International Journal of Computer Science and Network (IJCSN)
Volume 1, Issue 3, June 2012 www.ijcsn.org ISSN 2277-5420

Physical modeling of musical synthesiser

[1]Ms.Sofia K.Pillai, [2]Mrs.J.J.Shah

[1] Department of CSE
GHRCE,
Nagpur,Maharashtra,India

[2] Department of CSE
GHRCE,
Nagpur,Maharashtra,India

Abstract

This paper describes the approach of physical modeling of a string musical synthesizer, a keyboard. Physical modeling uses numerical computation to create audio sounds.

The different synthesizer technology used analog synthesis, frequency modulation, additive synthesis, sampling and physical modeling. This paper deals with the hardware implementation of such a technique with the use of APR 6016. Finite difference models have been used in a couple of different settings with regards to physical modeling. Finite difference models are done by replacing derivatives in physical systems with finite differences.

Keywords: - *Physical Modeling synthesis; Finite Difference Method; APR 6016*

1.Introduction

Sound synthesis is a method to create and transform sound using mathematical techniques. The success of physical modeling in synthesizing strings is that the sound generation process in these instruments can be described as an excitation of a transmission line or waveguide. There are different synthesizers available today in the market. The mathematical equations describing such structures are rather simple (wave equations) and straightforward to implement computationally. Each technique is based on a certain model capable of representing and generating a class of sounds. We can distinguish two types of models: On the one hand, there are models that describe the sound phenomenon or commonly referred as the sound signal. On the other hand, models are in use that describes the production mechanism of sound, which is usually a physical process and therefore these models are generally referred to as physical models. For the composer the main difference between the two types lies in the way the models are parameterized.

2.Working of a keyboard

The keyboard without a matrix circuitry, consisting of either 61 key keyboards with 62 wires connected to IO or keyboard with matrix.61 key keyboard can send signals to IC with 16 wires .Keyboard matrix consists of 8 X 6 wires with a mechanical switch at every section. The electronic or digital keyboard controller scans all columns if a key has been pressed. If a key in column has been pressed then it scans the row and then plays the key note. The keyboard can also be implemented using an IC with the different sound generated at the output using variable delay.

3.Physical modeling techniques

The physical modeling techniques can be divided into four groups namely waveguide synthesis, finite element methods, modal synthesis methods, and banded waveguides. This proposed work will be implemented using Finite Difference Method. While sometimes the instruments modeled are electronic gizmos like analog synthesizers the challenge is to model the tonal characteristics produced by acoustic instruments, including all of their performance gestures . Although physical modeling was not a new concept in acoustics and synthesis, having been implemented using finite difference approximations of the wave equation, it was not until the development of the Karplus-Strong algorithm, the subsequent refinement and generalization of the algorithm into the extremely efficient digital waveguide synthesis.

4.Logic for keyboard implementation

```
int getKeyNumber()
{
    cbi(row_port,row1);
    sbi(row_port,row2);
    sbi(row_port,row3);
    sbi(row_port,row4);
    column_port = 0xff;
    if(bit_is_clear(column_pin,c1))
    {
    return 1;
    }
    else if(bit_is_clear(column_pin,c2))
    {
    return 2;
    }
    else if(bit_is_clear(column_pin,c3))
    {
    return 3;
    }
    else if(bit_is_clear(column_pin,c4))
    {
    return 4;
    }
    else if(bit_is_clear(column_pin,c5))
    {
    return 5;
    }
    else if(bit_is_clear(column_pin,c6))
    {
    return 6;
    }
    else if(bit_is_clear(column_pin,c7))
    {
    return 7;
    }
    else if(bit_is_clear(column_pin,c8))
    {
    return 8;
    }
```

This scanning of rows will be implemented for four rows.

5.Use of APR 6016 Sound IC

The APR 6016 is Multi-level analog storage high quality is audio recording and playback IC with 16 minutes of recording and playback. It has dual mode storage of analog and/or digital data and eliminates the need for separate digital memory. It supports SPI interface thus allowing any commercial micro-controller to control the device. The APR6016 offers non-volatile storage of voice and/or data in advanced Multi-Level Flash memory. Up to 16 minutes of audio recording and playback can be accommodated. A maximum of 30K bits of digital data can be stored. APR6016 devices can be cascaded for longer duration recording or greater digital storage. Device control is accomplished through an industry standard SPI interface that allows a microcontroller to manage message recording and playback. This flexible arrangement allows for the widest variety of messaging options. The APR6016 is ideal for use in cellular and cordless phones, telephone answering devices, personal digital assistants, personal voice encoders and voice pagers. Single 3V power supply

- Low power consumption
- Playback operating current: 15mA typical
- Standby current: 1uA maximum
- Automatic power-down

6.Finite difference method

Finite difference models have been used in a couple of different settings with regards to physical modeling. Acoustics, specifically evaluating the dynamical behaviors of musical instruments. Piano strings, bar percussion instruments, square plate instruments, and the kettledrum are some of the instruments that have been studied using finite difference methods. One way is in the study of the

other situation is actual simulation of musical instruments. Finite differencing of string equations has been studied for many years, and was used to construct the first known digital model of vibrating strings in the 1970' Interest in finite element methods saw an increase in the mid-1990's in the context of a one-dimensional wave-equation Finite difference models are done by replacing derivatives in physical systems with finite differences. There are a couple ways this has been accomplished. One way consists of an approximation of the first partial derivative with respect to times.

$y(t,x) \approx [y(t,x)-y(t-T,x)]/T \approx [y(t+T,x)-y(t-T,x)]/2T.$

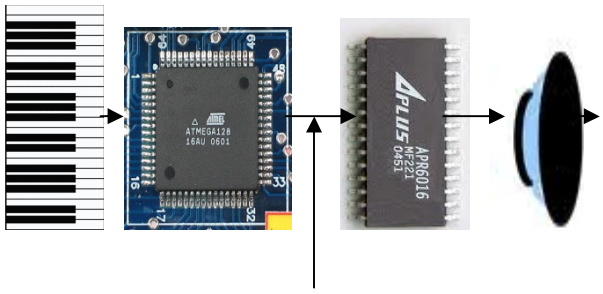
In this equation, represents time in seconds, is the position along the string, and *T* is the sampling period.

A variation is called centered finite difference. It requires an extra factor of two over sampling in its magnitude response for a given accuracy; however it holds the advantage of not introducing a time delay.

In this method samples of sound signals will be considered .By applying the Finite difference method difference between two samples will be taken. These samples will then be compared .If the value of the difference between the two samples is less than 1 then it will be given as the output otherwise it will be checked.

7.Architecture

The proposed system architecture is as given below:



Keyboard AVR Atmega APR 6016 Speaker

The keyboard will be used as the Input device .Whenever a key is pressed the sound will be generated. This sound signal will be passed on to the AVR At mega. It will be processed and sent to the APR 6016 via SPI Interface. The keys in the AVR will have different function .when first key will be pressed an option will be given

8.Implementation

```
/* vi:set et ts=4 sw=4 ai ft=c ff=dos: */

#include "MusicalInstruments.h"
#include "avr/delay.h"
#include "UART.h"
#include "LCD_4Bit.h"
#include <compat/deprecated.h>

void play(void);
void record(void);
void pwrup(void);
void one(void);
void zero(void);
void stop(void);

#define APR_DDR DDRD
#define APR_PORT PORTD
#define APR_PIN PIND
#define SCK 2
#define DI 3
#define CS 4
#define SAC 5

#define DATA_ONE sbi(APR_PORT,DI);
#define DATA_ZERO cbi(APR_PORT,DI);
#define SCK_ONE sbi(APR_PORT,SCK);
#define SCK_ZERO cbi(APR_PORT,SCK);
#define CS_ONE sbi(APR_PORT,CS);
#define CS_ZERO cbi(APR_PORT,CS);

#define KEY_PORT PORTB
#define KEY_DDR DDRB
#define KEY_PIN PINB
#define KEY1 4
#define KEY2 5
#define KEY3 6
#define KEY4 7

#define column_ddr DDRA
#define column_port PORTA
#define column_pin PINA
#define c1 0
#define c2 1
#define c3 2
#define c4 3
#define c5 4
#define c6 5
#define c7 6
#define c8 7

#define row_ddr DDRB
#define row_port PORTB
#define row1 0
#define row2 1
#define row3 2
```

```
#define row4 3

int getKeyNumber(void);

int main(int argc, char *argv[])
{
	char ch;
	//////////////////INIT THE SYSTEM/////////
	uartInit();
	lcd_init();
	sbi(APR_DDR,SCK);
	sbi(APR_DDR,DI);
	sbi(APR_DDR,CS);
	cbi(APR_PORT,SCK);
	cbi(APR_PORT,DI);
	cbi(APR_PORT,CS);

	cbi(APR_DDR,SAC);
	sbi(APR_PORT,SAC);
	CS_ONE;
	_delay_ms(100);
	SCK_ZERO;
	pwrup();

	column_ddr = 0x00;
	sbi(row_ddr,row1);
	sbi(row_ddr,row2);
	sbi(row_ddr,row3);
	sbi(row_ddr,row4);

	column_port = 0xff;

	cbi(row_port,row1);
	cbi(row_port,row2);
	cbi(row_port,row3);
	cbi(row_port,row4);
	//////////////////////////////////////////////////////////////

	KEY_DDR = 0x0F;

	char MODE = 'N';

	while(1)
	{
		if(MODE != 'R' && MODE != 'P')
		{
			lcd_display("Press a key",LINE1);
			lcd_display("To record/play",LINE2);
		}
		KEY_PORT = 0xF0;
		if(bit_is_clear(KEY_PIN,KEY1))
		{
			lcd_display("Record Mode",LINE1);
			lcd_display("Press (1-32)",LINE2);

			MODE = 'R';

			//Now store the sound from the PC

	while(bit_is_clear(KEY_PIN,KEY1));

		}
		else if(bit_is_clear(KEY_PIN,KEY2))
		{
			lcd_display("Play Mode",LINE1);
			lcd_display("Press (1-32)",LINE2);

			MODE = 'P';


	while(bit_is_clear(KEY_PIN,KEY2));
		}

		if(MODE == 'R')
		{
			if(getKeyNumber() != -1)
	{
		lcd_display("Sending Command",LINE2);
				uartPutc('R');

	uartPutc(getKeyNumber());
				ch = receiveByte();

				record();

	lcd_display("File Recorded",LINE2);


	while(getKeyNumber() != -1);
			}
		}

		if(MODE == 'P')
```

```
                {
                        if(getKeyNumber() != -1)
                        {
                        //Now play the sound via the
PC              lcd_display("Sending
Command",LINE2);
                                uartPutc('P');

        uartPutc(getKeyNumber());
                                ch = receiveByte();

                                play();

                lcd_display("File Played",LINE2);


        while(getKeyNumber() != -1);
                                }
                }
        }

        return 0;
}

void one(void)
{
        _delay_ms(1);
        DATA_ONE;
        SCK_ZERO;
        _delay_ms(1);
        SCK_ONE;
}
void zero(void)
{
        _delay_ms(1);
        DATA_ZERO;
        SCK_ZERO;
        _delay_ms(1);
        SCK_ONE;
}
void play(void)
{
    unsigned char i;

    CS_ZERO;
    zero();
    one();
    one();
    zero();
    zero();
    zero();
    zero();
    zero();
    zero();
    zero();
    zero();
    zero();
    zero();
    zero();
    zero();
    zero();
    zero();
    zero();
    CS_ONE;


    for(i=0;i<20;i++)
    {
      while(bit_is_set(APR_PIN,SAC)){}
      CS_ZERO;}}
```

9.results

The result given in the graphical form is as shown below:

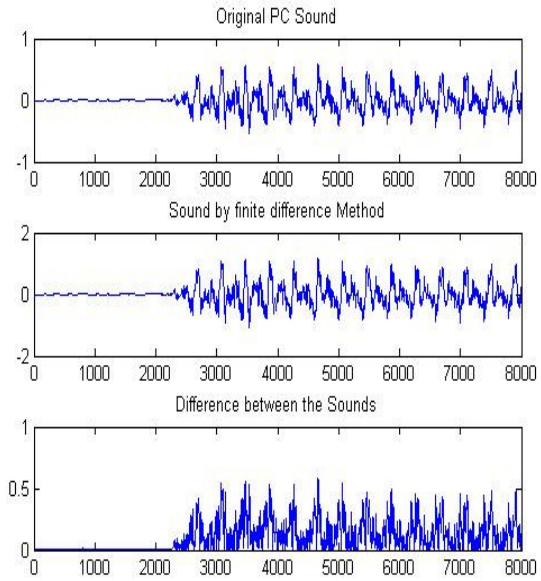


Fig.1.1 Result

The graph clearly shows the difference between the original sound sample and the sound synthesized by the microcontroller. The difference between the two sound the amplitude is double. This increases the sound quality of the synthesizer.

10.Conclusion

Thus the hardware implementation of synthesizers is presented through this paper. The sound quality is enhanced using the APR 6016 which supports 16 minutes of recording and playback. One main advantage of this synthesizer is user have a choice to record whichever sound they wish to record on a particular key and can play again .The graph clearly shows the difference between the original sound sample and the one which has been synthesized. Thus, improving the sound quality.

References

[1] M. Karjalainen, V. V¨alim¨aki, and T. Tolonen, ―Plucked String Models: from Karplus-Strong Algorithm to Digital Waveguides and Beyond, Computer Music J., vol. 22, no. 3, pp. 17-32, 1998.

[2] A. Fettweis, ―Wave Digital Filters: Theory and Practice,― Proc. of the IEEE, vol. 74, no. 2, pp. 270-327, 1986.

[3] J. Bensa, S. Bilbao, R. Kronland-Martinet, and J. O. Smith III, ―The simulation of piano string vibration: From physical models to finite difference schemes and digital waveguides, J. Acoust. Soc. Am., vol. 114, no. 2, pp. 1095–1107, 2003.

[4] K. Karplus and A. Strong, ―Digital synthesis of lucked string and drum timbres, Computer Music Journal, vol.7, no. 2, pp. 43–55, Jun. 1983.

[5] J. O. Smith, ―Physical modeling using digital waveguides, Computer Music Journal, vol. 16, no. 4, pp. 74–91, 1992