dos dados na porta série. Esta interface foi desenvolvida para ser capaz de guardar os dados recebidos na porta série, e apresentá-los como um ficheiro de mapa de bits. Foram feitos alguns testes recorrendo a esta interface como meio de apresentação da imagem, tal como alterar o tamanho da imagem capturada para 320 por 240. Uma vez que a câmara transmite em modo entrelaçado antes de ser configurada, ao capturar apenas os 320 por 240 obtínhamos uma imagem de um quadro com metade da resolução em altura e metade do tamanho total em largura. Outros testes foram feitos para a implementação das restantes partes do projecto, mas sobre estes falaremos nas secções respectivas.

Para terminar esta secção, resta apenas referir que para efectuar a transmissão via cabo de rede foi usado um exemplo de um Servidor de Internet fornecido pela Xilinx juntamente com o kit, no entanto verificou-se que este exemplo não era compatível com o hardware da placa. Desta forma foi necessário adaptar um exemplo de um Servidor de Internet disponível na página da AVNET que implementa o protocolo TCP/IP. Para interligar os módulos referidos nesta secção com este Servidor e com a codificação JPEG foram feitas algumas alterações no respectivo código deixando de ser usado o módulo usart e o módulo read foi substituído pelo módulo control que passou a implementar não só a leitura da BRAM, mas também o envio para a FIFO. Esta FIFO é um IP Core que está interligado com o barramento OPB para implementação no MicroBlaze.

10.2.3 Desenvolvimento da Interface gráfica

Desenvolveu-se uma interface gráfica em Delphi capaz de receber os dados enviados pela porta série a uma cadência de 115200 bits por segundo. O número de bytes a receber é 921600 (640*480*24/8), uma vez que a imagem que se pretende receber tem uma resolução de 640 por 480 pixeis e uma resolução de cor de 24 bits. Efectuou-se a recepção de uma imagem a preto e branco, pelos motivos apresentados na escolha do material, desta forma foi necessário alterar o número de bytes a receber para 307200 (640*480). Como na imagem a preto e branco R=G=B é necessário escrever o mesmo byte em 3 pixeis consecutivos do ficheiro. Esta interface é responsável por receber os bytes da porta série, o primeiro byte é guardado na posição 55 do ficheiro de mapa de bits (Image.bmp), os restantes bytes são guardados em posições consecutivas.

Foram desenvolvidas 3 aplicações de teste no decorrer do processo de transmissão de imagem via RS-232. No entanto, do ponto de vista da utilização não existem alterações entre as diferentes aplicações.

Uma vez que o modo de configuração "default" da câmara é o modo entrelaçado de 8 bits, foi necessário desenvolver uma interface que permite receber os bytes da porta série e guardá-los num ficheiro bmp. Esta aplicação é responsável por apresentar uma imagem de 640 pixeis de largura e 480 pixeis de altura, uma vez que a imagem captada pela câmara no seu modo "default" tem estas dimensões. Ao receber os dados enviados pela câmara no modo entrelaçado de 8 bits, obtemos duas imagens consecutivas que contêm informação relativa a apenas uma frame. Para visualizar a imagem correctamente foi necessário desenvolver uma aplicação que após receber os bytes da porta série, interlaça as linhas da primeira metade do frame com as linhas da segunda metade do frame. Desta forma obtém-se a imagem apresentada na figura seguinte.

Ilustração 17 - Aplicação desenvolvida em funcionamento.

Foi necessário criar uma nova aplicação capaz de receber os bytes da porta série e apresentá-los sob a forma de um mapa de bits de dimenções 320 por 240. Desta forma diminui-se o tempo de transmissão de imagem para cerca de um quarto.

Guia de funcionamento da aplicação

- Para efectuar a aquisição de uma imagem, deve premir o botão SW4 do Kit Virtex-4 da Xilinx;
- Abrir o executável do programa;
- Seleccionar a porta COMx onde pretende efectuar a recepção dos dados, verificar o conteúdo da etiqueta ConfirmeSelection, para confirmar que a escolha da porta COM foi efectuada com sucesso;
- Premir OpenComX e verificar o conteúdo da etiqueta ConfirmeOpenComX, que indica que a abertura da porta COMx foi efectuada com sucesso. Deve também verificar o conteúdo da etiqueta ConfirmeConfiguration que indica que a configuração da porta COMx foi alterada com sucesso.

- Premir ReadFromComX ;
- Imediatamente após premir ReadFromComX deve alterar o botão 1 do Kit Virtex-4 da Xilinx para iniciar a transmissão da imagem guardada na BRAM;
- Cerca de vinte segundos após o início da transmissão de imagem deve surgir uma etiqueta com a seguinte informação: "Actualize image", deve verificar se a etiqueta ConfirmeReceiveData, indica que a transmissão foi efectuada com sucesso;
- Premir ActualizeImage e seleccionar Image.bmp

10.3 Implementação do Servidor de Internet

Foi necessário efectuar um estudo aprofundado deste servidor de Internet para proceder com alterações no código do mesmo. Inicialmente efectuaram-se os seguintes comandos para testar a página Web fornecida pela AVNET:

- Configurar o endereço IP em: Painel de Controlo> Ligações de Rede> TCP/IP (protocolo Internet) para: 192.168.1.40 e a máscara de sub rede para: 255.255.0.0.
- Guardar LX60_Web_Server_Design no c:
- Abrir system.xmp
- Software> Build All User Applications
- Device Configuration> Update Bitstream
- Iniciar> Programas> Xilinx ISE 8.1i> Accessories> Impact
- Usando este Software programou-se a placa com o ficheiro.bit
    - c: > LX60_Web_Server_Design > Implementation > download.bit
- Debug > Launch XMD
- Abrir o URL: http://192.168.1.105:8080/, onde 192.168.1.105 é o endereço IP da placa. Verificou-se o correcto funcionamento da página Web apresentada.

Posteriormente optou-se por transmitir uma imagem de 28 por 18 pixeis. Foi necessário converter um ficheiro de mapa de bits em código hexadecimal, para este efeito foi usado o código seguinte:

```
int main(int argc,char *argv[]){
     if(argc==3){
```

```
        FILE *f=fopen(argv[1],"rb");
        printf("unsigned char %s[]={\n",argv[2]);
        while(!feof(f)){
                printf("%d,",fgetc(f));
        }
        printf("};");
        fclose(f);
    }else{
        //printf("Usage: \"conv\" <image file> <variable name>   \">\"
<destiation file>");
    }
}
```

Quando a linha de código: "conv image.bmp bmp_image> bmp.h" é executada, cria o header bmp.h que contém o array bmp_image. Este header foi adicionado ao projecto e com a informação deste header criou-se o ficheiro image.bmp na função void create_index (void) em web.c, usando as funções do Sistema de Ficheiros de memória (LibXil Memory File System - MFS) disponibilizado pelo EDK.

```
if ( (fd = mfs_file_open("image.bmp", MFS_MODE_CREATE)) == -1) {
    xil_printf("image cant create\r\n");
    exit(-1);
  }
  if ( (fd = mfs_file_open("image.bmp", MFS_MODE_WRITE)) == -1) {
    xil_printf("image cant open for write\r\n");
    exit(-1);
  }

  if ( (tmp = mfs_file_write(fd, image_bmp, sizeof(image_bmp))) == 0) {
    xil_printf("cant write image\r\n");
    exit(-1);
  }
  mfs_file_close(fd);
```

Este ficheiro é posteriormente enviado via cabo de rede. Uma vez que os pacotes do protocolo TCP/IP têm uma capacidade máxima de 1500 bytes, esta imagem de 28 por 18 pixeis tem a dimenção de 1350bytes (24*18*3+54) e o header que identifica a informação enviada como um mapa de bits tem a dimensão de 70 bytes, logo é possível enviar toda a informação relativa a esta imagem num pacote. Para este efeito utilizaram-se as seguintes linhas de código, em void do_proc_req (int n):

```
unsigned char bmp_hdr[] = "HTTP/1.1 200 OK\r\nContent-length:
921659\nContent-type: image/bmp\r\n\r\n";

else if(strcmp(fname,"image.bmp")==0){
        xil_printf("IMAGE 0\r\n");
        memcpy(sndptr, bmp_hdr, strlen(bmp_hdr));
        memcpy(sndptr+strlen(bmp_hdr), image_bmp, sizeof(image_bmp));

        n = xilsock_send(web_conns[n].s, sendbuf,
strlen(bmp_hdr)+sizeof(image_bmp));
        xil_printf("image sent\n\r");
}
```

Após verificar a transmissão desta imagem via Ethernet, optou-se por implementar o hardware do sistema nesta aplicação, começou por se implementar um sistema capaz de acender o led6 (na placa) e escrever na fifo o valor FFAAFFAA sempre que é premido o botão SW4. Para este efeito foi necessário efectuar estes comandos:

- Hardware > Create or Import Peripheral > Create templates for a new peripheral > To an XPS project > Name: fifo > On-chip Peripheral Bus(OPB) > Select FIFO > Select Include Read FIFO ;
- c:> LX60_Web_Server_Design> pcores> fifo_v1_00_a> devl> projnav> fifo.ise
- Adicionou-se ao projecto o módulo teste.v;
- Adicionaram-se todas as portas de entrada e saída do módulo a fifo.vhd;
- Adicionaram-se todas as portas de entrada e saída a
    - c:> LX60_Web_Server_Design> pcores> fifo_v1_00_a> data> fifo_v2_1_0.mpd:

```
PORT sw4 = "", DIR = I
PORT led7 = "", DIR = O
PORT led6 = "", DIR = O
```

- Para adicionar o módulo teste.v foi necessário inserir em c:> LX60_Web_Server_Design> pcores> fifo_v1_00_a> data> fifo_v2_1_0.pao, a seguinte linha de código:
  - lib fifo_v1_00_a teste verilog;
- O projecto fifo.ise foi sintetizado e detectados todos os erros;
- No XPS em Project Information Area> IP catalog> Project Repository seleccionou-se fifo_0 e adicionou-se o projecto a Filters> Bus interface (estabeleceu-se a ligação entre o OPB e SOPB de fifo_0);
- Adicionou-se a Filters> Ports todas as portas externas;
- Adicionou-se a Filters> Addresses o endereço base da fifo, este endereço não deve provocar conflitos com os restantes endereçamentos (0x78c00000).

Em web.c foi necessário efectuar a aquisição do valor escrito na fifo:

```
volatile Xuint32 *address = (Xuint32*)0x78c00200;

xil_printf("fifo: %x\r\n", *address);
```

Premindo sw4 verificou-se que o led6 efectivamente acendeu e foi enviado para a porta COMx o valor FFAAFFAA escrito na FIFO. Desta forma, efectuaram-se os mesmos passos para implementar o código de aquisição de imagem no XPS, este código foi previamente testado usando a interface gráfica produzida no Delphi.

Durante a implementação deste servidor de internet verificou-se que existiam limitações a nível das dimensões dos ficheiros a transmitir, uma vez que a nível do protocolo TCP/IP o servidor não atende pedidos efectuados, separados por um curto intervalo de tempo. Optou-se por se enviar uma imagem bmp de 24 por 18 pixeis, esta imagem é ampliada e desta forma é possível visualizar a existência de movimento, pois é efectuada a actualização da imagem de 1 em 1 segundo.

Em web.c foi necessário adicionar o código de aquisição de dados da fifo e posterior criação do ficheiro a ser apresentado:

```
else if(strcmp(fname,"image.bmp")==0){
        xil_printf("IMAGE teste\r\n");

      if ( (imag = mfs_file_open("image.bmp", MFS_MODE_READ)) == -1) {
          xil_printf("image cant open for write image test\r\n");
          exit(-1);
      }
      fileLength2 = mfs_file_lseek(imag, 0, MFS_SEEK_END);
      xil_printf("filelength2: %d\n\r", fileLength2);

      if ( (tmp2 = mfs_file_read(imag, buf_imag_read, fileLength2)) == 0)
{
          xil_printf("cant read from image\r\n");
          exit(-1);
      }
      k=55;
      for(i=55;i<=486;i++){
          byte[i]=*address;
         buf_imag_read[k]=byte[i];
          k=k+1;
         buf_imag_read[k]=byte[i];
         k=k+1;
         buf_imag_read[k]=byte[i];
         k=k+1;
      }
      xil_printf("byte1:%x\r\n",byte[1300]);
      xil_printf("buf_imag_read:%x\r\n",buf_imag_read[1300]);
      mfs_file_close(imag);

      memcpy(sndptr, bmp_hdr, strlen(bmp_hdr));
      memcpy(sndptr+strlen(bmp_hdr),                    buf_imag_read,
sizeof(buf_imag_read));
      num   =   xilsock_send(web_conns[n].s,   sendbuf,   strlen(bmp_hdr)+
```

```
sizeof(buf_imag_read));
      xil_printf("size of buf_imag_read: %d\n\r", sizeof(buf_imag_read));
      xil_printf("image sent\n\r");
      }
```

Devido às reduzidas dimensões da imagem bmp apresentada, foi necessário implementar um codificador JPEG. O procedimento de apresentação da imagem JPEG é equivalente à apresentação da imagem BMP, as únicas diferenças residem no facto de se guardar a imagem a partir do primeiro byte do ficheiro e não ser necessário efectuar a replicação dos bytes recebidos. Verificou-se que é possível apresentar na página de internet uma imagem de 32 por 32 pixeis. Em web.c foi necessário adicionar o seguinte código:

```
else if(strcmp(fname,"image1.jpg")==0){
       xil_printf("IMAGE 1\r\n");
        memcpy(sndptr, jpg_hdr, strlen(jpg_hdr));
        memcpy(sndptr+strlen(jpg_hdr),  image_jpg3,  sizeof(image_jpg3));

       num         =         xilsock_send(web_conns[n].s,         sendbuf,
strlen(jpg_hdr)+sizeof(image_jpg3));
        xil_printf("image sent\n\r");
      }
```

Na página de Internet ilustrada abaixo é possível verificar a apresentação de uma fracção da imagem captada pela câmara. Esta imagem tem as dimenções de 24x18 pixeis e foi aumentado o seu tamanho recorrendo à diminuição da resolução da imagem em software. De momento está a ser efectuada a diminuição da resolução da imagem em hardware, pois desta forma será possível observar uma imagem mais perceptível, uma vez que será apresentado um maior campo de visão.

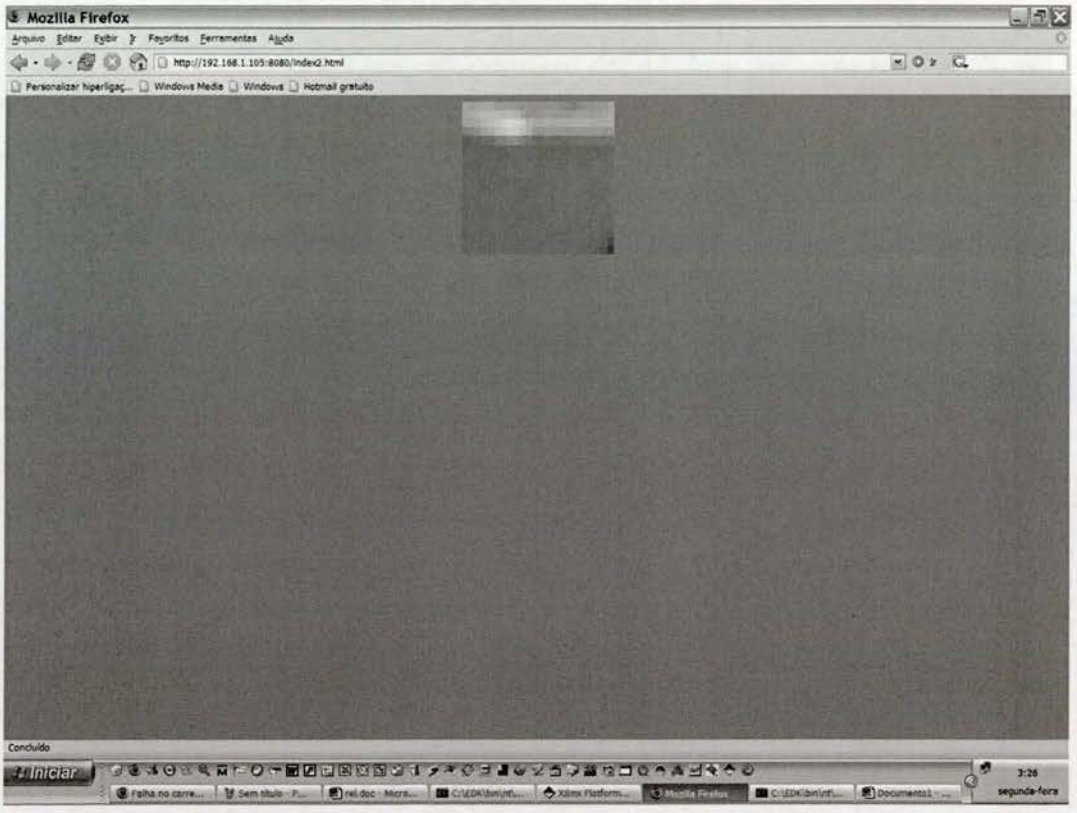


Ilustração 18 - Página de internet.

A imagem JPEG que se espera vir a encontrar tem as dimensões de 32x32 pixeis, na ilustração abaixo apresentada, pode-se verificar no início da página a apresentação de 4 imagens. Devido a atrasos na implementação do código de servidor de Internet não foi possível apresentar os testes resultantes da compressão da imagem JPEG. Portanto a apresentação desta imagem está dependente de alguns testes que se encontram em desenvolvimento. As restantes funcionalidades do servidor de Internet permitem acender os 4 leds do kit correspondentes a um valor inserido em hexadecimal. Após premir um ou vários "switchs" pode verificar o valor em binário correspondente ao "switch" premido.

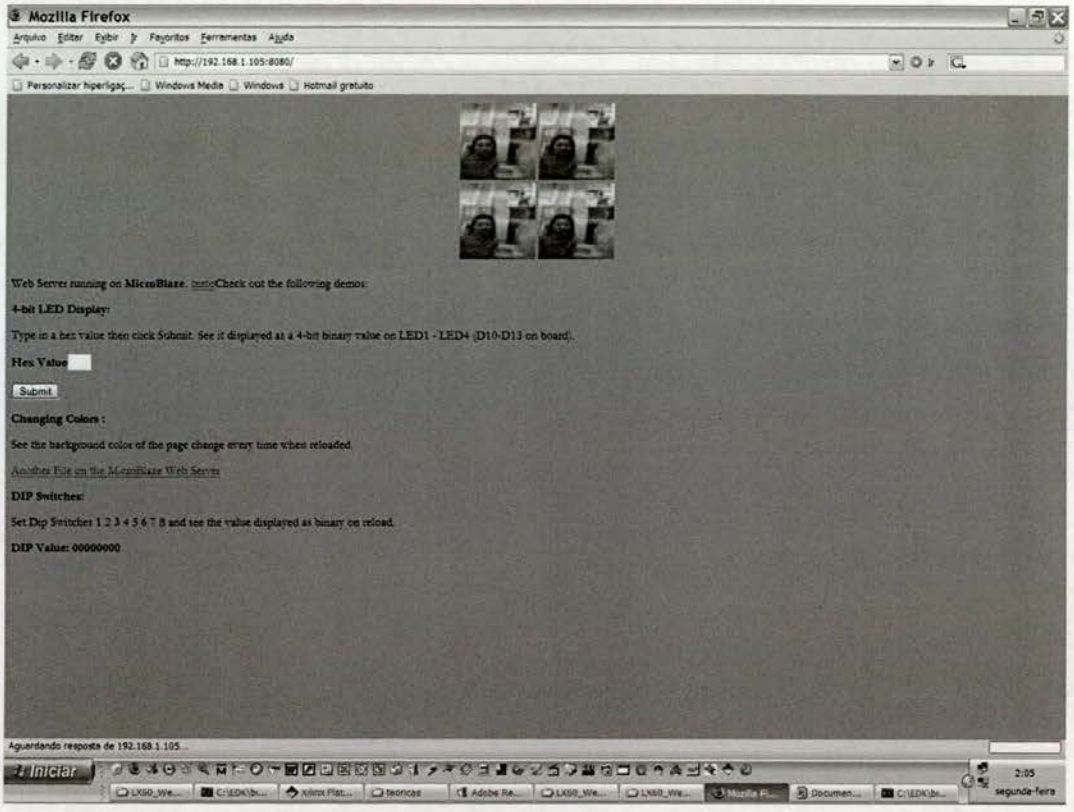


Ilustração 19 - Página Internet com imagem em JPEG.

11 Conclusão

Todos os objectivos propostos a nível de hardware foram cumpridos uma vez que foi possível apresentar na interface gráfica implementada uma imagem de 640x480 pixeis com atraso de 26 segundos. Uma vez que o projecto foi desenvolvido em hardware foi necessário ultrapassar grandes dificuldades a nível de ruídos, de faltas de sincronismo, de ausência ou escassez de informação nas folhas de características do material utilizado, entre outros aspectos em que a prática difere da teoria. Apesar de todo o esforço e tempo entregue a esta parte do projecto, consideramos que foi atingido o principal objectivo do mesmo uma vez que a aquisição e apresentação de uma imagem foram conseguidas com sucesso. Os módulos desenvolvidos nesta primeira fase poderão ser utilizados em futuras implementações que utilizem imagem.

Após ultrapassar esta fase, implementou-se um servidor de Internet capaz de apresentar numa página de Internet uma imagem captada pela câmara. Desta forma concluímos que esta fase teve um resultado bastante satisfatório uma vez que foram ultrapassadas as principais dificuldades de implementação. O protocolo TCP/IP usado neste servidor foi desenvolvido pela Avnet utilizando as livrarias da Xilinx que possuem limitações muito restritivas ao seu uso, por consequente não foi possível alcançar os objectivos iniciais relativos à apresentação de uma imagem com as dimensões e a uma taxa de transmissão desejados. Uma vez que cada pacote enviado tem a capacidade de 1514 bytes, e as livrarias da Xilinx não permitem que o servidor responda a sucessivos envios de pacotes num curto intervalo de tempo, não é possível a apresentação de imagens com ocupação superior a 1500bytes, o que corresponde a uma imagem de 24x18 pixeis em bmp.

Dado que a codificação da imagem em JPEG se encontrava em desenvolvimento e possibilita para o mesmo tamanho a apresentação de imagens de dimensões superiores. Foi testada a apresentação na página de uma imagem em JPEG de 32x32 pixeis, o que corresponde a um aumento da área visível da imagem de 58%. A resolução da imagem captada pela câmara foi reduzida para ser possível a apresentação de uma imagem mais perceptível e desta forma aumentar a qualidade do sistema.

Durante a implementação deste projecto foram encontrados muitos imprevistos que foram ultrapassados, com estes imprevistos surgiu constantemente a necessidade de implementar novas soluções. Durante a duração deste projecto foi para nós um privilégio trabalhar com uma placa de

tecnogia tão avançada como a XC4VL60-FF668 com uma FPGA Virtex-4. Esta placa possibilitou a implementação de todo o hardware e software pretendido, verificaram-se apenas algumas limitações das livrarias do software a nível da transmissão da imagem via cabo de rede. Por estas razões e muitas mais foi necessário ao longo de todo o projecto uma atitude de dinamismo e constante dedicação. Após a conclusão do projecto atingiu-se uma fase de satisfação por terem sido atingidos todos os requisitos de hardware e de software, apesar de as taxas de transmissão e dimensão da imagem a transmitir via Internet não serem as desejadas, o sentimento de satisfação mantém-se uma vez que foram resolvidos todos os restantes problemas.

12 Bibliografia

DOUGLAS E. COMER, INTERNETWORKING with TCP/IP – Principles, Protocols, and Architectures, fourth edition.

PHILIPS SEMICONDUCTORS, THE I 2C-BUS SPECIFICATION, Versão 2.1, Janeiro 2000

OMNIVISION, Advanced Information Preliminary OV7620/OV7120, Versão 2.1, Julho, 2001

BORLAND SOFTWARE CORPORATION, Developer's Guide, www.borland.com

BORLAND SOFTWARE CORPORATION, Component Writer's Guide, www.borland.com

MICROSOFT WINDOWS, Bitmap Format, http://netghost.narod.ru/gff/vendspec/micbmp/bmp.txt

HAMILTON, ERIC, JPEG File Interchange Format, C-Cube Microsystems, Versão 1.02, Setembro, 1992

CCITT, THE INTERNATIONAL TELEGRAPH AND TELEPHONE CONSULTATIVE COMMITTEE, Terminal Equipment and Protocols for Telematic Services, T.81, 09/92

Xilix Virtex-4 Evaluation kit-User guide

AvBus Breakout Mechanical guide

SILVA, MANUEL, Formação em OrCAD 2005, guia de referência

Entre outros guias, páginas de Internet e folhas de características de componentes.