A律13折线PCM编码转换器

# 代码

#include <iostream>

#include <iomanip>

#include <cmath>

#include <algorithm>

#include <vector>

#include <string>

using namespace std;

/\*\*

 \* A律13折线编码器

 \*

 \* 输入：±1内的浮点数

 \* 输出：8位PCM码组

 \*/

class A13Coder {

protected:

    vector<uint8\_t> table;  // 输入电压到PCM码组的转换表

public:

    A13Coder() {

*this*->buildTable();  // 建立输入电压到PCM码组的转换表

    }

    uint8\_t convert(double rawInput) {

        double voltInput = *this*->limitRange(rawInput);  // 限制输入电压范围

        int normInput = *this*->normalize(voltInput);  // 将输入电压转换为量化单位

        uint8\_t sign = normInput > 0 ? uint8\_t(1 << 7) : 0;  // 根据符号获得极性码

        uint8\_t para = *this*->table[abs(normInput)]; // 查表获得段落码和段内码

        return sign + para;

    }

private:

    // 建立输入电压到PCM码组的转换表

    void buildTable() {

        table = move(vector<uint8\_t>(2048 + 1));

        vector<int> interval = {  // 量化电平共计8段，每段内间隔如下

            1, 1, 2, 4, 8, 16, 32, 64

        };

        int normInput = 0;

        uint8\_t code = 0;  // 当量化电平为0时，PCM码组为0

        for (int i = 0; i < 8; ++i) {

            for (int j = 0; j < 16; ++j) {

*this*->table[normInput] = code;

                normInput += interval[i];

                code += 1;  // 量化电平每增加一个段内间隔，PCM码组值加1

            }

        }

        // 补充表的空缺位置

        for (int i = 1; i <= 2048; ++i) {

            if (*this*->table[i] == 0) {

*this*->table[i] = *this*->table[i-1];

            }

        }

    }

    // 将输入电压限制在±1之间

    double limitRange(double volt) {

        volt = max(-1.0, volt);

        volt = min(+1.0, volt);

        return volt;

    }

    // 将输入电压化为量化单位

    int normalize(double volt) {

        return round(volt / 1 \* 2048);

    }

};

// 字符串工具

class StringUtil {

public:

    // 将数字转换为二进制串

    string toBinary(uint8\_t number) {

        string rslt(8, '0');

        for (uint8\_t i = 0; i < 8; ++i) {

            if (number & (1<<(7-i))) {

                rslt[i] = '1';

            }

        }

        return rslt;

    }

};

// 测试函数

int main() {

    StringUtil util;

    A13Coder coder;

    for (double input : {-1.0, -0.62, -0.22, -0.17, 0.0, 0.17, 0.22, 0.62, 1.0}) {

        cout << setw(5) << setprecision(2) << input << "V"

            << " => "

            << util.toBinary(coder.convert(input))

            << endl;

    }

}

# 测试用例及输出

