Uma Arquitetura de *Hardware* e *Software* Livres para um Telefone VoIP[1]

J. C. S. Américo Filho[2], D. F. M. Mota[2], R. C. Mendes[2], J. A. N. da Silveira[3] e J. C. M. Mota[2]

***Resumo*—Neste trabalho, propomos uma solução de hardware e software para um telefone VoIP. O processador ADSP-BF518 compõe o núcleo do hardware. Todo o software desenvolvido e utilizado é livre.**

***Palavras-Chave*—VoIP, BF518, SIP, Liblinphone, RTP, DSP.**

Abstract— **In this paper we propose a solution of hardware and software for a VoIP phone. The processor ADSP-BF518 makes up the core hardware. All software developed and used is free.**

***Keywords*— VoIP, ADSP-BF518, SIP, Liblinphone, RTP, DSP.**

INTRODUÇÃO

A tecnologia de voz sobre IP (VoIP) possibilita a utilização de redes IP, como a Internet, para transmissão de sinais de voz. Diferente das redes de comutação de circuito, onde há um canal de comunicação dedicado, as redes IP fundamentam-se em comutação por pacotes, onde os atrasos podem ser longos e variáveis e, eventualmente, ocorrem perdas de pacotes. A natureza de tais redes, portanto, impõe desafios técnicos para a realização de telefonia digital de qualidade [1].

Neste artigo, propomos uma arquitetura de hardware e software de um telefone VoIP. Destacamos que tanto o hardware quanto o software aqui descritos são livres e disponíveis para a comunidade através do projeto Blackfin IP Phone [2].

ARQUITETURA DE HARDWARE

Para uma utilização eficiente dos recursos de rede, os sinais de voz, antes de serem tratados e transmitidos, necessitam ser codificados. Na recepção, por sua vez, os sinais são recuperados e devem ser decodificados para em seguida recomporem o sinal de voz transmitido. Além da codificação e decodificação, outros tratamentos, como cancelamento de eco e controle automático de ganho, fazem parte da recuperação do sinal. Todo esse processamento de sinais deve ser executado em tempo real, de forma que o uso de um processador de sinais digitais (DSP) se faz necessário. Além de processar sinais, um telefone VoIP deve tratar pacotes de rede e gerenciar a interface com o usuário. Essas duas atividades são melhor executadas por processadores de propósito geral (GPP). Assim, do ponto de vista de hardware, pode-se utilizar um circuito integrado que possui, no mesmo encapsulamento, um DSP e GPP. Outra possibilidade é utilizar um DSP com características de GPP, como controlador LCD e Ethernet. Embora menos flexível, a última solução é mais barata e atende aos requisitos da aplicação, de modo que utilizaremos esta arquitetura no *hardware* aqui proposto.

A Figura 1 apresenta o diagrama de blocos da solução de *hardware* da proposta. O processador utilizado, ADSP-BF518, é um DSP de ponto fixo de 32 bits da família *Blackfin* fabricado pela Analog Devices. Os transdutores de áudio e seu circuito de condicionamento são representados pelo bloco "Circuitos de Áudio". As conversões A/D e D/A são realizadas pelo circuito de codec e a interface Ethernet é implementada pelo bloco "PHY Ethernet". A interface com o usuário é composta de um teclado matricial, um display LCD, além de botões de funções e LEDs indicativos. Também é disponibilizado um conector para cartão de memória microSD.

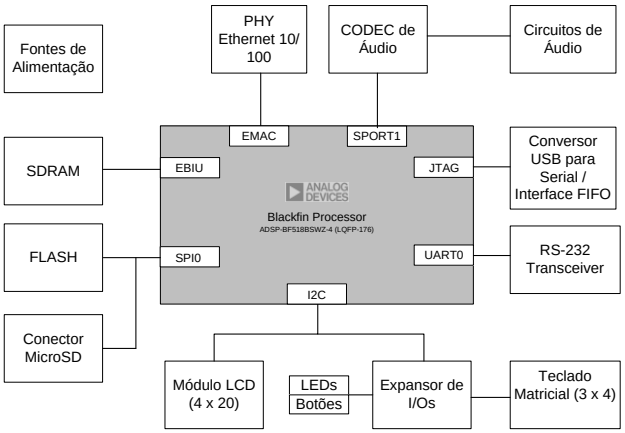


Diagrama de blocos da arquitetura de hardware.

ARQUITETURA DE SOFTWARE

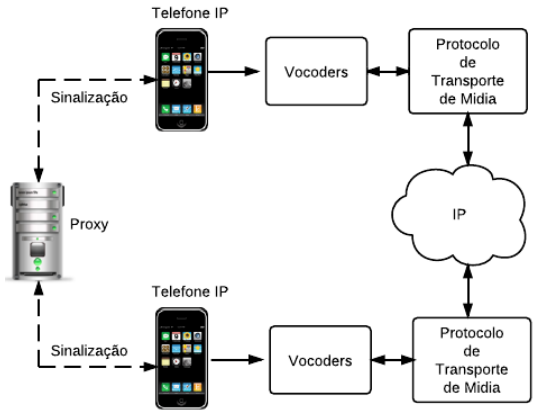
Um telefone VoIP é uma aplicação complexa o suficiente para justificar a utilização de um Sistema Operacional (SO) embarcado. Além de ocultar a complexidade do hardware, disponibilizar um sistema de arquivos e recursos de rede, o uso de um SO permite a reutilização de código já desenvolvido, reduzindo assim o tempo de projeto. De fato, implementações dos protocolos de sinalização, protocolos de transporte de mídia e processamento de streams de áudio encontram-se disponíveis para os principais sistemas operacionais embarcados.

O sistema operacional *uClinux* é livre, possui uma ampla comunidade de desenvolvedores e já encontra-se portado para a família de processadores *Blackfin*, o que o faz dele uma escolha natural como SO da arquitetura de software proposta. O sistema operacional é transferido para memória por outro programa denominado *bootloader*. O *U-Boot*, assim como o *uClinux*, é livre, extensamente utilizado e suporta o processador ADSP-BF518, de forma que o empregamos para a realização do *boot* do sistema.

A Fig. 2 esquematiza as principais partes de um sistema VoIP. A criação, manutenção e término da conexão entre os nós de comunicação são realizados pelo protocolo de sinalização. Com o intuito de reduzir os requerimentos de

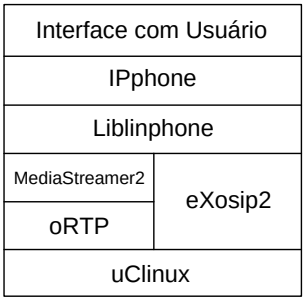
1 Trabalho com suporte financeiro parcial da CAPES, CNPq e do BNB.
2 Autores do GPSI/DETI/UFC.
3 Autor do LESC/DETI/UFC

largura de banda da rede, o áudio é codificado antes da transmissão e decodificado na recepção através de vários codecs padrões. Os sinais de voz codificados são transmitidos através da rede governados por um ou mais protocolos de transporte.



Principais partes de um sistema VoIP.

Há, para cada uma destas partes, bibliotecas de *software* de código aberto que podem ser integradas para o desenvolvimento da aplicação final. A Fig. 3 apresenta os pacotes que compõem a arquitetura de software proposta.



Arquitetura de Software Proposta.

INTERFACE COM O USUÁRIO

No nível mais alto da arquitetura está a camada de software responsável pela interface com o usuário. A sua implementação é baseada em uma máquina de estados finitos com eventos gerados tanto pelo usuário, através do teclado e botões de funções do telefone, quanto pela camada de software abaixo – *IPphone* – responsável pelos eventos relacionados à telefonia.

IPPHONE

Nesta camada, são gerados todos os eventos telefônicos utilizados pelo nível acima, como "Usuário Ocupado", "Chamada em Curso", entre outros. O gerenciamento das listas de chamadas, a manutenção da lista de contatos e a interface com a camada abaixo são realizados pelas funções disponibilizadas pela camada IPphone.

LIBLINPHONE

Liblinphone é uma biblioteca que implementa todas as funcionalidades de telefonia IP, como estabelecimento de sessão, gerenciamento de chamadas e Proxy, controle de parâmetros de mídia e rede dentre outros. Suas funções são de alto nível e fazem uso de outros componentes de software como oRTP, eXosip2 e mediastreamer2.

EXOSIP2

A aplicação proposta é um telefone VoIP compatível com o protocolo SIP. O protocolo de controle de sessão SIP objetiva estabelecer a presença e localização de usuários, assim como configuração, modificação e término de sessões [3]. SIP é utilizado em conjunto com um protocolo de controle de mídia denominado SDP, que negocia parâmetros de inicialização de mídia, como tipo de codec e endereço da porta de destino. Exosip2 é uma biblioteca de agente usuário SIP que oculta a complexidade de utilizar o protocolo SIP para estabelecer sessões de multimídia. Essa biblioteca é utilizada por Liblinphone.

ORTP

Após o estabelecimento e configuração da sessão, a mídia pode fluir entre os nós da rede através de algum protocolo de transporte, tais como TCP ou UDP. As redes IP prestam serviços de melhor esforço, portanto não há garantias quanto ao atraso e sua variação, o que compromete a qualidade das transmissões multimídia de tempo-real. Dessa forma, deve ser utilizado algum protocolo de transporte de mídia na camada de aplicação para garantir a qualidade do serviço. O protocolo RTP [4] sobre UDP é largamente utilizado para transporte de áudio e vídeo em aplicações de tempo-real. A biblioteca oRTP, que implementa a pilha RTP, é utilizada pela API Liblinphone.

MEDIASTREAMER2

Todo o recebimento e envio de fluxos de áudio - incluindo captura, codificação e decodificação - é realizado pela biblioteca mediastreamer2. Outras tarefas desempenhadas por esta biblioteca incluem o cancelamento de eco, o controle automático de ganho e o envio e recebimento de pacotes RTP. Os recursos de som são acessados através de dispositivos ALSA e diversos codecs tais como Speex, G.711, GSM e iLBC são suportados.

RESULTADOS E CONCLUSÕES

Neste artigo propomos uma arquitetura de *hardware* e *software* livres de um telefone VoIP. É possível, conforme apresentado, desenvolver todo o *software* do projeto utilizando bibliotecas de código aberto. Além de reduzir custos de projeto, essa abordagem proporciona flexibilidade de desenvolvimento, já que todo o código fonte é disponível e pode ser modificado conforme a conveniência do projetista.

O telefone VoIP desenvolvido foi testado em um cenário composto por um IP PBX e um telefone VoIP proprietário. O registro no servidor Proxy, a realização e recebimento de chamadas foram realizados com sucesso, validando, assim, o *hardware* e *software* da proposta.

REFERÊNCIAS

Oode B., "Voice over Internet Protocol", Proceedings of the IEEE2002, 90(9):1495.

Projeto Blackfin IP-Phone. Disponível em: http://code.google.com/p/blackfin-ip-phone-hw

M. Handley, M. Schulzrinne, E. Schooler, etc, "SIP: Session Initiation Protocol", IETF(RFC3261), June, 2002.

M. Schulzrinne, S. Casner, R. Frederick, etc, "RTP: A transport Protocol for Real-Time Application", IETF. January, 1996.