RESEARCH ARTICLE OPEN ACCESS

Implementation of Low cost VoIP Media Stream Encryption Device using Linux

Mr. Sachin C Malke[1,] Mr. Girish Talmale[2]
[1,] *Student, G.H Raisoni Collage Of Engineering*
Nagpur. Email: sachininfo123@gmail.com Mob: 8149442326
[2,] *Assistant Prof. G.H Raisoni Collage of Engineering*
Nagpur. Email:girishtalmale@gmail.com

ABSTRACT:
Our Embedded project is to design and develop a low cost feature which is based on embedded platform for VOIP media using ARM architecture. It has various features and classification algorithms and the product is to be used as VOIP media device communications platform for multiple applications areas. The design of a VoIP media stream encryption device e can be deployed between the VOICE OVER INTERNET PROTOCAL (VOIP) terminal, dedicatedly used for the encryption/de-encryption of the VoIP signal and the RTP (Real-time Control Protocol) voice packet. The encryption flow of the packet is described when the VoIP protocol is SIP (Session Initiation Protocol) and the encryption algorithm is RC4. The embedded system is designed for transferring voice calls through Internet protocol by typing IP address on TFT display and this project is for low cost phone calls, which using SAMSUNG Corporation chips as core processor.

Keywords- *ARM, VOIP, RTP, SIP*

I. INTRODUCTION

Internet telephony refers to communications services like voice, facsimile, and/or voice-messaging applications that are transported via the Internet, rather than the public switched telephone network (PSTN). The basic steps involved in originating an Internet telephone call are conversion of the analog voice signal to digital format and compression/translation of the signal into Internet protocol (IP) packets for transmission over the Internet; the process is reversed at the receiving end.

In early days the two devices can communicate through internet. The communication is in the form of text by typing it from keyboard. If the devices want to communicate through voice, we require PSTN. By using PSTN No free calls possible and installation of PSTN PBX requires extra wiring, expensive proprietary hardware. The main disadvantage of existing system is we can communicate with other device by typing the text only. We cannot communicate through voice calls.

1.1 VOIP

VoIP refers to a way to carry phone calls over an IP data network, whether on the Internet or our own internal network. A primary attraction of VoIP is its ability to help reduce expenses because telephone calls travel over the data network rather than the phone company's network.

1.2ARM architecture

A RISC-based computer design approach means ARM processors require significantly fewer transistors than typical processors in average computers. This approach reduces costs, heat and power use. These are desirable traits for light, portable, battery-powered devices including smart phones, laptops, tablet and notepad computers, and other embedded systems. A simpler design facilitates more efficient multi-core CPUs and higher core counts at lower cost, providing higher processing power and improved energy efficiency for servers and supercomputers.

1.3 RTP

Real Time Transport Protocol defines a standard packet format for delivering audio and video over the internet. Real-time Transfer Protocol (RTP) provides end-to-end delivery services for data (such as interactive audio and video) with real-time characteristics.

1.4 SIP

SIP (Session Initiation Protocol) is a signaling protocol used to create, manage and terminate sessions in an IP based network. SIP is limited to only the setup and control of sessions.

II. HARDWARE DESIGN OF VOIP

The general hardware structure of the VOIP based on ARM processor is shown in Fig1. S3C2440 processor is used as core of the hardware.

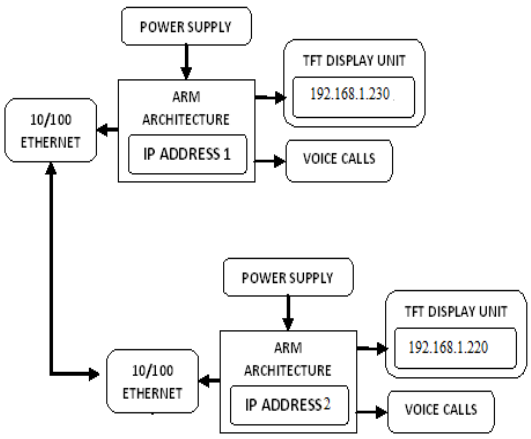

Fig1: Diagram of hardware design.

The main frequency of Samsung S3C2440 is 400MHz and can up to of a peek frequency of 533MHz. JTAG (Joint Test Action Group) is an international test protocol standard, software simulation, single-step debug and vivi-boot download can be carried out through the JTAG port, it's a simple and efficient means of developing and debugging embedded systems. The SDRAM capacity in the system is 64MB, working voltage is 3.3V, data bus is 32bit,and SDRAM clock frequency can reach up to 100MHz.Flash memory is divided into 64M NAND flash and 2M NOR flash The S3C2440 CPU chip supports two kinds of boot modes: booting at the NAND flash and booting at the NOR flash. The allocation of the storage space of the chip selections is different in the two boot modes. For supporting boot loader in the NAND Flash a buffer named Steppingstone is present in SDRAM. When the system get started, the first 4Kbyte content in NAND Flash is loaded to the Steppingstone and be executed. When Start up code, the contents of the NAND Flash are copied to the SDRAM. Main program will be executed from the SDRAM based on the completion of copy.

III. SOFTWARE DESIGN

Software development process based OS includes: the establishment of cross-compiler, the creation of root files system, the transplant of Boot loader, the porting of embedded Linux, and the development VOIP media stream. ARM Linux gcc is the cross compiler used. Boot loader vivi is used here. The function of Boot loader is to initialize the hardware devices, establish memory mapping tables, thus establish appropriate hardware and software environment, provides interface to send commands to target board and prepare for the final call to the operating system kernel. Linux is used as operating system because Linux system is having a hierarchical structure and completely opens its kernel source. Linux can port to a wide range of hardware platforms, and can run in most of the architecture. Linux has a comprehensive set of editing, debugging and other development tools, graphical interface, a powerful network supporting and rich applications. In addition, the kernel can be reduced by configuring it.

1 FLOW CHART

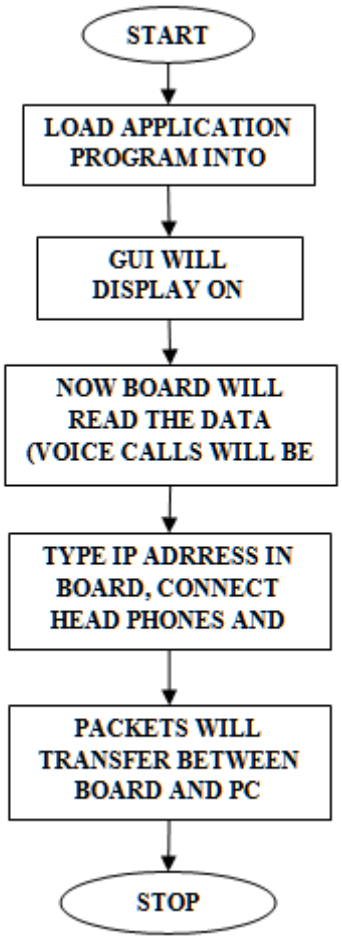

Fig2: VOIP process flow

```
#include "widget.h"
#include "ui_widget.h"
Widget::Widget(QWidget *parent) :
   QWidget(parent),
   ui(new Ui::Widget)
{
   ui->setupUi(this);
   ip = (char *) malloc (14*sizeof(char));
   index = 0;
   rate = 8000;
   this->initialize_server();
}
Widget::~Widget()
{
   delete ui;
}
void Widget::appendText(char ch) {
   if (ch == '-') {
      if (index == 0) {
      }
      else if(index > 0) {
         index--;
         ip[index] = '\0';
         ui->lineEdit->setText(QString(ip));
      }
   }
   else if (index == 15) {
   }
   else {
      ip[index] = ch;
      index++;
      ip[index] = '\0';
      ui->lineEdit->setText(QString(ip));
   }
}
int Widget::CreateDevice(int read) {
   int dummy; // just for ioctl
   int channels = 1; // 1 channel, non-stereo (mono)
   int stereo = 0; // no stereo mode (1 channel)
   int format = AFMT_U8; // 8-bit unsigned, this one works fine
   //    int format = AFMT_S16_LE; // 16-bit signed (little endian), this one produces garbled sound...
   int file = open("/dev/dsp", read ? O_RDONLY : O_WRONLY, 0);
   if (-1 == file) // check is error occurred during opening?
   {
      perror("open");
      return -1;
   }
   // we have to configure our audio device to read data in desired format
   dummy = format;
   if (-1 == ioctl(file, SNDCTL_DSP_SETFMT, &dummy) || dummy != format)
   {
      perror("ioctl SNDCTL_DSP_SETFMT");
      return -1;
   }
   dummy = channels;
   if (-1 == ioctl(file, SNDCTL_DSP_CHANNELS, &dummy) || dummy != channels)
   {
```

```
        perror("ioctl SNDCTL_DSP_CHANNELS");
        return -1;
    }
    dummy = stereo;
    if (-1 == ioctl(file, SNDCTL_DSP_STEREO, &dummy) || dummy != stereo)
    {
        perror("ioctl SNDCTL_DSP_STEREO");
        return -1;
    }
    dummy = rate;
    if (-1 == ioctl(file, SNDCTL_DSP_SPEED, &dummy) || dummy != rate)
    {
        perror("ioctl SNDCTL_DSP_SPEED");
        return -1;
    }
    return file;
}
int Widget::initialize_server() {
    int sock;
    int file = 0;
    int bytes_read;
    socklen_t addr_len;
    char recv_data[1024];
    struct sockaddr_in server_addr , client_addr;
    if ((sock = socket(AF_INET, SOCK_DGRAM, 0)) == -1) {
        perror("Socket");
        exit(1);
    }
    server_addr.sin_family = AF_INET;
    server_addr.sin_port = htons(5000);
    server_addr.sin_addr.s_addr= INADDR_ANY;
    bzero(&(server_addr.sin_zero),8);
    if (bind(sock,(struct sockaddr *)&server_addr,
        sizeof(struct sockaddr)) == -1)
    {
        perror("Bind");
        exit(1);
    }
    addr_len = sizeof(struct sockaddr);
    printf("\nUDPServer Waiting for client on port 5000");
    fflush(stdout);
    file = CreateDevice(0);
        if (-1 == file)
        {
            return EXIT_FAILURE;
        }
        pid_srv = fork ();
        if (pid_srv == 0) {
            while (1)
            {
              bytes_read = recvfrom(sock,recv_data,1024,0,
                        (struct sockaddr *)&client_addr, &addr_len);
              recv_data[bytes_read] = '\0';
              int result = write(file, recv_data, bytes_read);
            }
        }
}
void Widget::on_pushButton_2_clicked()
```

```
{
   this->on_pushButton_3_clicked();
   char *str = (char *)malloc (10*sizeof(char));
   sprintf (str,"kill %d",pid_srv);
   system (str);
   free (str);
   exit (0);
}
void Widget::on_b_1_clicked()
{
   this->appendText('1');
}
void Widget::on_b_2_clicked()
{
   this->appendText('2');
}
void Widget::on_b_3_clicked()
{
   this->appendText('3');
}
void Widget::on_b_4_clicked()
{
   this->appendText('4');
}
void Widget::on_b_5_clicked()
{
   this->appendText('5');
}
void Widget::on_b_6_clicked()
{
   this->appendText('6');
}
void Widget::on_b_7_clicked()
{
   this->appendText('7');
}

void Widget::on_b_8_clicked()
{
   this->appendText('8');
}
void Widget::on_b_9_clicked()
{
   this->appendText('9');
}
void Widget::on_b_0_clicked()
{
   this->appendText('0');
}
void Widget::on_b_dot_clicked()
{
   this->appendText('.');
}
void Widget::on_b_back_clicked()
{
   this->appendText('-');
}
void Widget::on_pushButton_clicked()
{
   int sock;
   int file = 0;
   struct sockaddr_in server_addr;
```

```
struct hostent *host;
int size = 160;
char send_data[size];
QString str;
str = ui->lineEdit->text();
std::string stdstr = str.toStdString();
host= (struct hostent *) gethostbyname( stdstr.c_str() );
if ((sock = socket(AF_INET, SOCK_DGRAM, 0)) == -1)
{
perror("socket");
exit(1);
}
server_addr.sin_family = AF_INET;
server_addr.sin_port = htons(5000);
server_addr.sin_addr = *((struct in_addr *)host->h_addr);
bzero(&(server_addr.sin_zero),8);
    pid_cli = fork ();
    if (pid_cli == 0) {
        file = CreateDevice(1);
        if (-1 == file)
        {
            //return EXIT_FAILURE;
        }
    while (1)
    {
          int ret = read (file,send_data,size);
       sendto(sock, send_data, ret, 0,
            (struct sockaddr *)&server_addr, sizeof(struct sockaddr));
    }
    }
}
void Widget::on_pushButton_3_clicked()
{
   char *str = (char *)malloc (10*sizeof(char));
   sprintf (str,"kill %d",pid_cli);
   system (str);
   free (str);
}
```

IV. TEST RESULTS

To prove the validity of the encryption process, the packet capture is done before and after the encryption, as shown in Fig. 2. Fig 2 explains how to send voice calls from board to PC. By typing IP address on touch screen display unit we can make a call. Through headphones we can talk via Ethernet.

V. CONCLUSION

Design of a VoIP media stream encryption device based on ARM9 CPU and embedded Linux is presented in this letter, which can be deployed between the Soft Switch or IP-PBX and the VoIP terminal. The VoIP encryption device can execute encryption /de-encryption by cooperate with VoIP terminals, and make VOIP terminal factories dedicatedly research of VOIP technology itself rather than encrypt scheme design and its updating. The design in this paper can be optimized in terms of multiple signaling and RTP steam encryption according to the changing of the memory capacity and running speed, and takes on a broad prospect.

REFERENCES

[1] D.Richard Kuhn, "Security Consideration for Voice Over IP Systems," NIST Special Publication 800-58.
[2] David Endler, "Hacking Exposed VoIP: Voice Over IP Security Secrets & Solutions," McGraw-Hill Osborne Media, 1 edition, Nov. 28, 2006.
[3] http://www.micrel.com KS8695X _Datasheet.
[4] http://www.micrel.com R1.1 ProgrammingGuide.
[5] X.Lai, "On the Design and Security of Block Ciphers," ETH Series in Information Processing, vol.1, Konstanz:Hartung_Georre Verlag, 1992.

[6] A.F.Webster and S.E.Tavares, "On the Design of S-box. Advances in Cryptology-Crypto 85," Springer-Verlag, pp.524~5341985.
[7] Daniele Rizzetto et al, "A voice over IP service Architecture for integrated Communication", IEEE Networks, May/June, 1999, pg. 34-40.
[8] Bill Douskalis, "IP Telephony – The Integration of Robust VoIP Services", Prentice Hall PTR 2000.
[9] Bur Goode, "Voice over Internet Protocol (VoIP)", Proceedings of the IEEE, Vol.90, No.9, September 2002, pg. 1495-1517
[10] Oliver Hersent et al, "IP Telephony – Packet based Multimedia Communication System", Addison Wesley 2000.