

**DSP应用实验报告**

实验十二：FIR滤波器的DSP实现

院 系：电子工程与光电技术学院

专 业：电子信息工程

姓 名：赵婧萱

学 号：920104330118

指导老师：李彧晟

2023年5月29日

目录

[12.1 实验目的 3](#_Toc71191846)

[12.2 实验仪器 3](#_Toc71191847)

[12.3 实验内容 3](#_Toc71191848)

[12.4 实验步骤 3](#_Toc71191849)

[12.4.1滤波器系数求解 3](#_Toc71191850)

[12.4.2 数据定标 5](#_Toc71191851)

[12.4.3 FIR计算模块设置 8](#_Toc71191852)

[12.4.4 FIR算法实现 9](#_Toc71191853)

[12.5 算法功能验证 9](#_Toc71191854)

[12.6 实际幅频特性曲线 13](#_Toc71191855)

[12.7 算法实时性验证 14](#_Toc71191856)

[12.8实验思考和问题分析 16](#_Toc71191857)

[12.9 实验总结 17](#_Toc71191858)

实验十二：FIR滤波器的DSP实现

# 12.1 实验目的

1.巩固数字 FIR 滤波器的概念

2.理解定点 DSP 中数的定标、有限字长、溢出等概念

3.理解算法实现中实时的概念

4.掌握 DSP 开发过程以及基本调试方法

# 12.2 实验仪器

计算机，TMS320F28335 DSP教学实验箱，XDS510 USB仿真器，示波器，信号源

# 12.3 实验内容

针对 FIR 算法，设计滤波器系数，完成数据的定标，查看滤波器特性曲线。建立工程，编写 DSP 的主程序，并对工程进行编译、链接，利用现有 DSP 平台实现 FIR 滤波器算法，通过信号源、示波器理解滤波器特性，验证实现与理论设计的一致性。

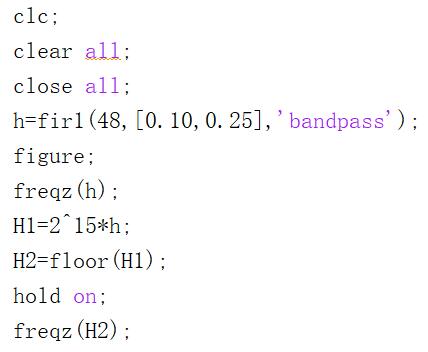
# 12.4 实验步骤

12.4.1滤波器系数求解

本实验在实验11的基础上，对采集到的数据进行滤波处理，嵌入FIR运算。我们采用Matlab软件设计滤波器系数。利用自带函数fir1（），输入相关参数即可得到滤波器系数。其完整命令如下：

其中，n为滤波器阶数，Wn为归一化截止频率（此处指与采样频率一半进行归一化），Wn对应在幅频特性曲线上-6dB点的频率数值，ftype为滤波器类型，分为低通、带通、高通、带阻形式，window为使用的窗函数，可以为hamming（汉明窗）、hanning（汉宁窗）、chebwin（切比雪夫窗）等形式；h为产生的滤波器系数。

根据实验要求，我们小组需要设计带通滤波器。设定相关参数，最终设计为一个48阶、频率范围为2kHz到5kHz的带通滤波器，采样频率设定为40kHz。Matlab设计程序如下：



对产生的滤波器系数用freqz命令查看其幅频、相频特性曲线。操作后如**图1**所示：

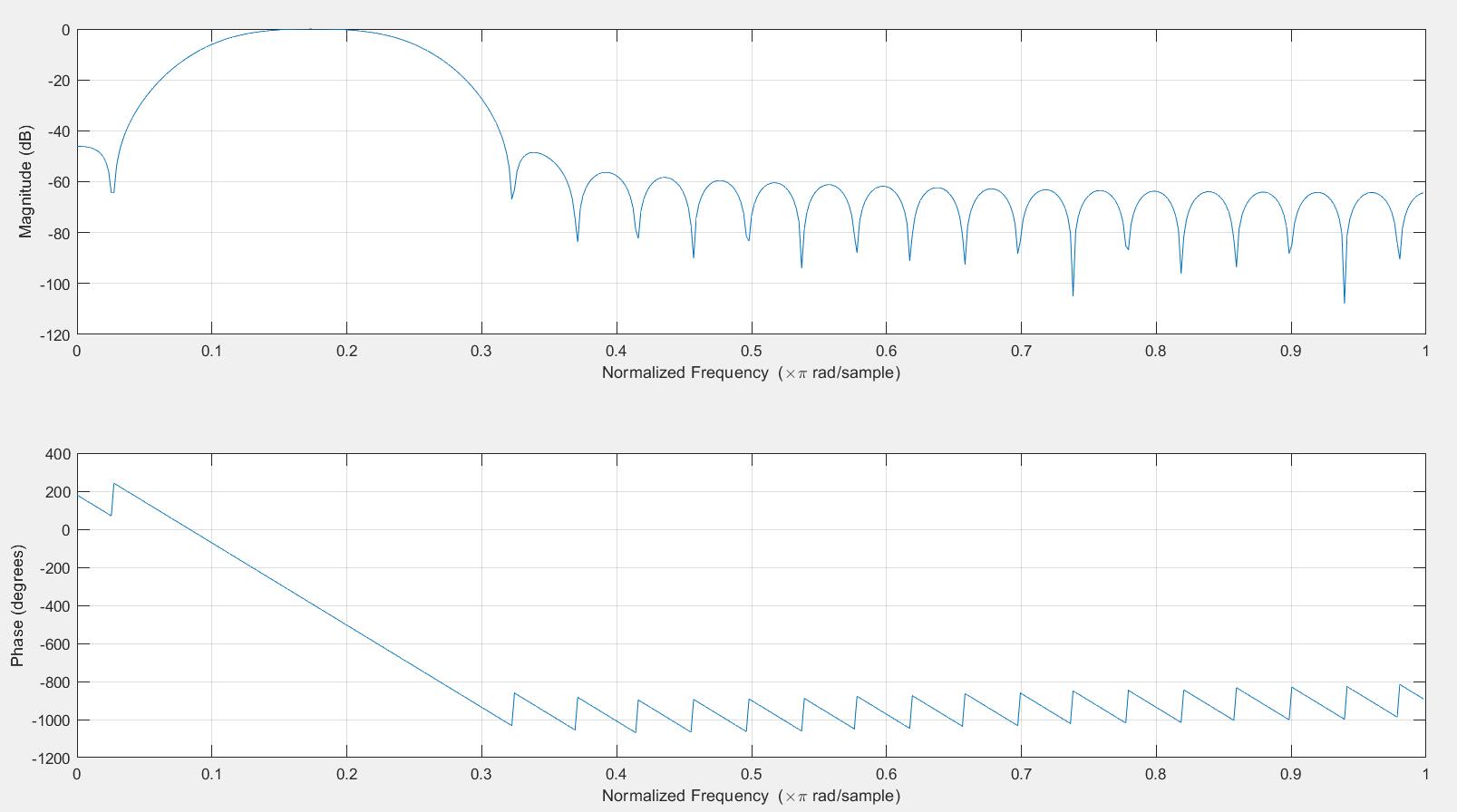


图1：带通滤波器的幅频/相频特性曲线

同时，在工作区得到滤波器各阶系数，如**表1**所示：

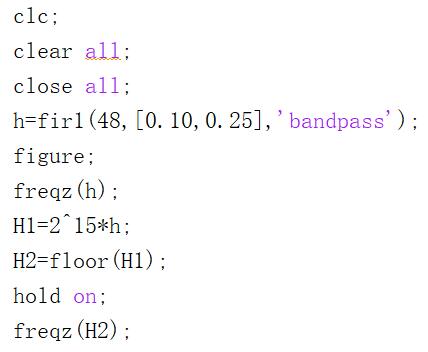
表1：滤波器各阶滤波器系数



将所得的滤波系数保存下来，为后续数据的定标修改做准备。

12.4.2 数据定标

取定标Q=15，即将数据乘，之后取整，其余的数值也做此操作。对应Matlab的命令如下：



式中，floor()函数功能为“向下取整”。通过上述系列操作得到如**表2**所示的定标后的滤波器系数；

表2：数据定标后的滤波器系数



利用freqz（）函数作出定标后系数的幅频、相频特性曲线，同时利用Matlab命令“hold on”将定标前后的特性曲线展示在同一张绘图上以便于对照。具体实现效果如**图2**所示，其中红色标线为数据定标后，由于数据扩大并且采用舍入运算截断小数，在某些特征点上会产生误差，在后续实际测试阶段可能会影响性能。

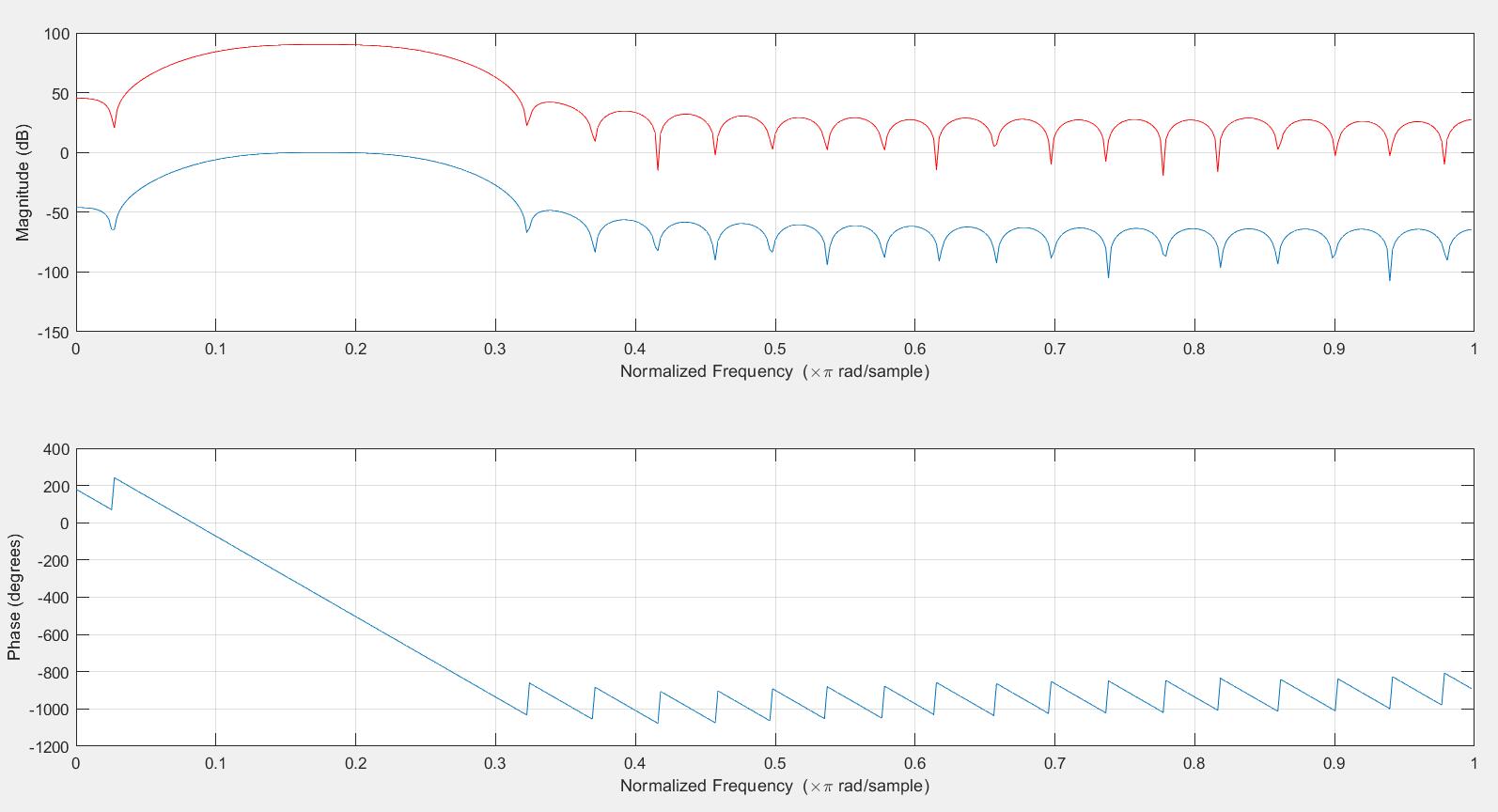


图2：数据定标后的幅频相频特性曲线

由此，我们完成了h系数从浮点数到定点数的转换。利用Matlab软件，设置函数对设计的FIR滤波器进行仿真验证。实现代码如下图所示：

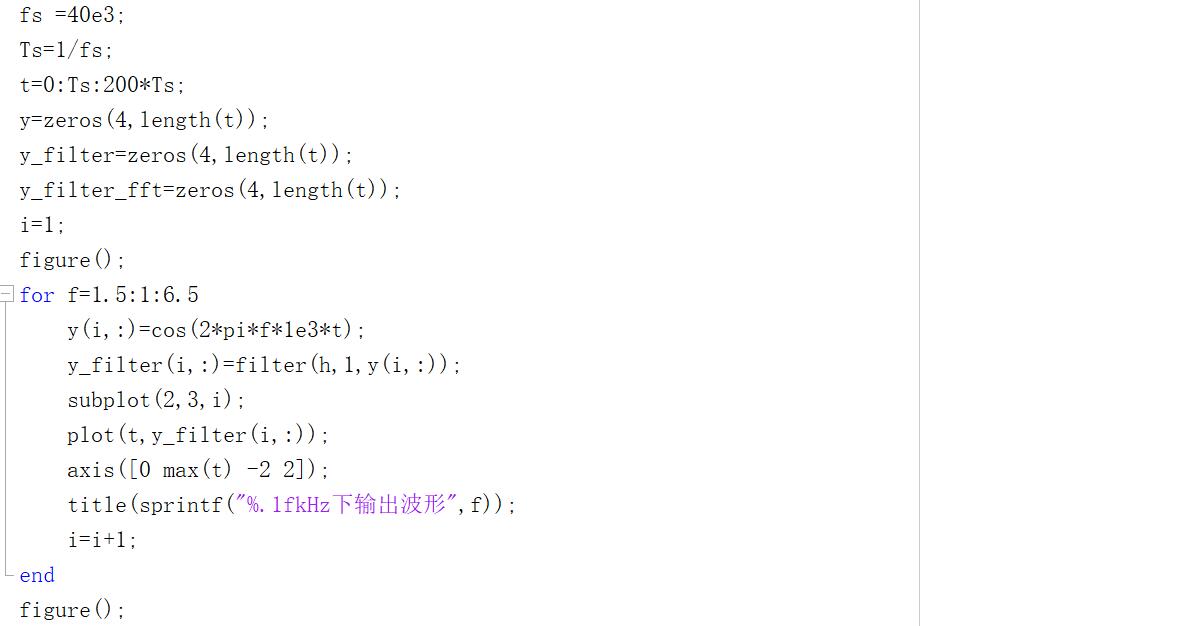


图3：Matlab验证滤波器功能

仿真后为得到特定频率下输出波形图，如下列**图4、图5**所示：

1.输入波形（频率分别对应1.5、2.5、3.5、4.5、5.5、6.5kHz）

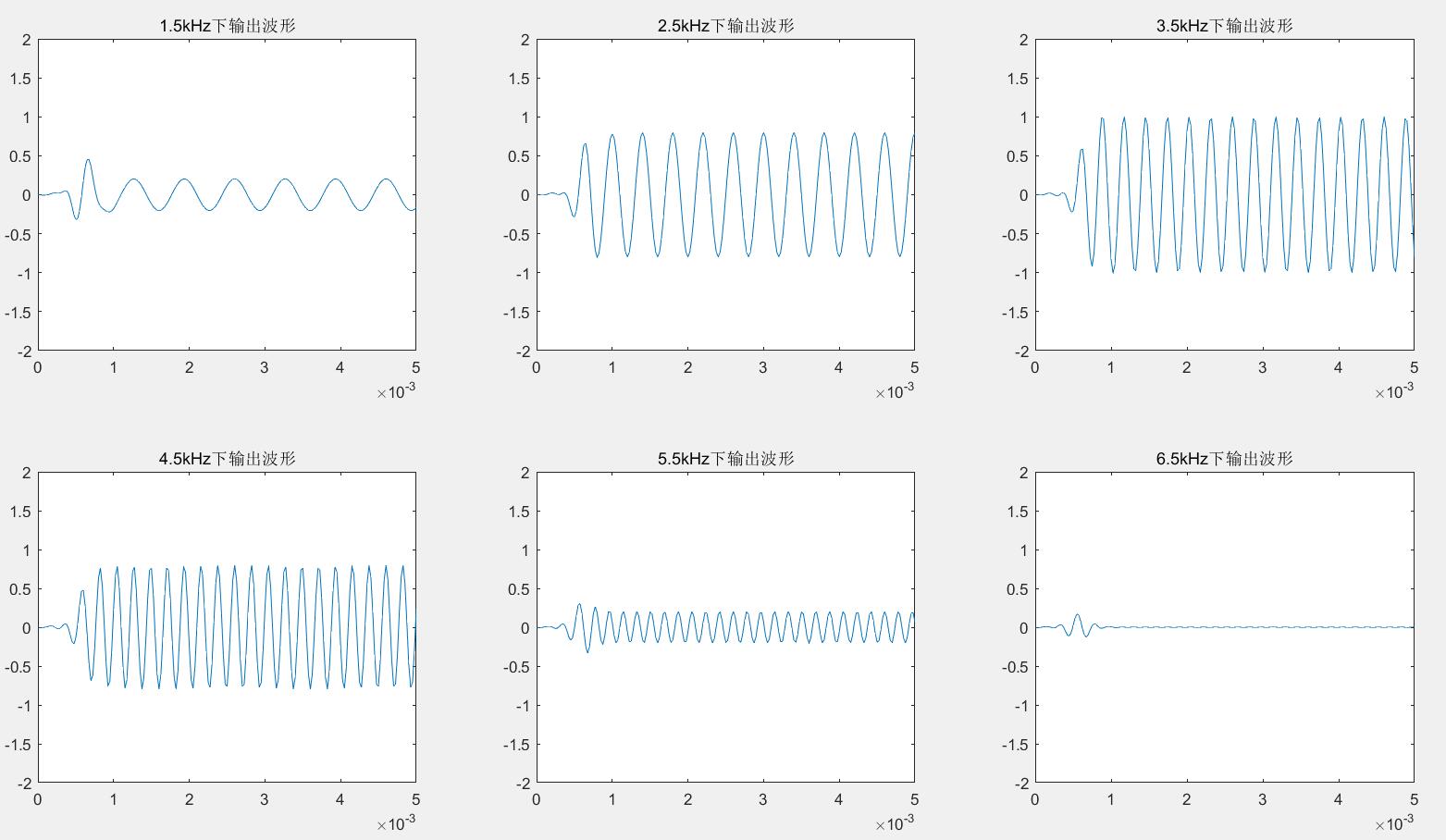


图4：Matlab仿真输入波形

2.输出滤波波形（频率分别对应1.5、2.5、3.5、4.5、5.5、6.5kHz）

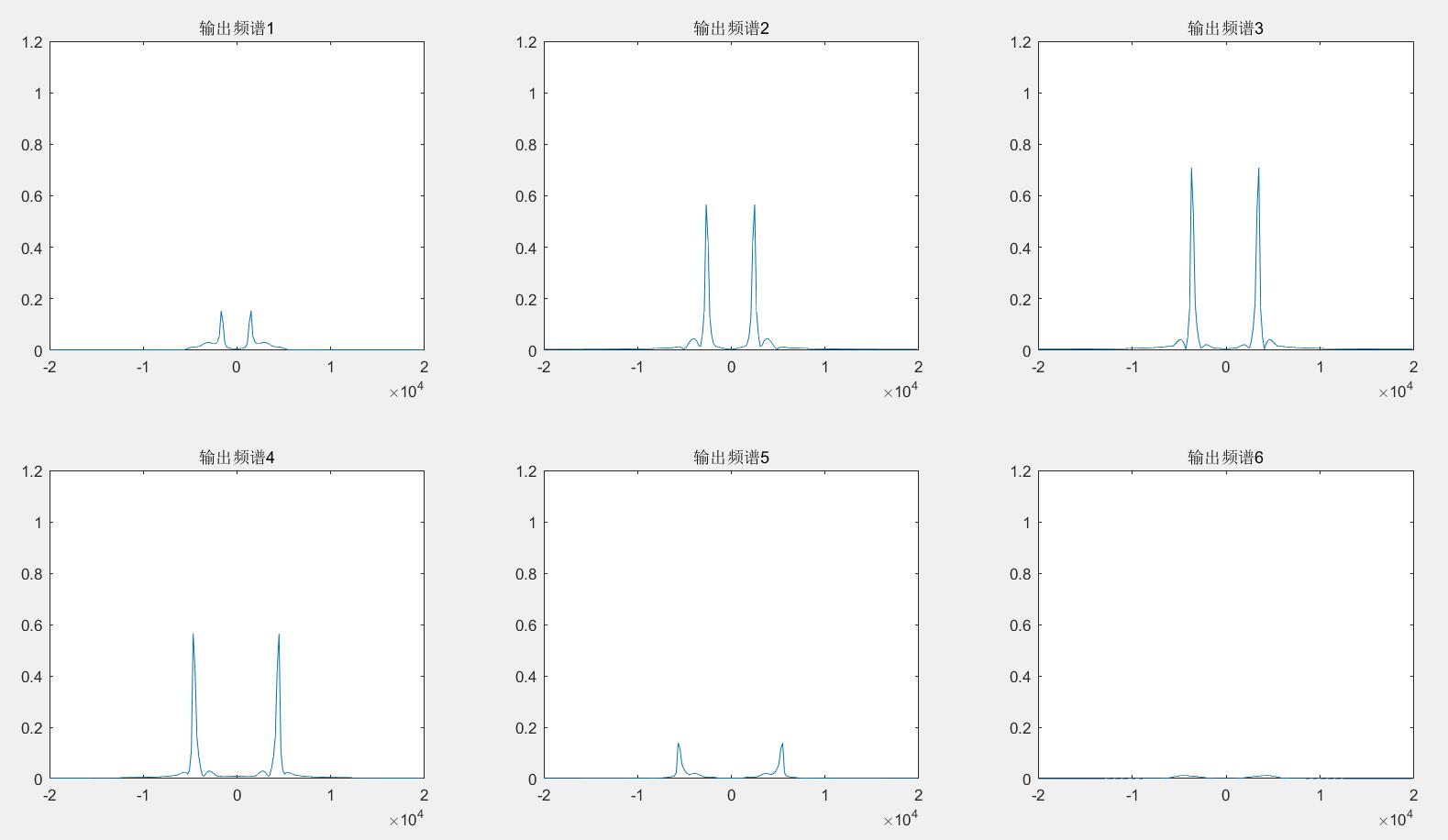


图5：Matlab仿真输出波形

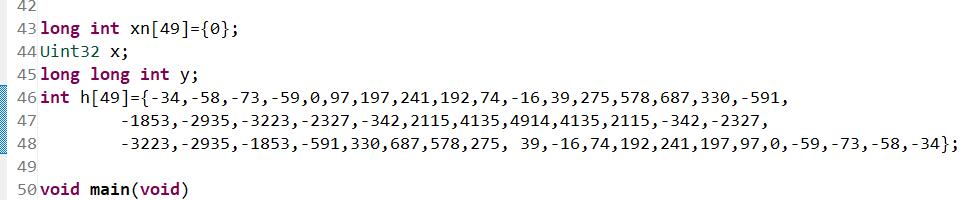
仿真模拟的结果如下：在2kHz到5kHz的通带内，正弦波输入时，可以得到正常的输出信号，随着输入频率逐步逼近截止频率，输出波形的幅度逐渐变小，超出一定范围之后，模拟输出波形的频率不再符合输入且幅度极小，将该仿真结果保留，作为后续实际验证的参照。

12.4.3 FIR计算模块设置

有限长单位冲激响应滤波器（FIR）差分方程表示为：

其中，h为滤波器系数、x为输入的数字信号、y为FIR滤波器的计算输出、N为滤波器阶数。由此可得，一个N阶滤波器计算，需要N+1个滤波器系数，N+1个数字输入；每得到一个y的值，需要N+1次乘法以及N次加法。另外，N阶滤波器需要保存当前的N+1个输入信号数值。

程序中，由于涉及到16位有符号、无符号数相乘等运算，涉及到字长的变化。x为定点数（16位，低4bit无效）、h为浮点数（16bit），二者相乘为32bit；为保证相乘运算两方均为同类型，我们将h改为定点数（即数据定标方式），则运算出y为定点数，输出给DA之前需截取16位定点数；同时考虑到数据动态范围，y的位数不能限定于32bit，于是我们设定x为unsigned int类型，y为long long int类型，xn数组为long int类型；计算模块功能如图所示：



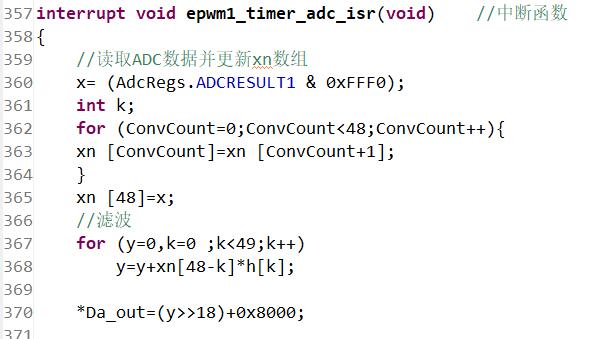


图6：FIR差分方程计算功能

12.4.4 FIR算法实现

为实现滤波器输出的实时更新，在每一个采样信号到来时，需要系统知晓该信号前48个信号的数值。构建数组xn保存最新的N个采样数，每次调整完需要删除最旧的数据，将输入信号前移（即数组后移一位），使最新采样点数据存至最后，数组始终未按时间顺序排列的最新数据，实现该功能的代码如**图7**所示：

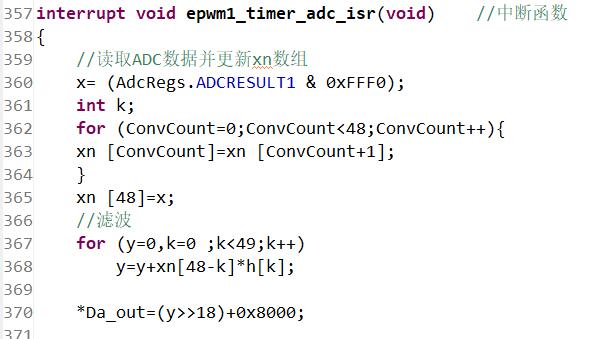


图7：输入信号更新实现代码

将上述两部分代码写入**interrupt** **void** **epwm1\_timer\_adc\_isr**(**void**)，即完成FIR滤波器代码编写。

# 12.5 算法功能验证

设置信号源输出为峰值1V、偏移500mVpp的正弦波，运行程序，逐步提高信号源输出频率，观察示波器的输出波形。如**图8-图25**所示：

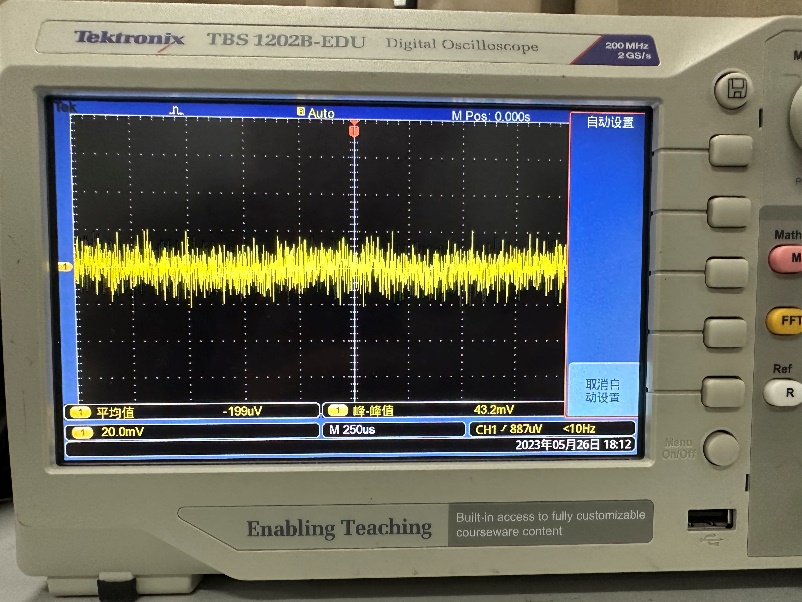
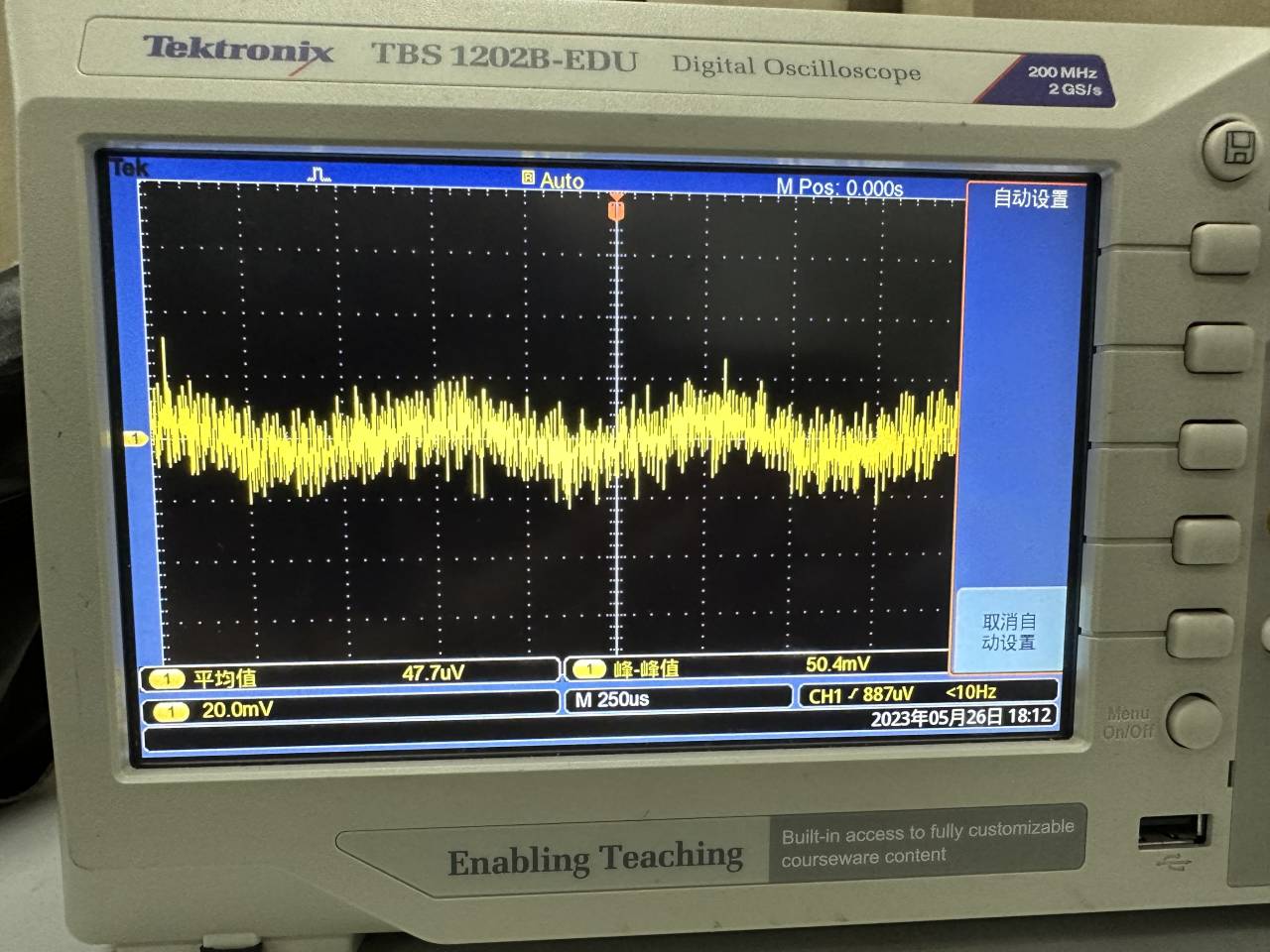
 

图8：1000Hz时的输出波形 图9：1300Hz时的输出波形

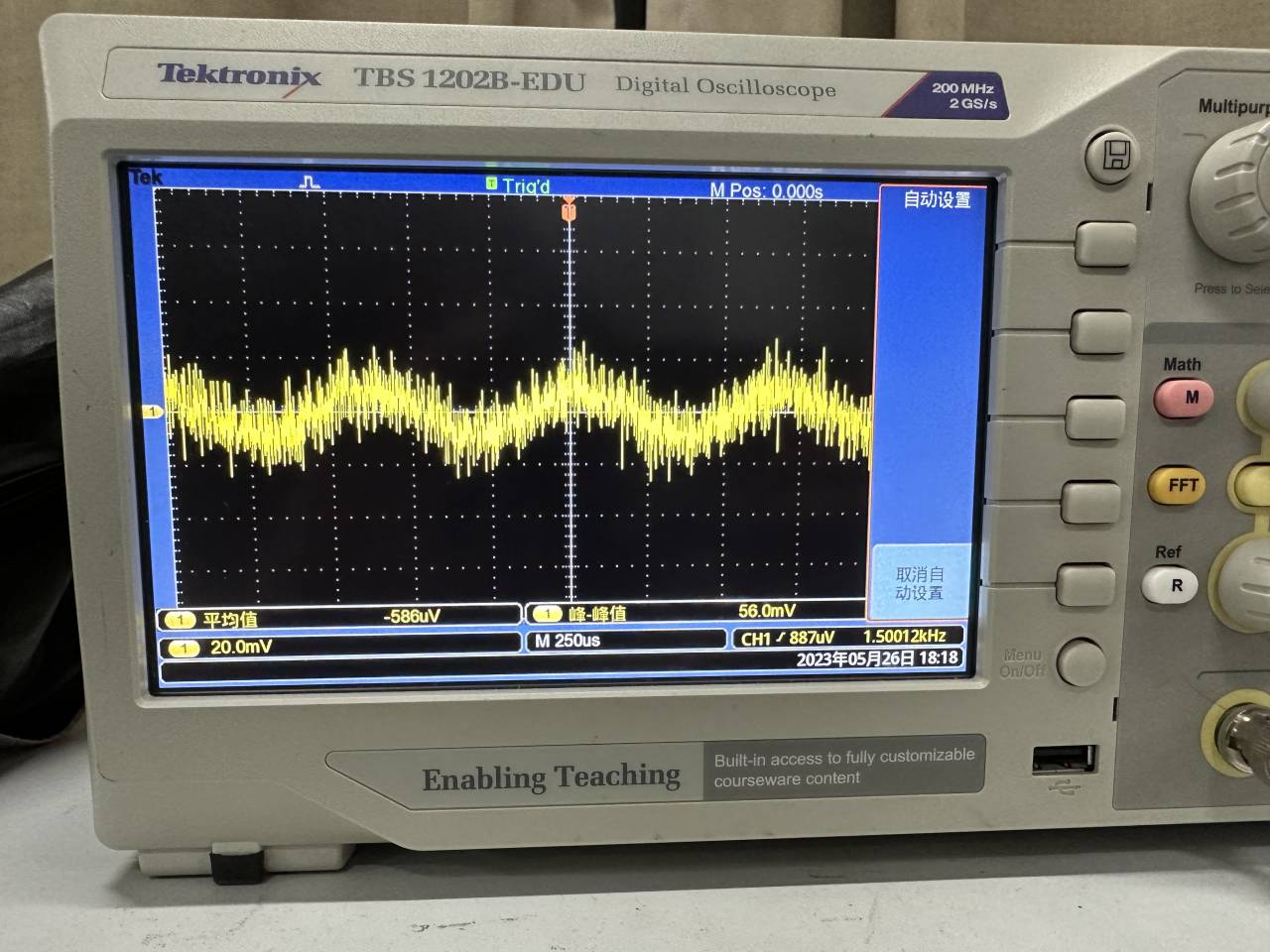
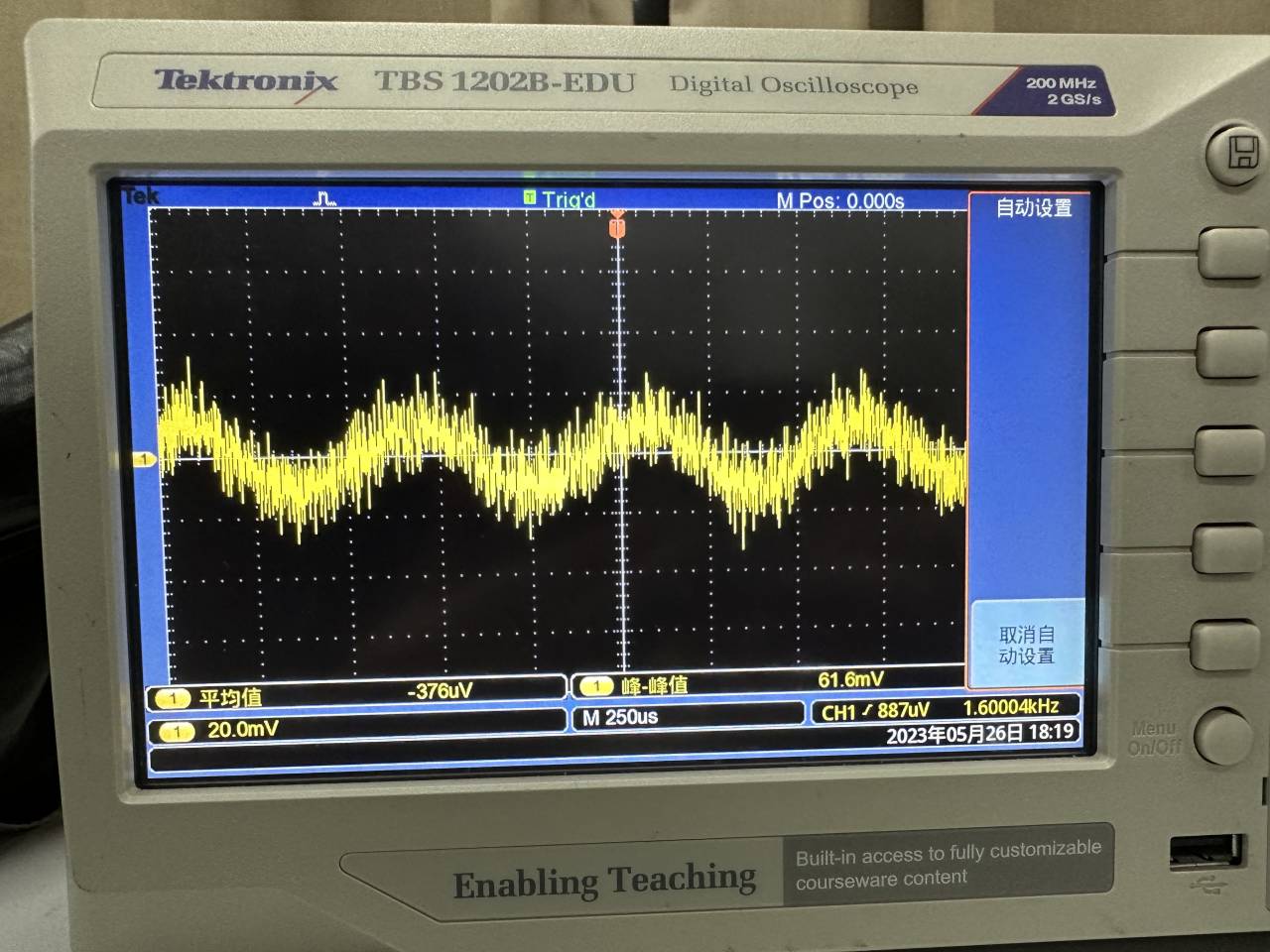
 

图10：1500Hz时的输出波形 图11：1600Hz时的输出波形

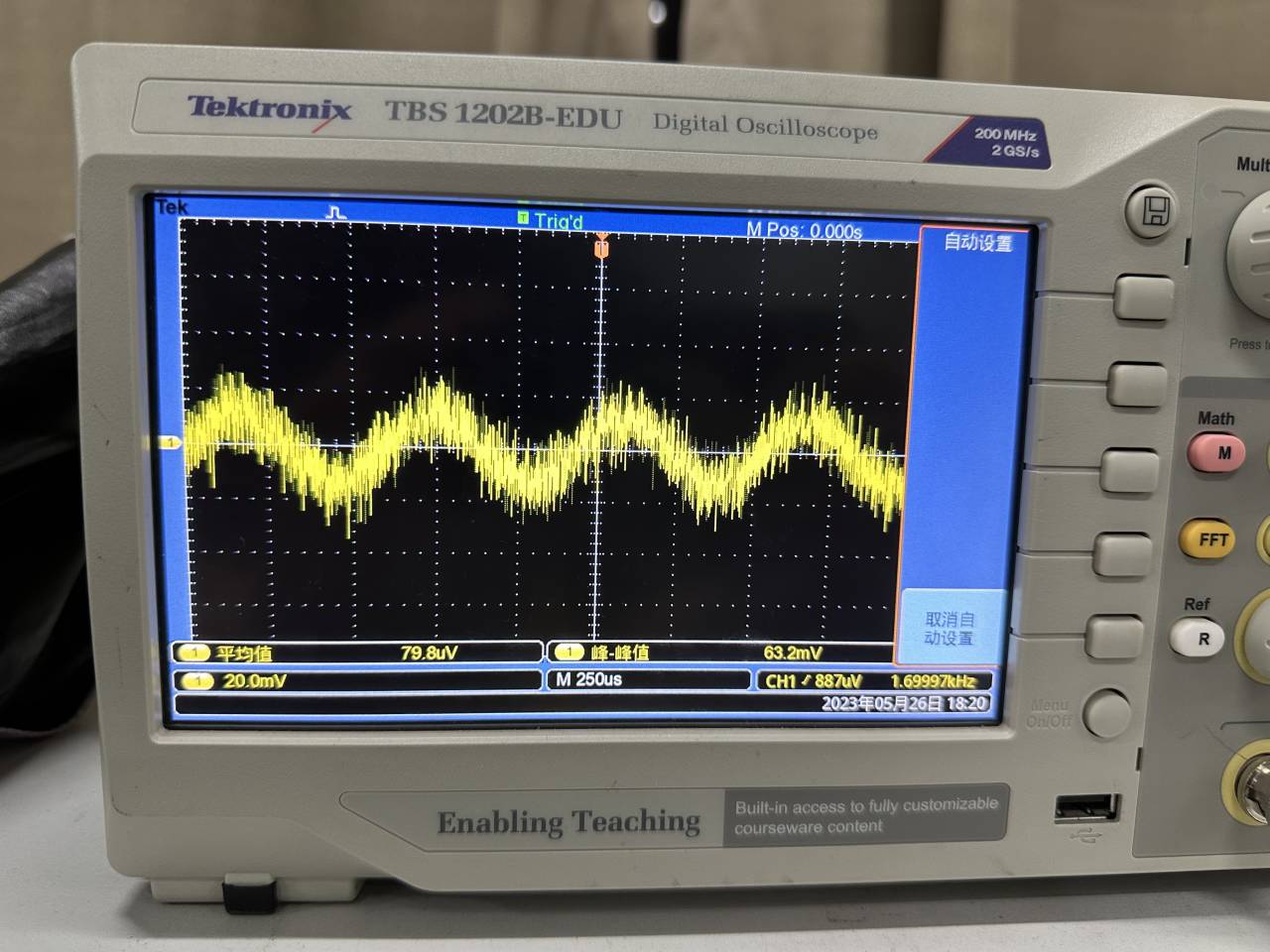
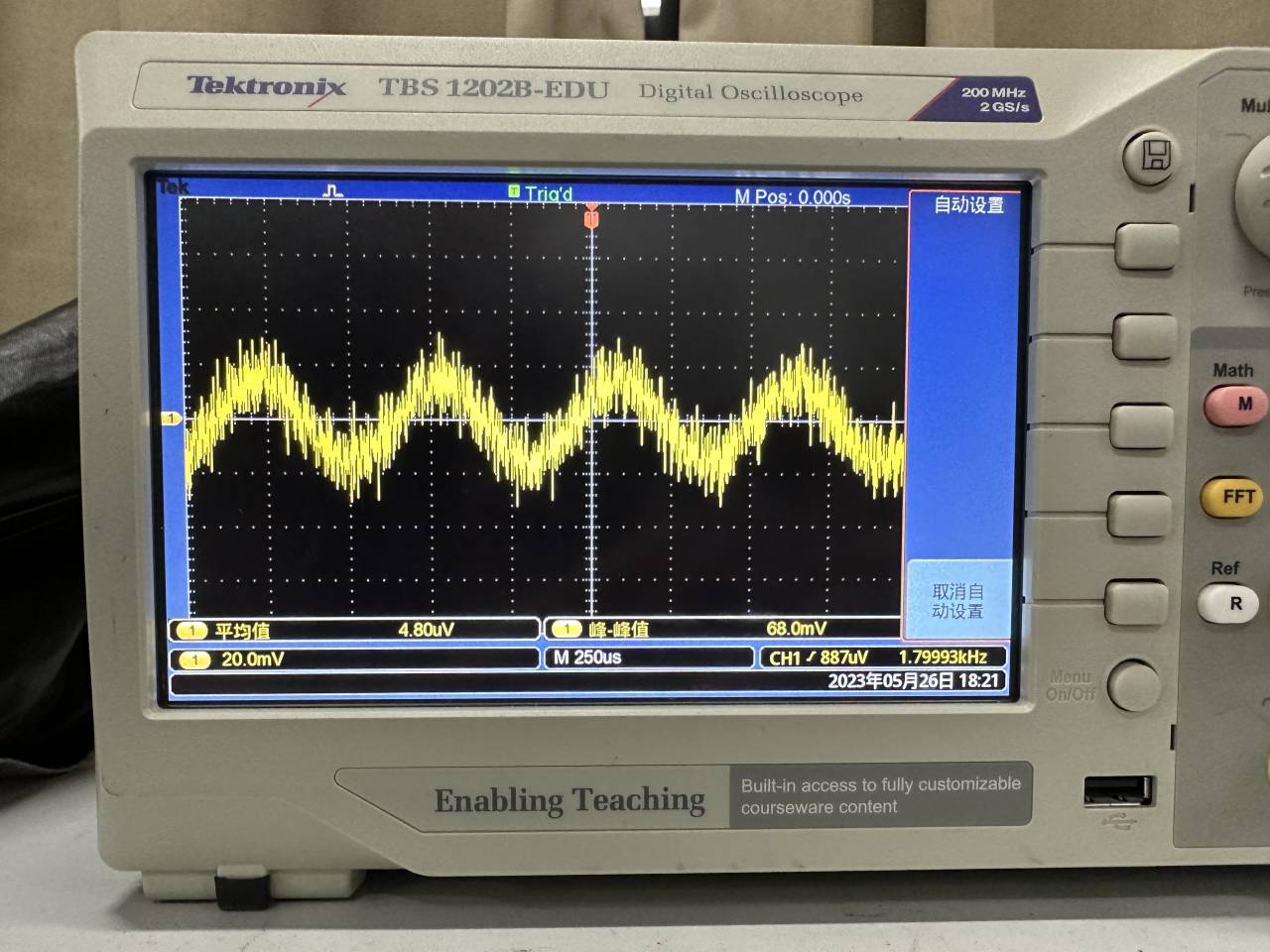
 

图12：1700Hz时的输出波形 图13：1800Hz时的输出波形

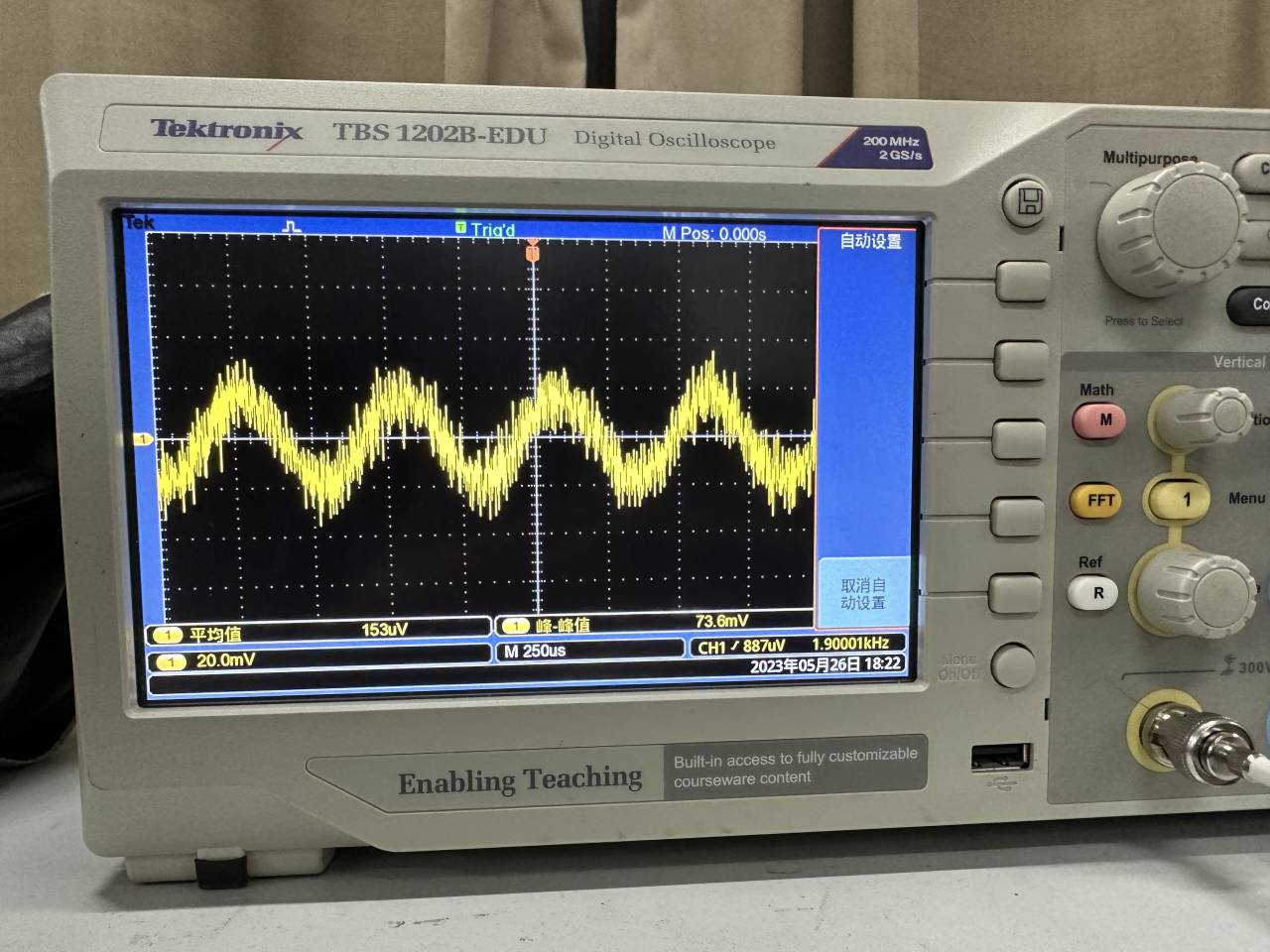
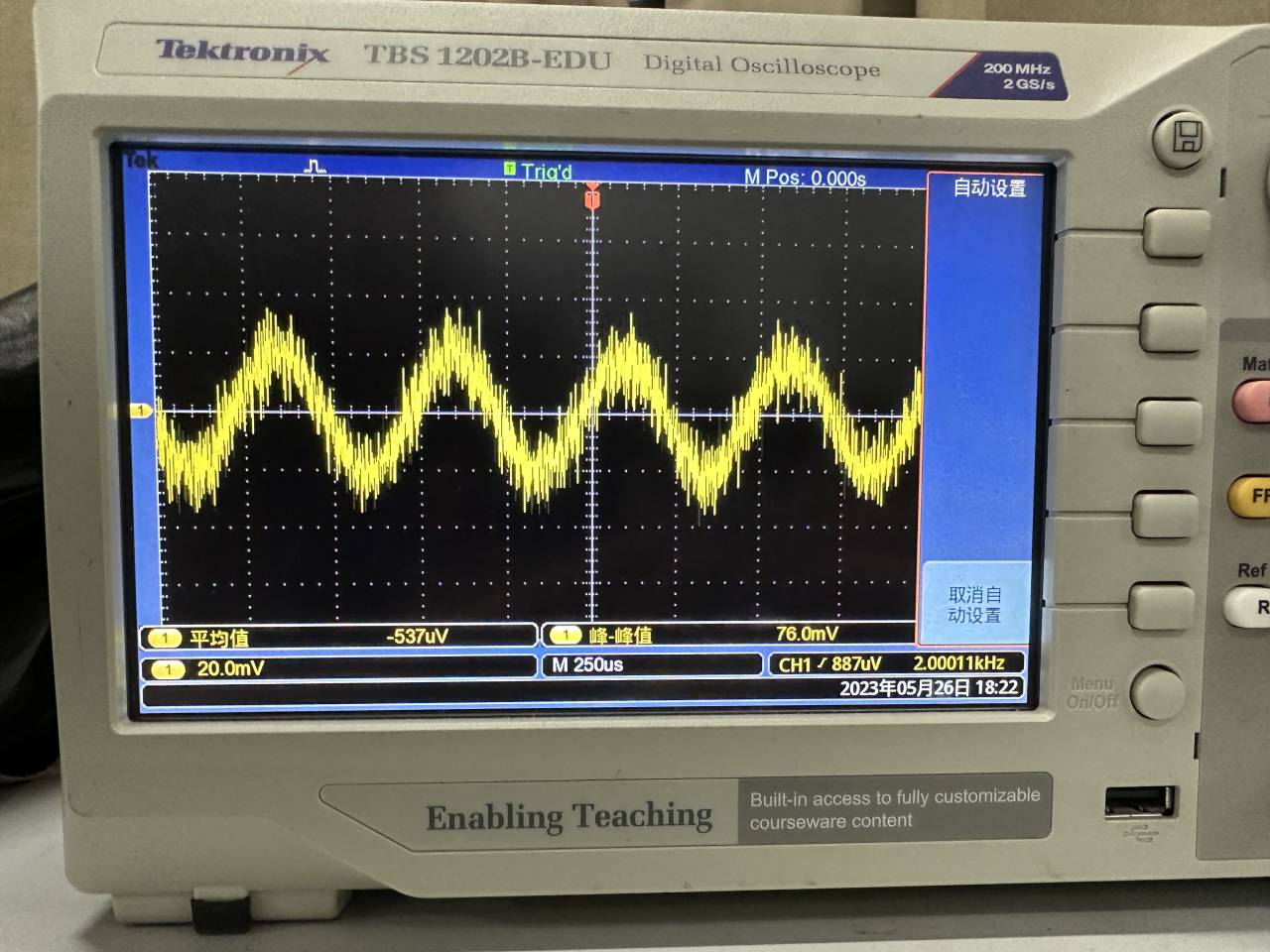
 

图14：1900Hz时的输出波形 图15：2000Hz时的输出波形

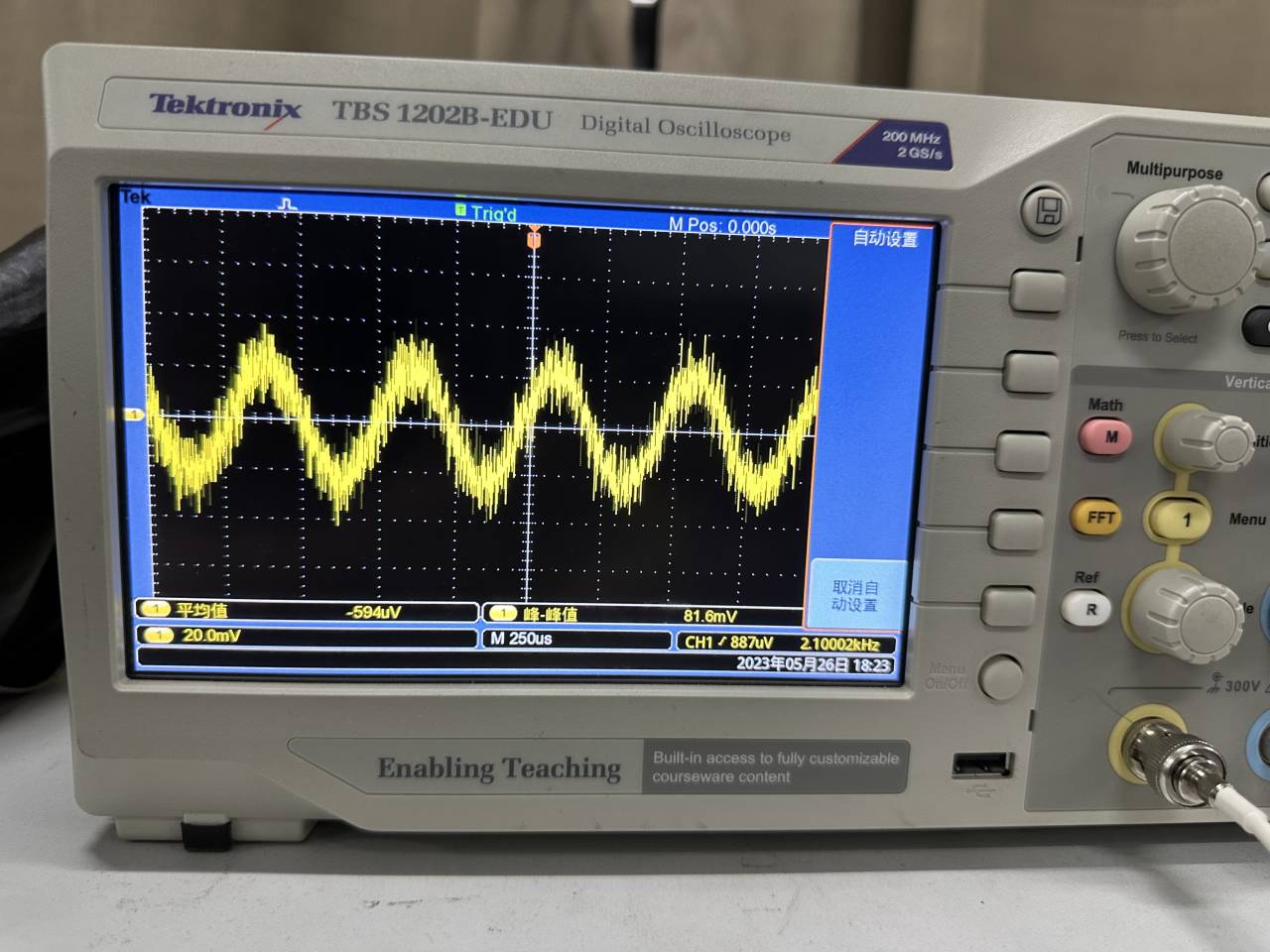
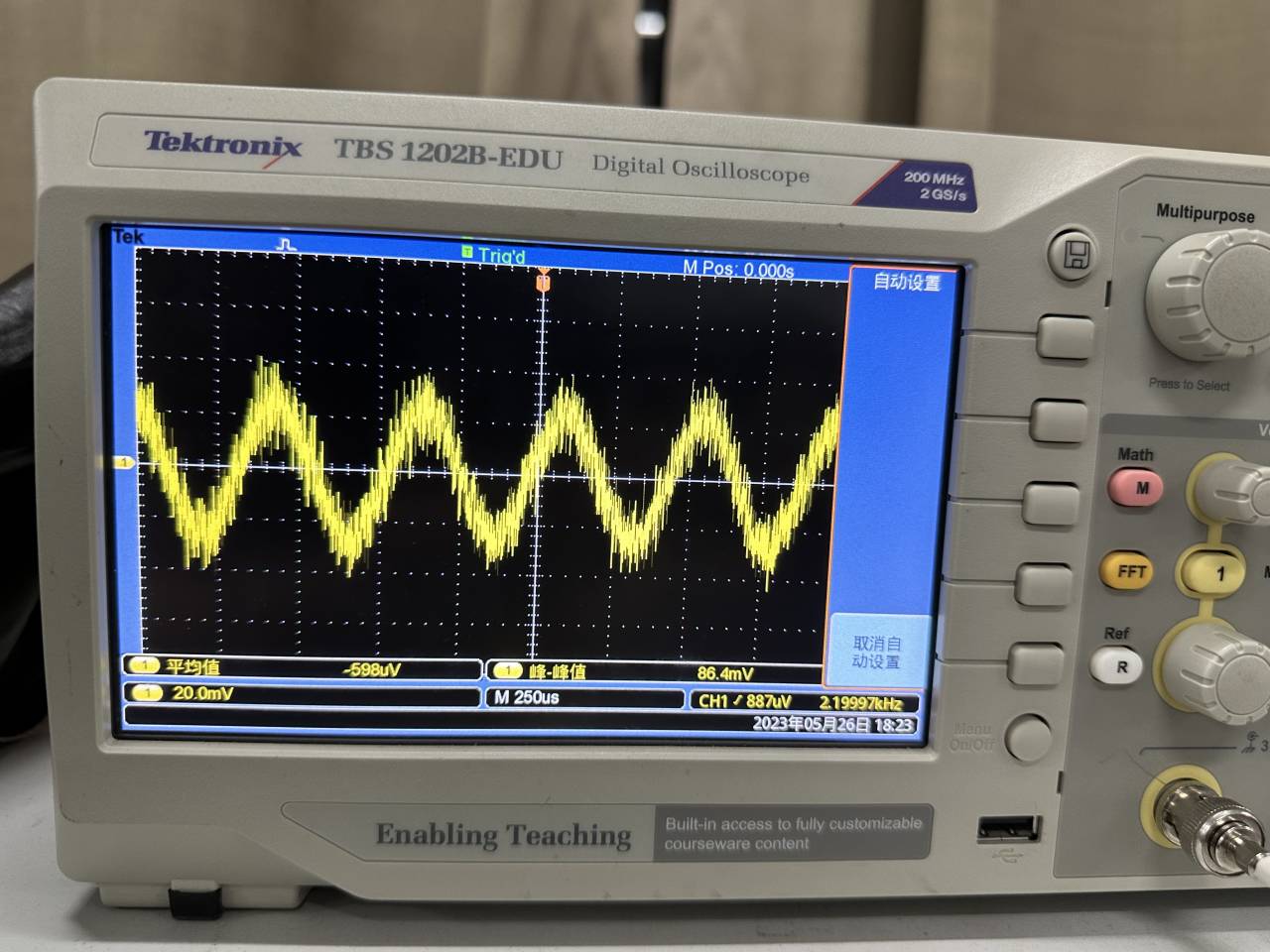
 

图16：2100Hz时的输出波形 图17：2200Hz时的输出波形

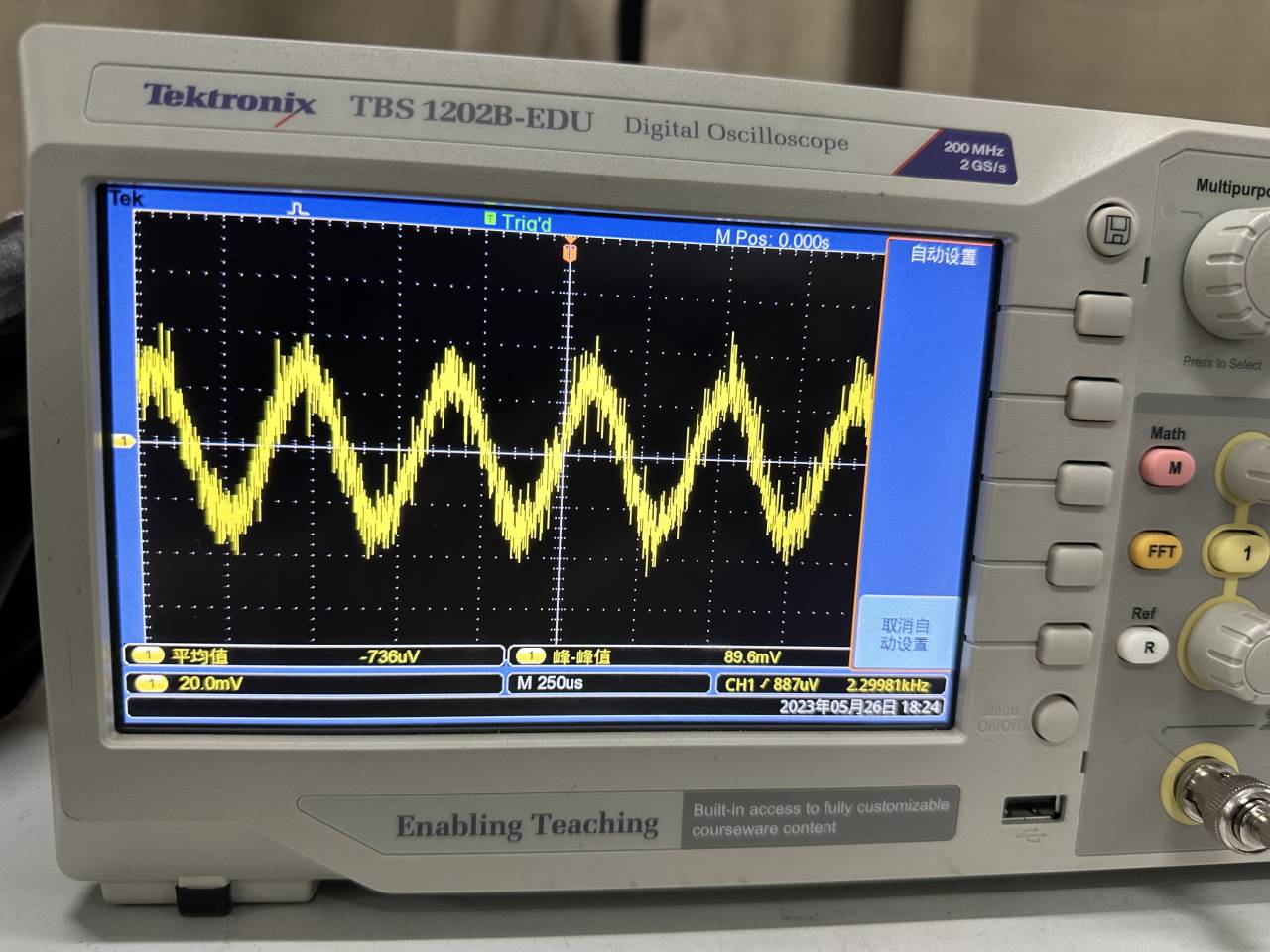
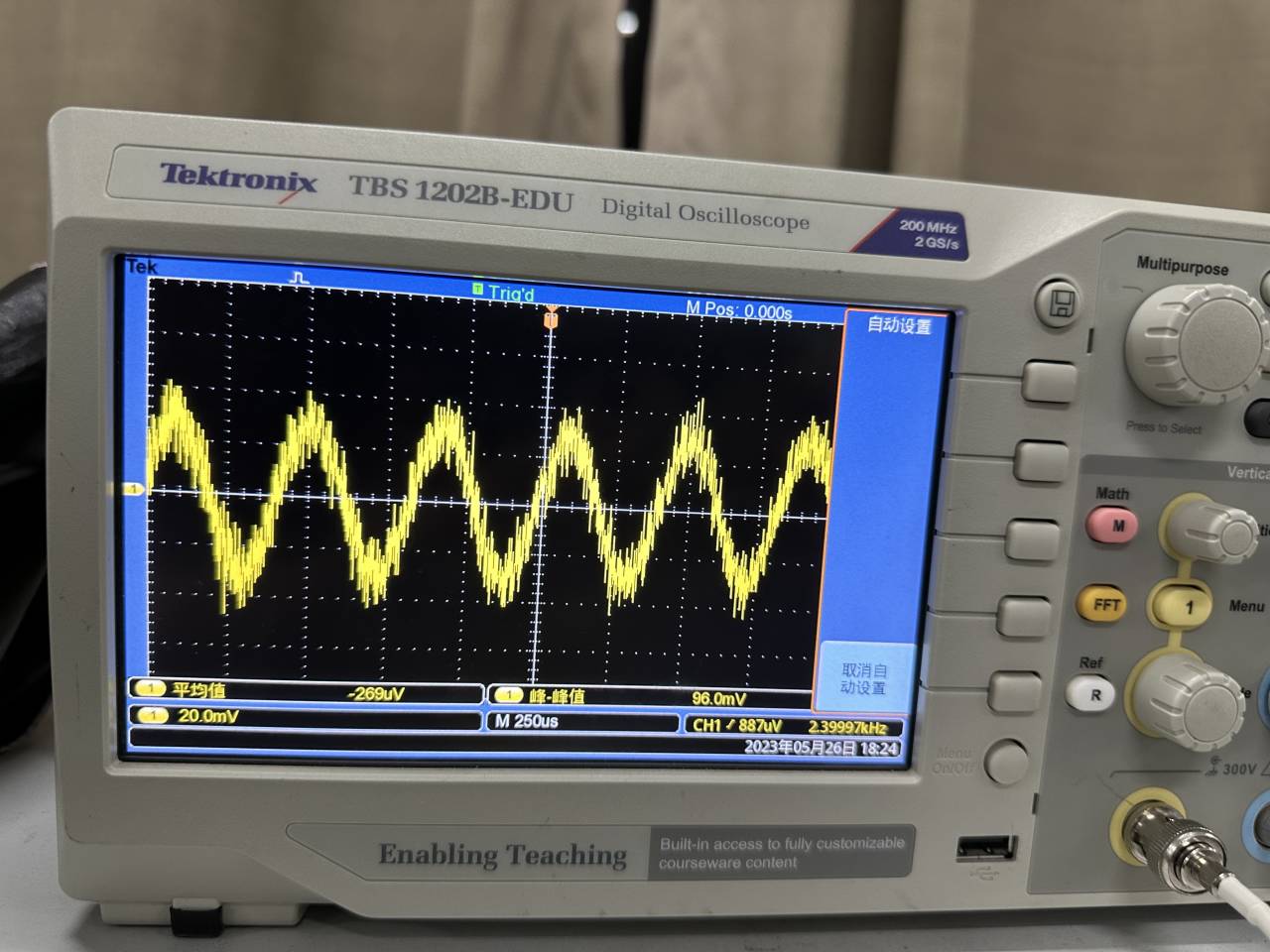
 

图18：2300Hz时的输出波形 图19：2400Hz时的输出波形

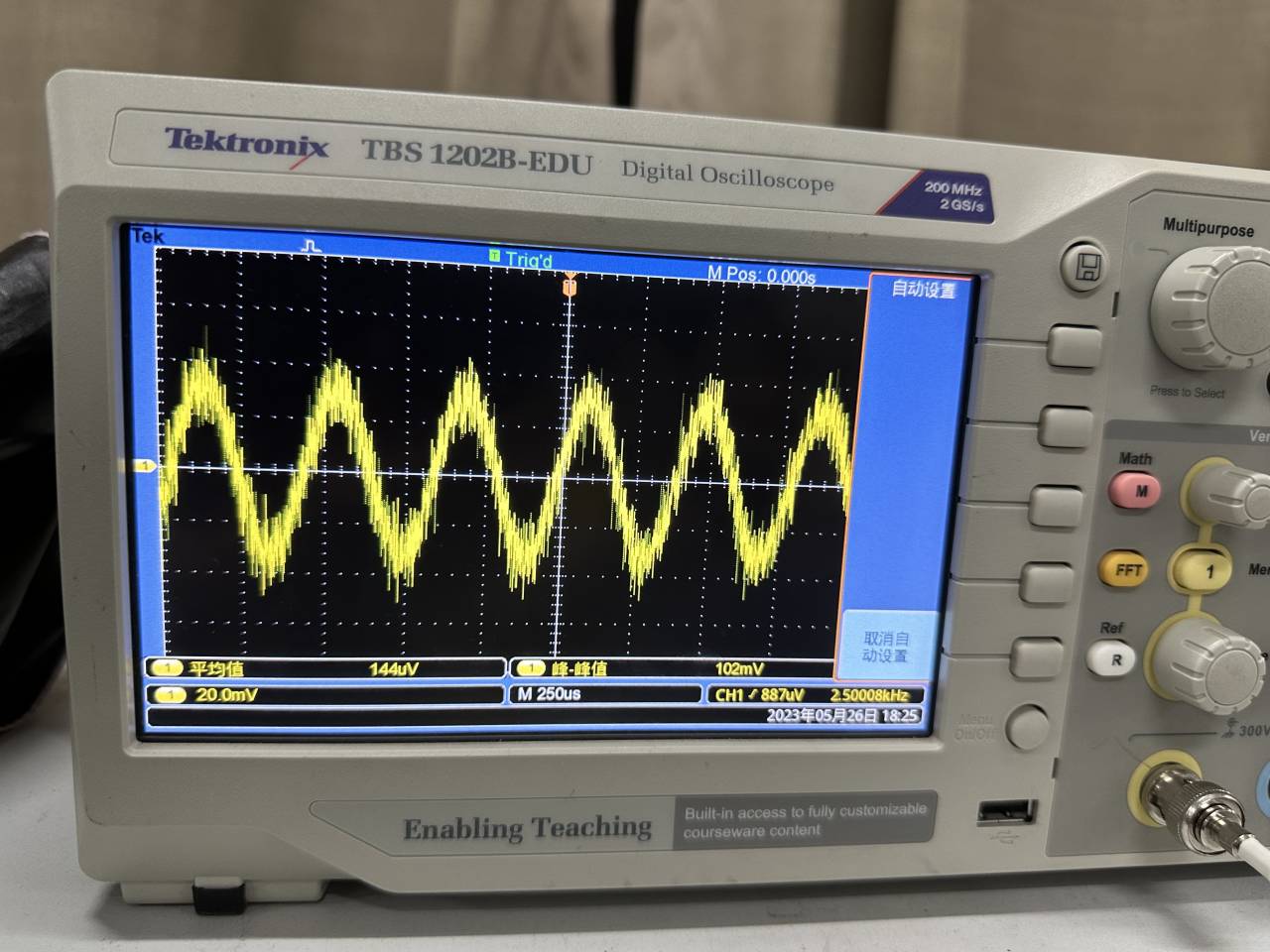
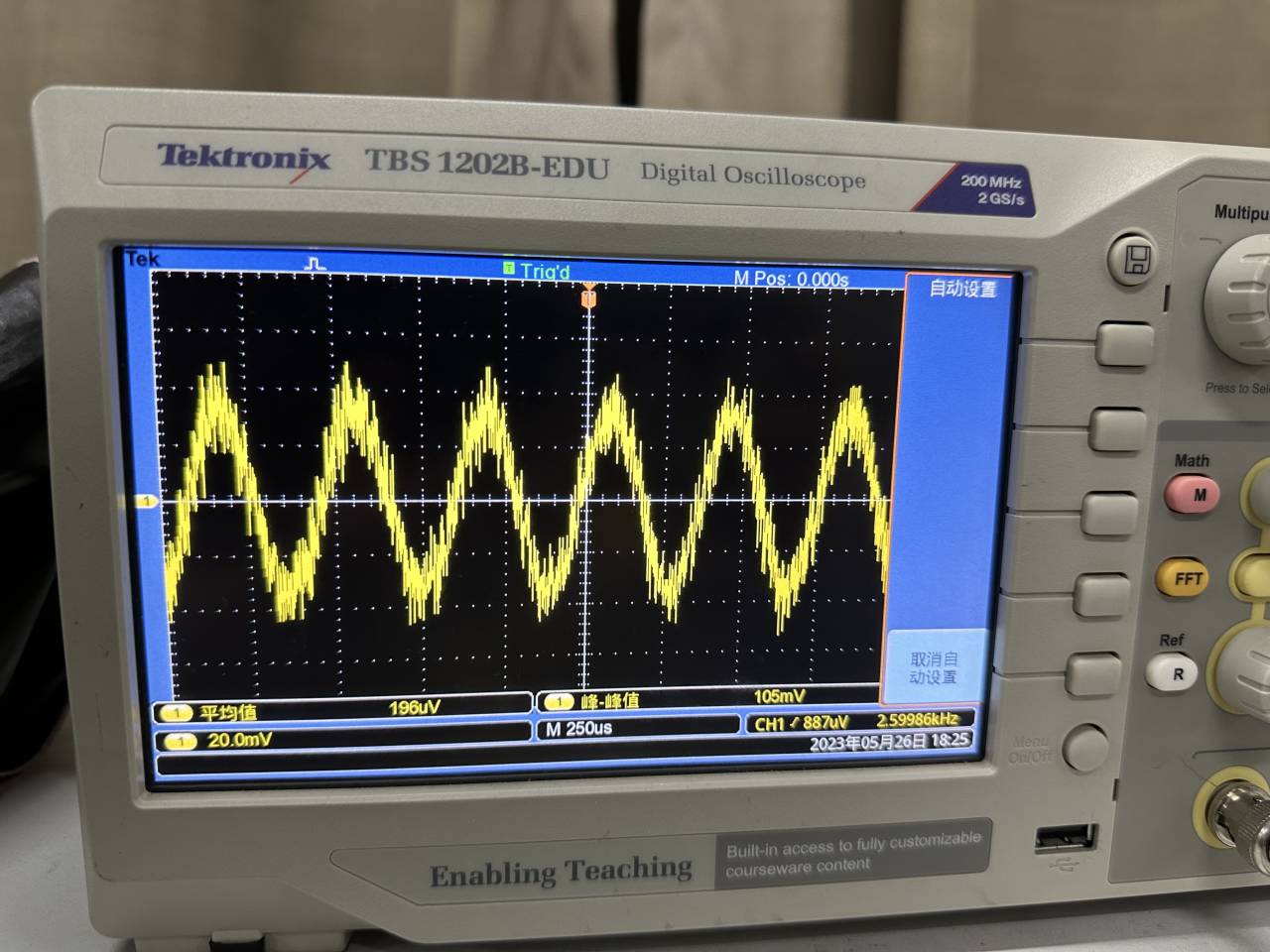
 

图20：2500Hz时的输出波形 图21：2600Hz时的输出波形

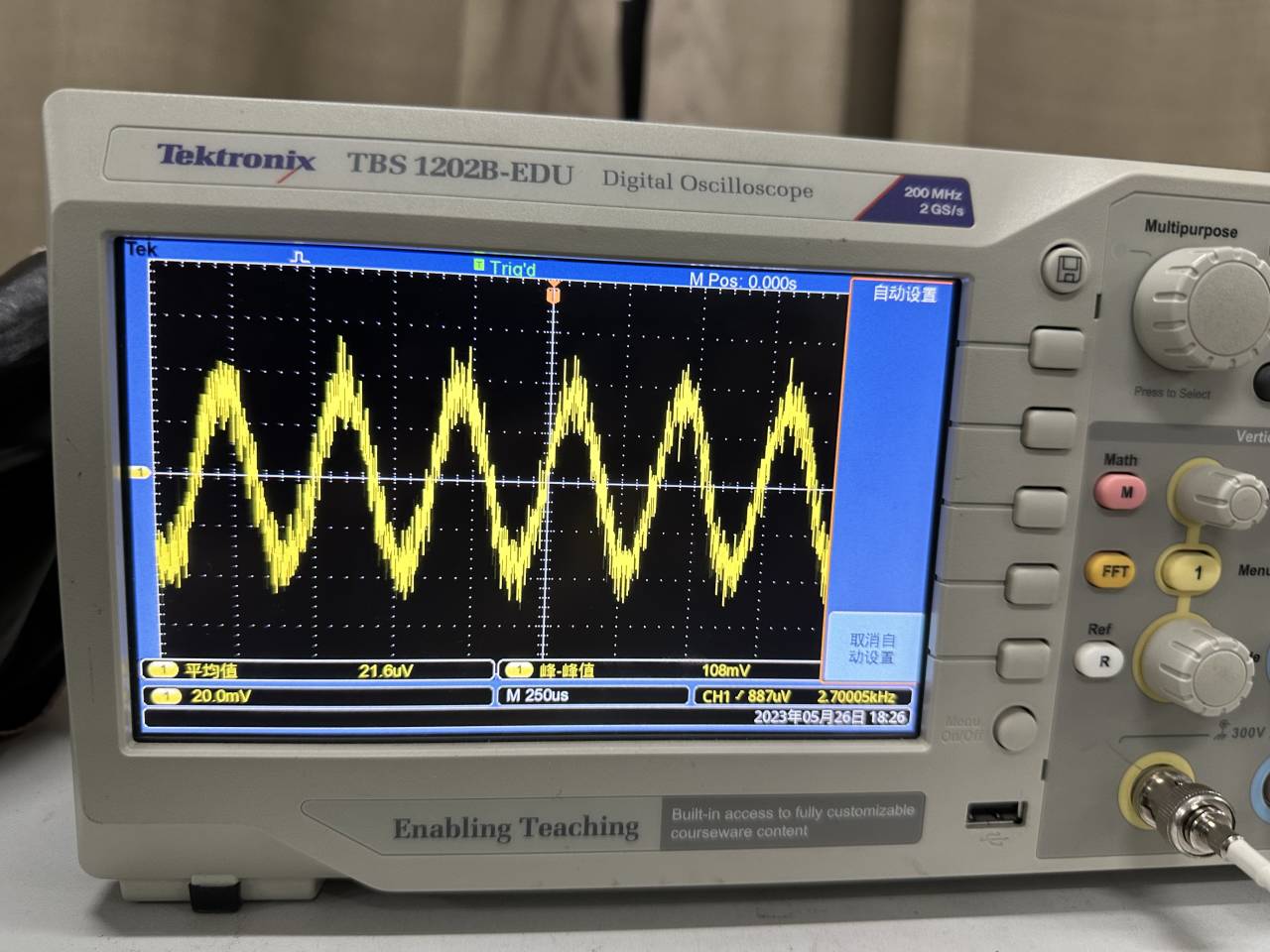
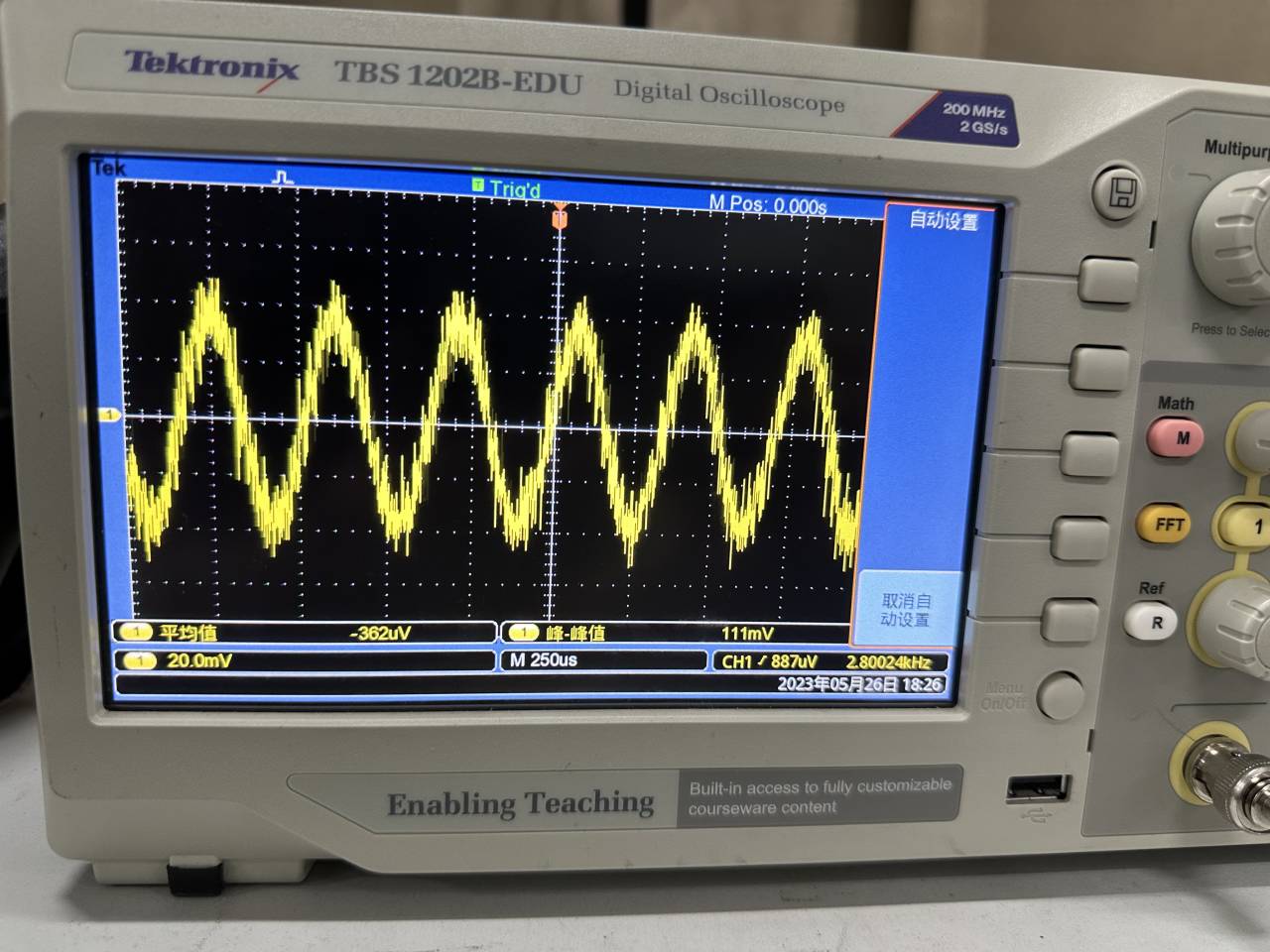
 

图22：2700Hz时的输出波形 图23：2800Hz时的输出波形

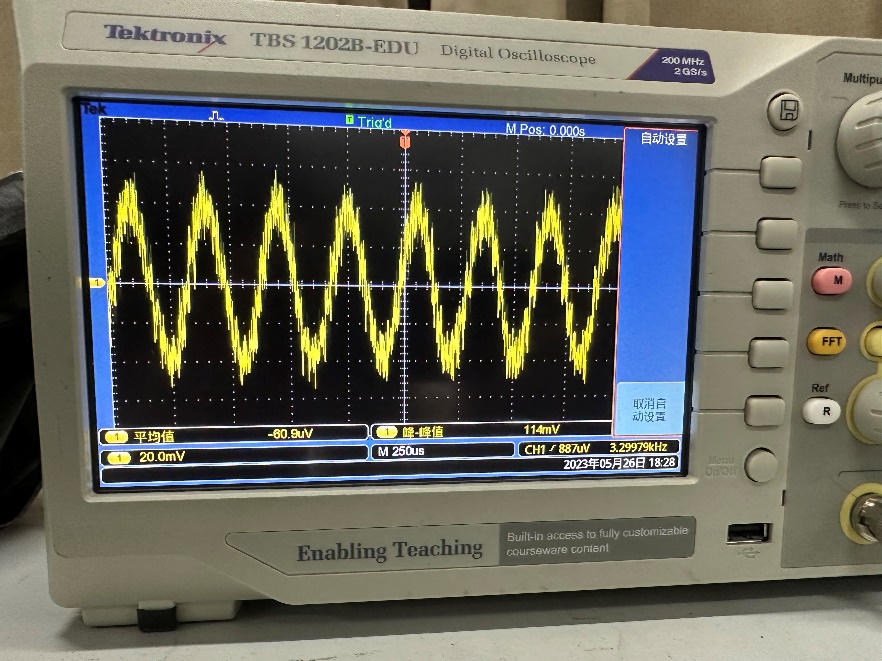
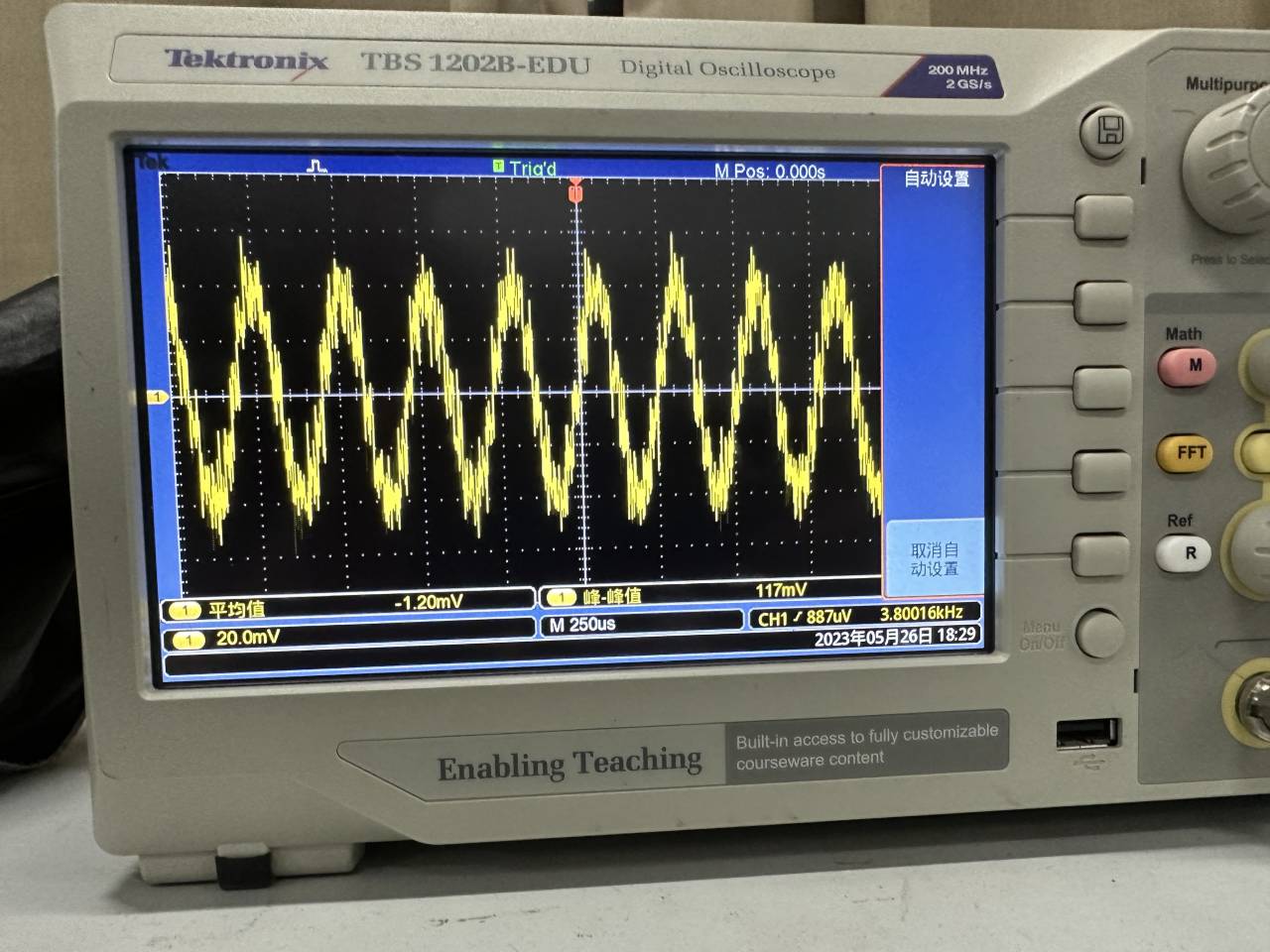
 

图24：3300Hz时的输出波形 图25：3800Hz时的输出波形

可以看到随着输入频率的升高，示波器输出幅度逐渐增大，达到2.5kHz后，输出幅度趋于稳定区间，直到4.5kHz开始衰减，输出波形逐渐趋于直线；随着输入频率的继续增大，输出频率也不再正确反映输入信号，此时可以认为超出5kHz的信号被滤除。综合以上的功能验证，我们可以看出测试结果基本符合设计的带通滤波器幅频特性曲线。

衰减部分的具体输出波形如**图26-图33**所示：

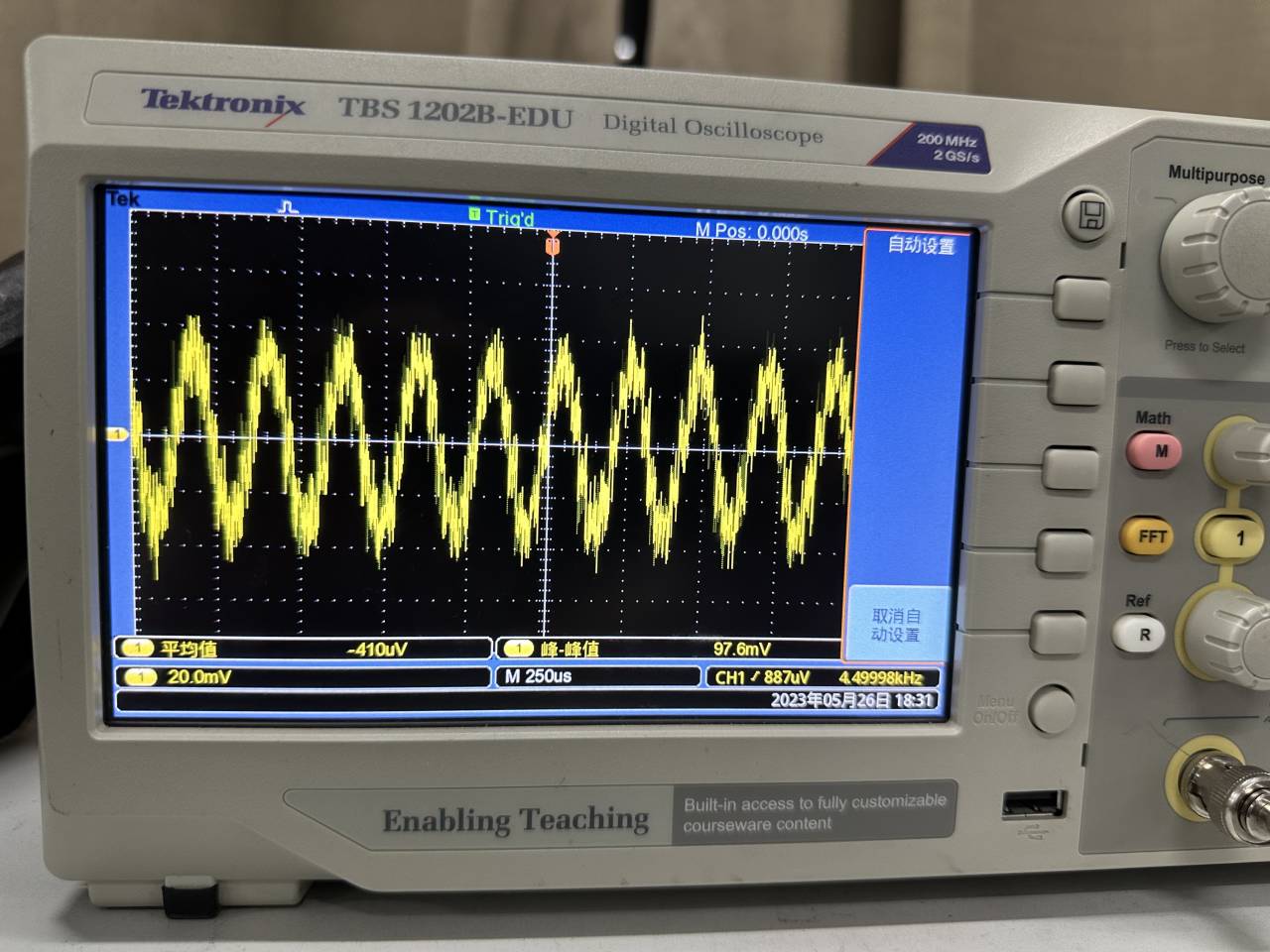
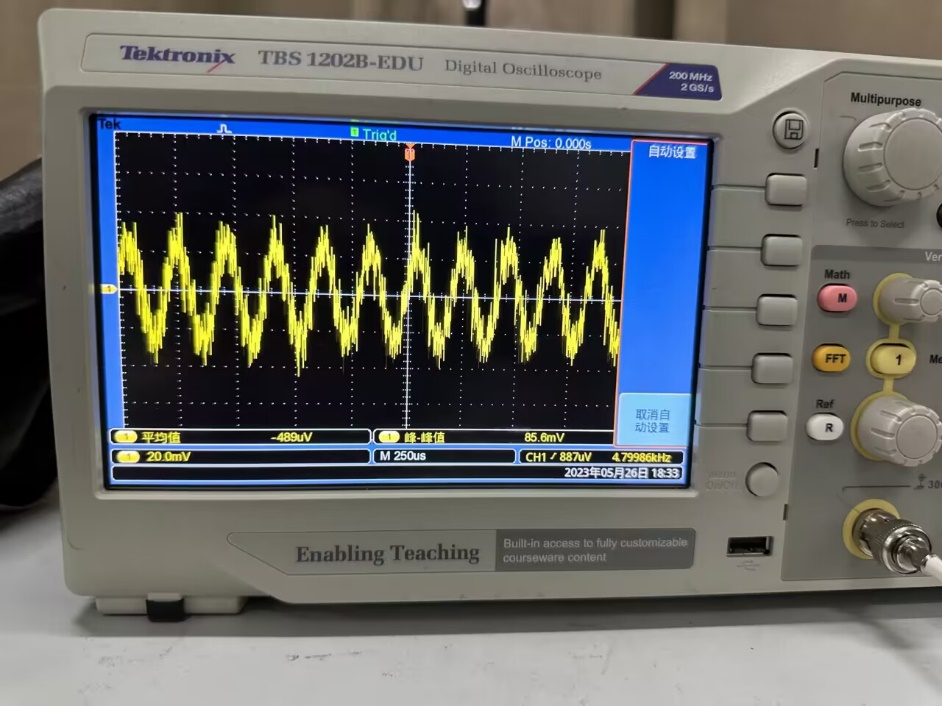
 

图26：4500Hz时的输出波形 图27：4800Hz时的输出波形

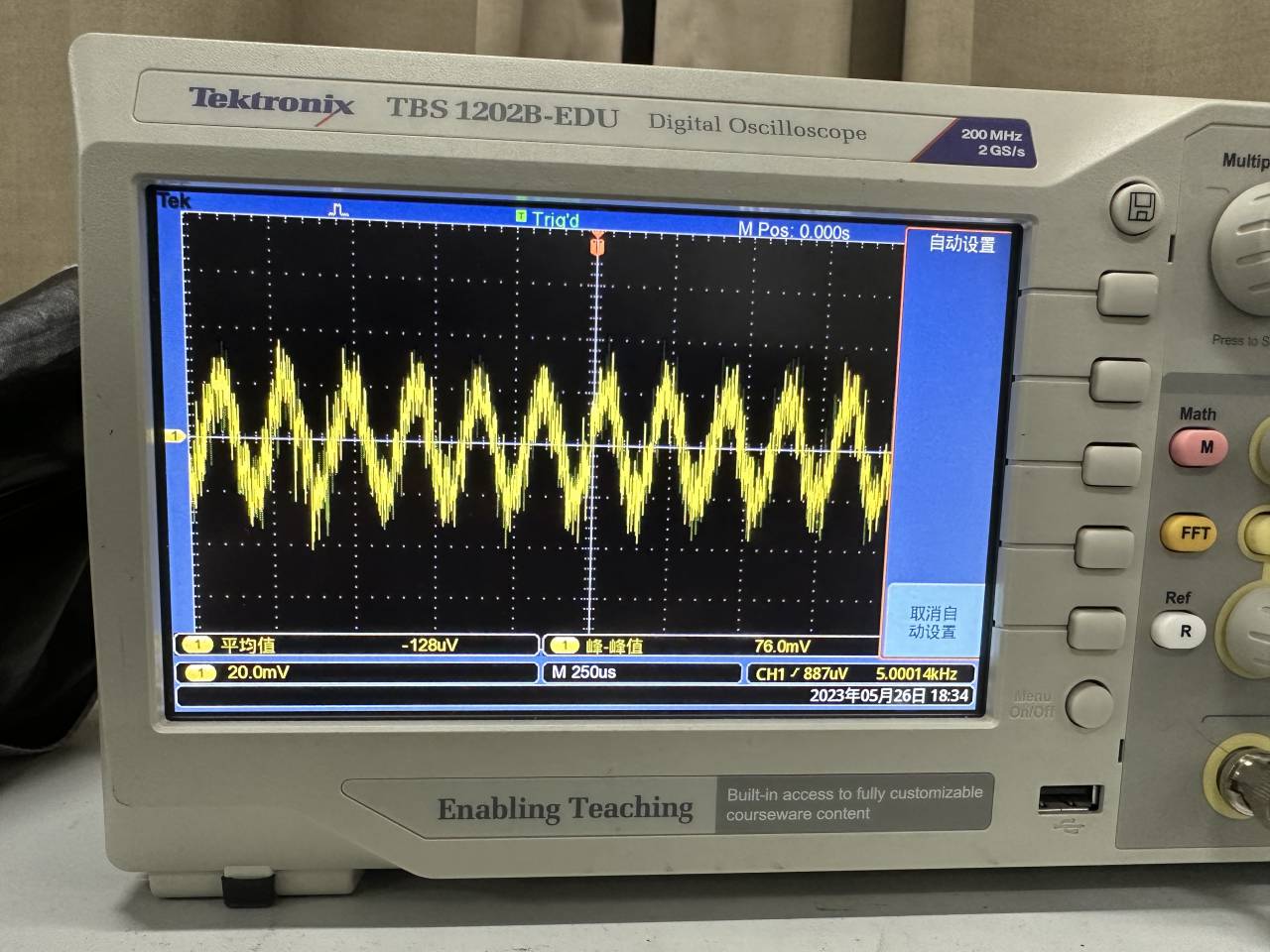
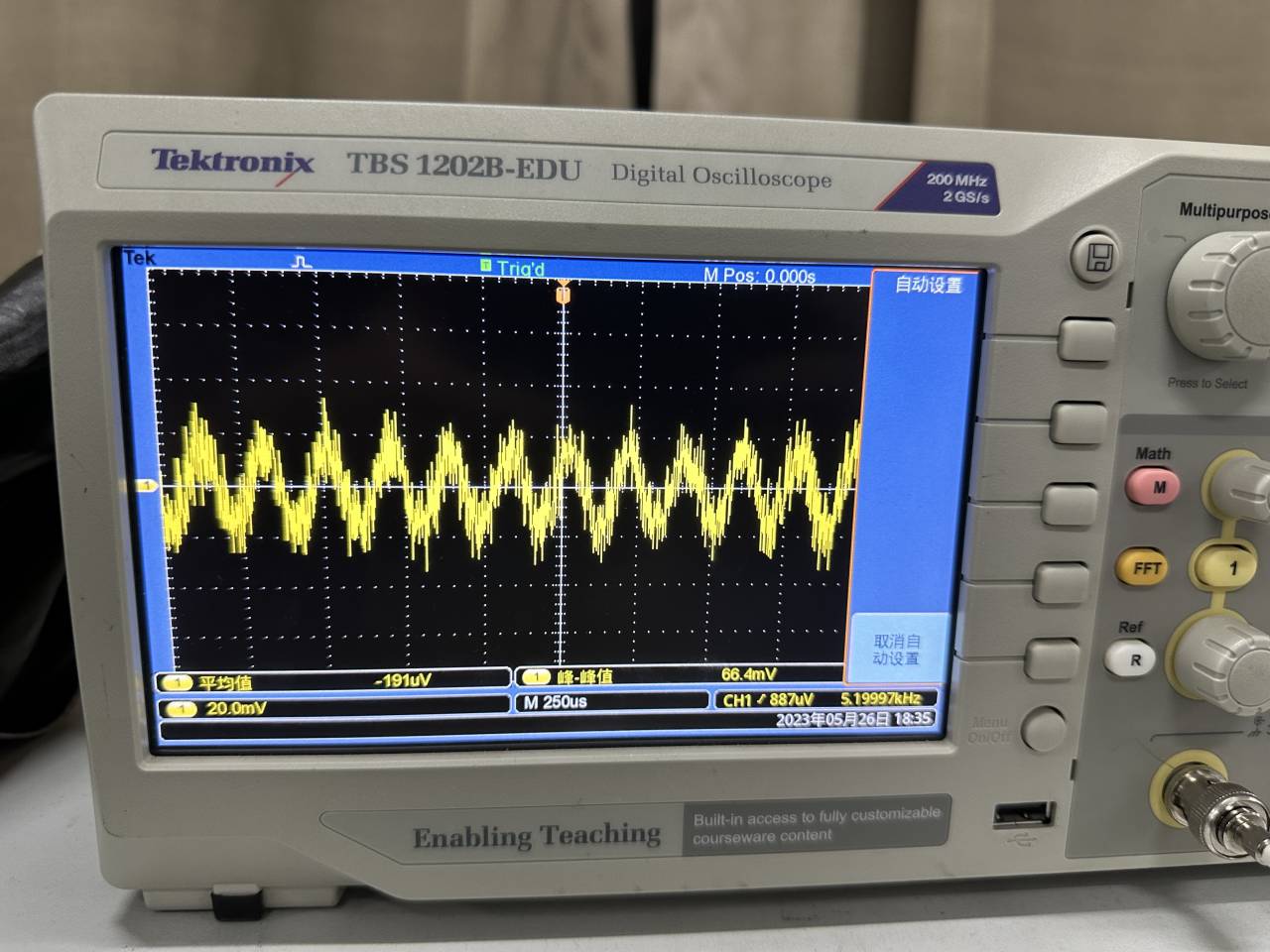
 

图28：5000Hz时的输出波形 图29：5200Hz时的输出波形

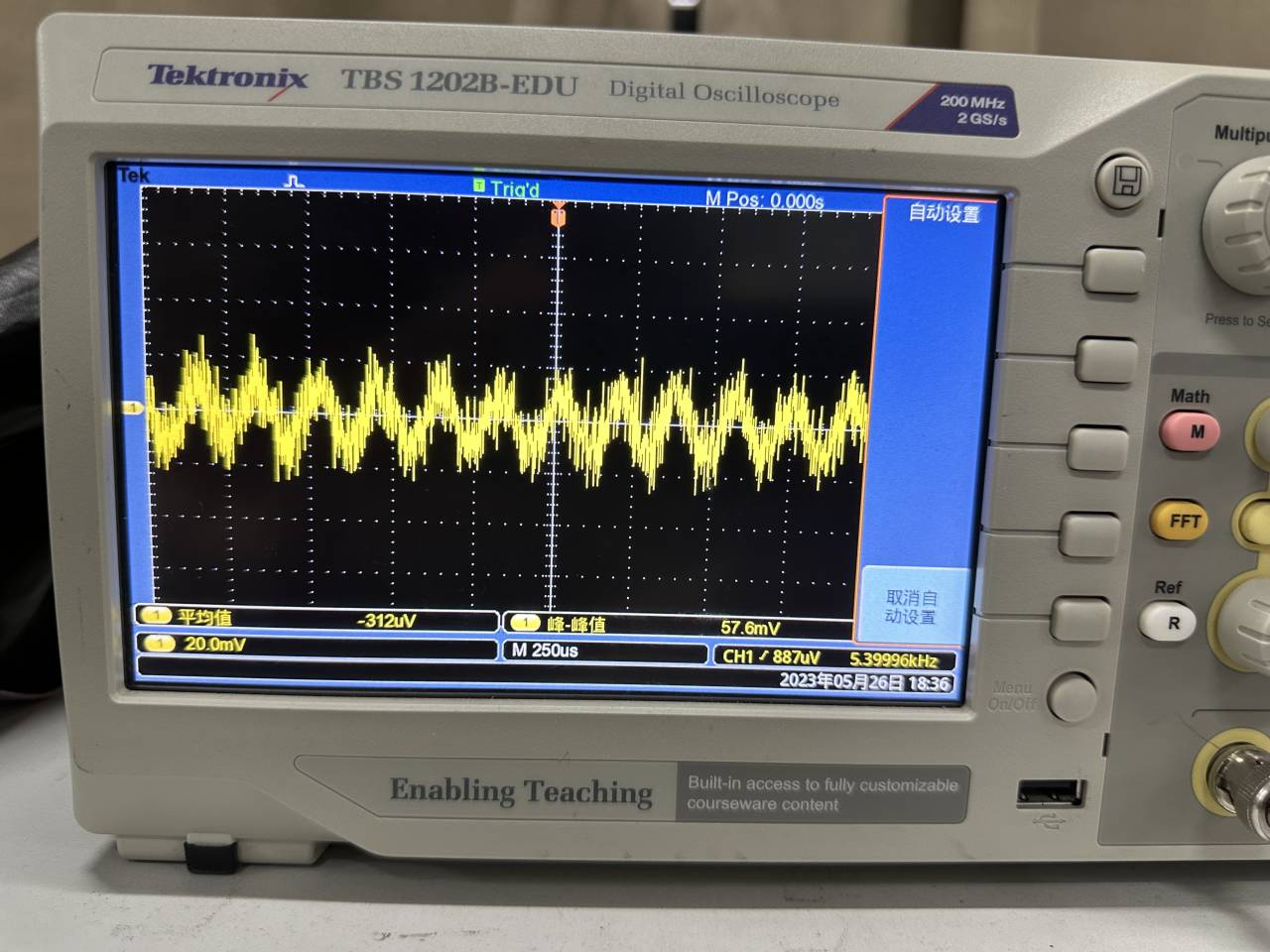
 

图30：5400Hz时的输出波形 图31：5600Hz时的输出波形

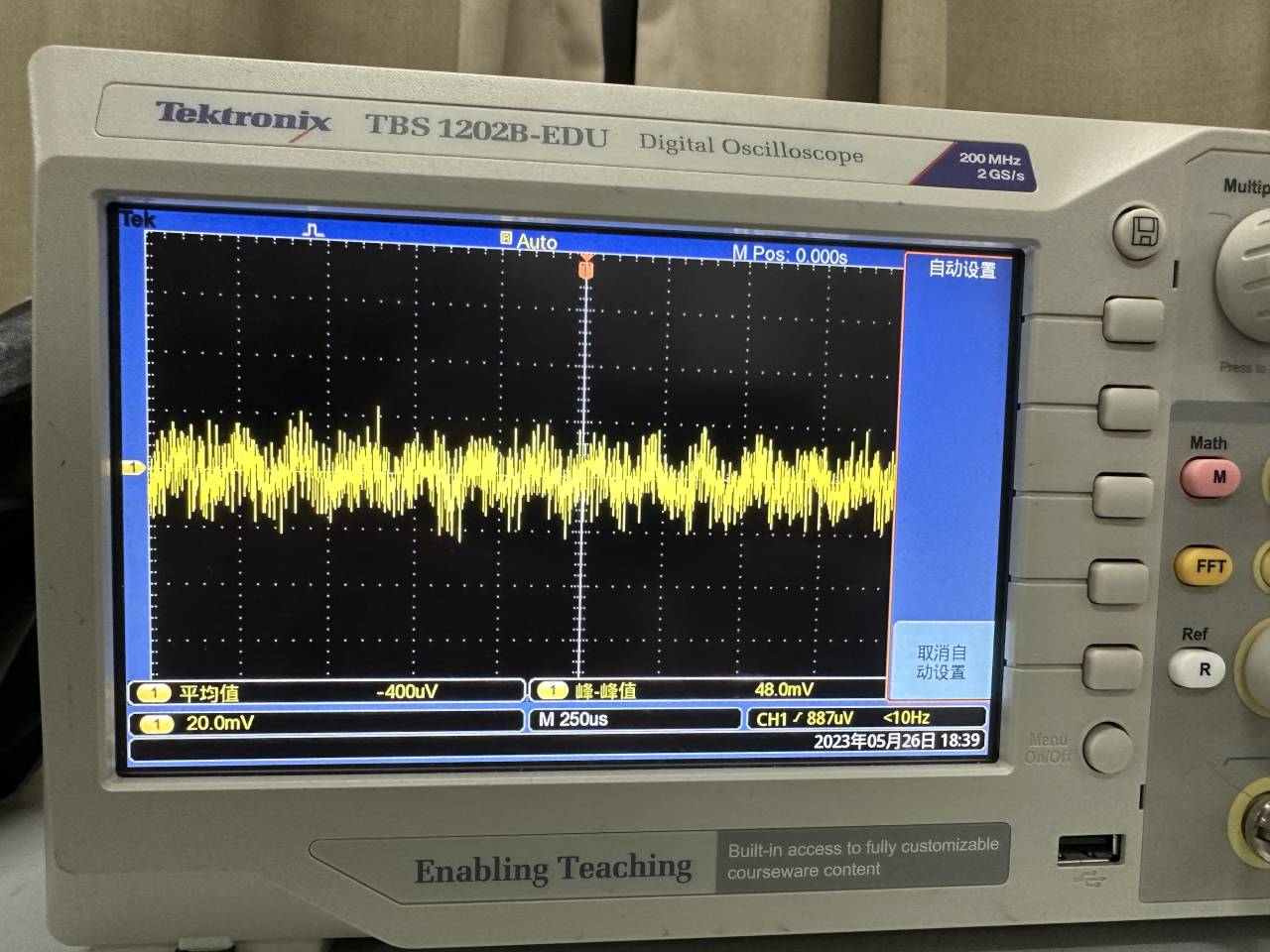
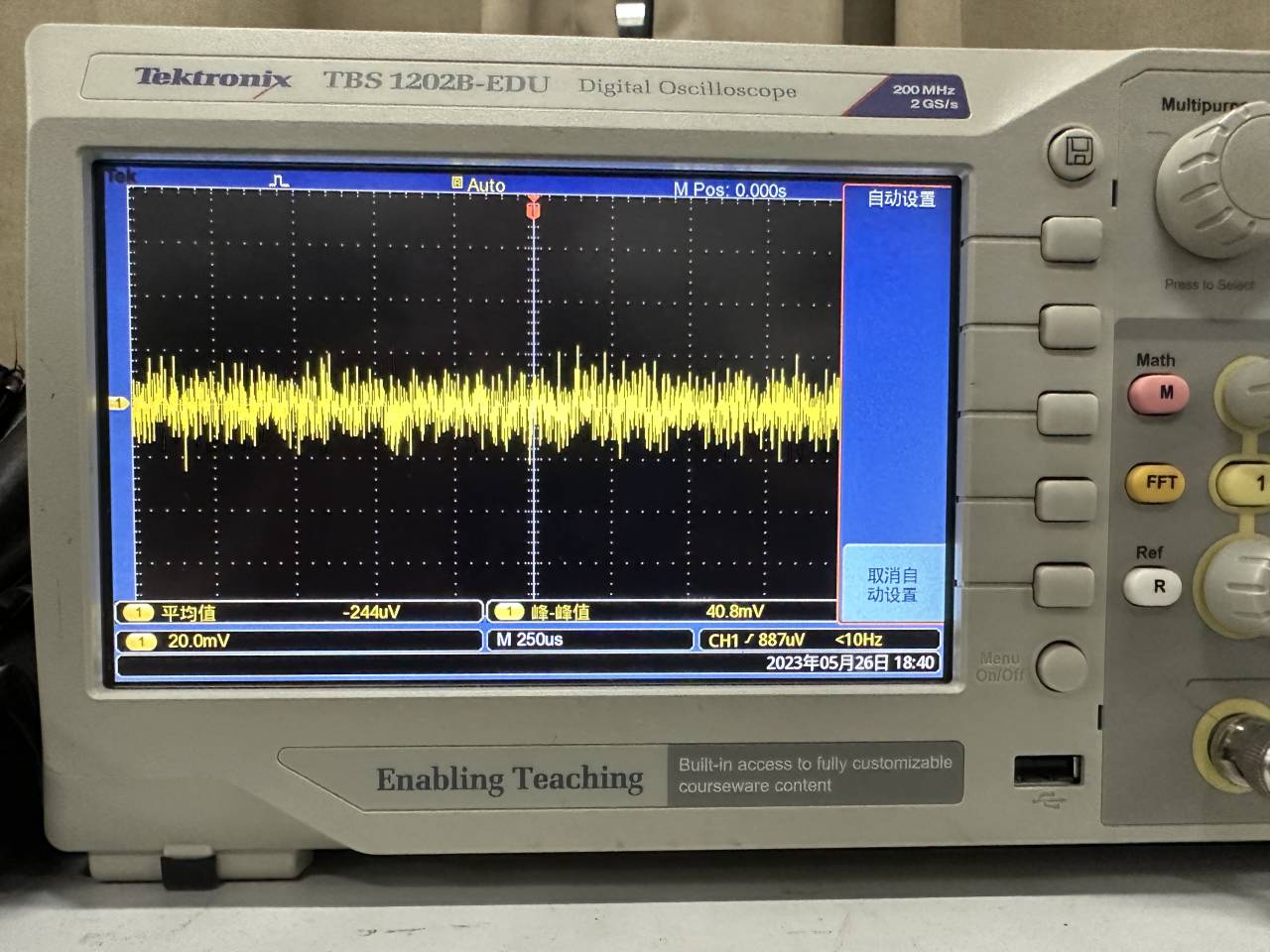
 

图32：5800Hz时的输出波形 图33：6000Hz时的输出波形

# 12.6 实际幅频特性曲线

改变输入正弦信号的频率，记录下对应的峰值幅度，描点作图并与理论幅频特性曲线进行比较。具体波形见12.5；经过整理，得到实际的幅频特性情况记录表，如**表3**所示：

表3：实际幅频特性数值记录表



根据表格作图，得到实际的幅频特性曲线，如**图34**所示：

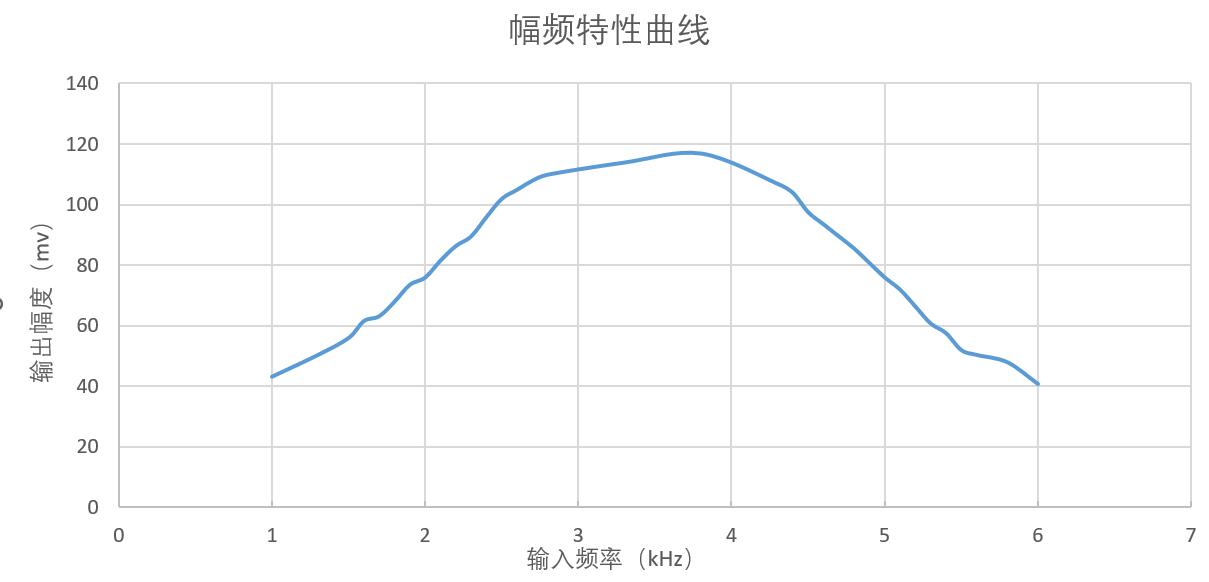


图34：实际幅频特性曲线

可以看出在误差允许范围内，实际幅频特性曲线较为良好地贴合理论值，即表明FIR带通滤波器的设计较为合理。

# 12.7 算法实时性验证

仿照Lab\_11实验思考的设计，修改中断部分代码，在进入中断时给DA高电平，中断结束后给DA低电平。具体代码如**图35**所示：



图35：验证算法实时性代码

在示波器上显示方波状，可知本算法具有实时性；利用示波器自带光标测量采样周期和算法工作计算时间，如**图36、图37**所示：

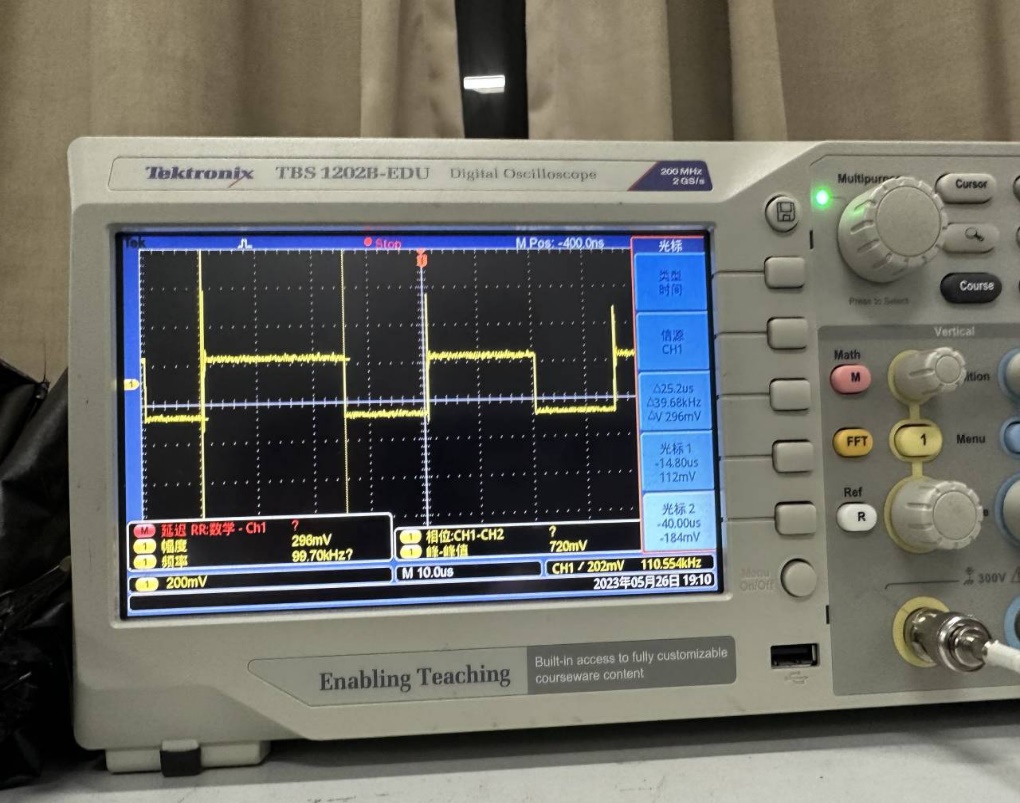
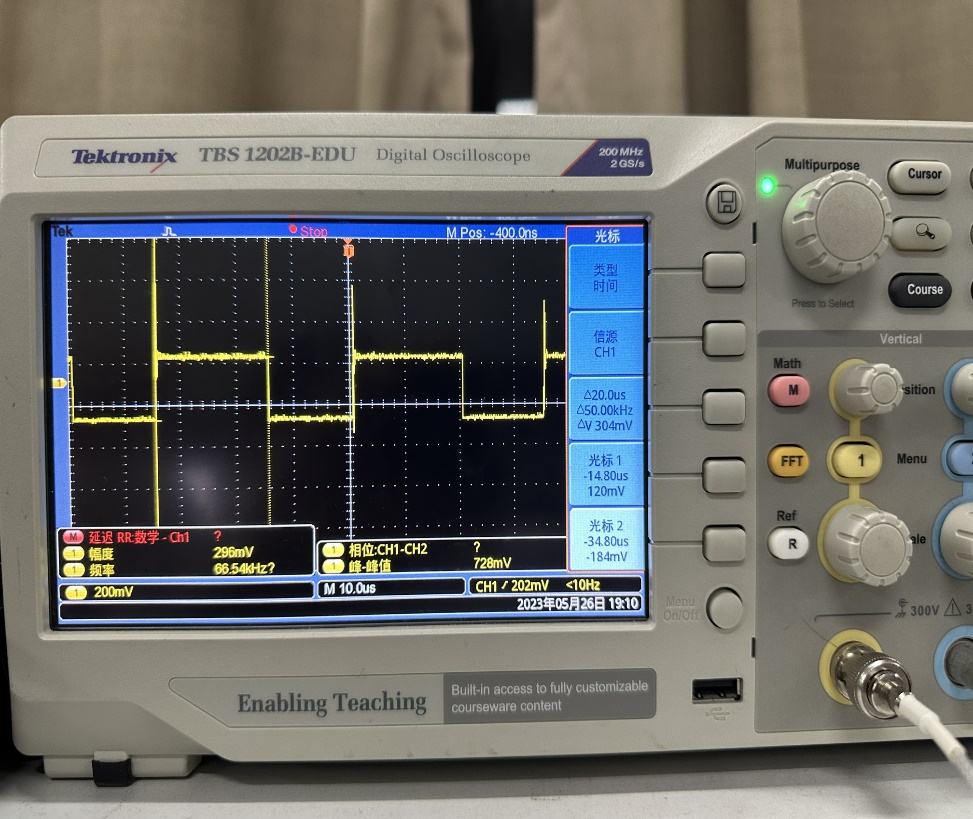
 

图36：测量采样周期 图37：测量中断工作时间

由图可知，FIR滤波器工作时间为20.0us采样周期为25.2us，即DSP芯片设置的采样频率，算法工作时间小于采样时间，符合实时性要求。

# 12.8实验思考和问题分析

**1.指出FIR滤波器系数的设计参数指标**

在MATLAB中，fir1函数用于设计FIR（有限脉冲响应）数字滤波器。它的使用方法如下：

h = fir1(N, Wn, filterType, windowType)

参数的含义如下：

N：滤波器的阶数（长度减1）。较高的阶数可以提供更陡的滤波特性，但会增加计算复杂度。

Wn：归一化的截止频率或频率范围。对于低通或高通滤波器，它是一个标量，表示截止频率。对于带通或带阻滤波器，它是一个长度为2的向量，表示频率范围。截止频率或频率范围的取值范围是0到1，其中1对应于采样率的一半。

filterType：滤波器类型。可以选择的滤波器类型包括：'low'（低通）、'high'（高通）、'bandpass'（带通）和'stop'（带阻）。

windowType（可选）：窗函数类型。常见的窗函数类型包括：'rect'（无窗函数）、'hamming'（Hamming窗）、'hanning'（Hanning窗）和'blackman'（Blackman窗）等。

函数的输出是一个FIR滤波器的系数向量h，可以将其应用于输入信号使用filter函数进行滤波。

在本实验中，我们设计了一个48阶、数字频率范围为0.1π到0.25π 的带通滤波器，没有加窗（或是加矩形窗）。

**2.观察各种输入信号通过数字滤波器系统之后的输出波形，解释信号失真的原因。**

信号失真的原因主要有两个方面。

首先，由于DSP的输出字长限制，当对输入信号进行量化时，存在着量化误差。DSP系统将模拟信号转换为数字信号时，需要将连续的模拟信号离散化表示，但是输出位数有限，无法完全表达输入信号的所有细节。因此，输出信号的位数小于滤波器输出的理论值，这导致了截取误差的产生。截取误差会引入额外的噪声成分，使得输出信号与原始信号存在差异。

另外，信号通过数字滤波器时，会受到滤波器的特性影响。例如，当信号通过低通滤波器时，滤波器会抑制通带范围外的高频信号成分，只允许通过通带范围内的低频信号成分。如果输入信号包含了通带范围外的频率分量，这些频率分量会被滤波器抑制，从而引起输出波形的失真。这种失真通常表现为波形的形状变化、频率成分的丢失或畸变。

**3.由汇编语言编写的FIR滤波器程序，其运算时间高于C语言，分析C语言效率低的原因。**

汇编语言相比C语言具有更高的效率，主要是因为它直接操作底层硬件资源，更加贴近计算机体系结构的操作。影响C语言效率底下的原因可能有以下几个：

编译器优化限制：C语言的编译器在将代码转换为机器语言时进行了一系列优化，但优化会导致一些性能上的损失。与此相比，汇编语言代码是直接由程序员编写的，可以针对特定的硬件架构进行高度优化。

内存访问效率：C语言在处理FIR滤波器时会使用数组存储滤波器系数和输入数据。但C语言的数组访问是通过指针和偏移量进行的，这涉及到多级寻址和间接寻址的操作，会引入额外的开销。汇编语言可以更直接地访问内存和寄存器，减少了寻址和间接寻址的开销。

控制流开销：C语言的控制流结构通常需要使用分支指令来实现，这会引入额外的分支预测和分支跳转开销。

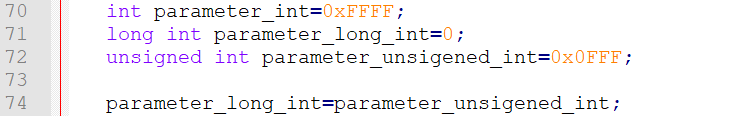
**4.DSP芯片存储器字长转换关系验证**

在程序中，涉及到16位有符号、无符号数相乘等运算，涉及到字长的变化，需要先对DSP芯片中字长变化的关系进行验证，主要分3部分：

|  |
| --- |
| **（1）验证有符号长整型变量能否直接接受有符号数与无符号数相乘的结果**  在主程序中编写验证代码如下：运行后，查看变量结果如图所示。则可知，这种运算方式与期望值（-4095）不一致，不可以使用这种方式来进行字长转换。 |

**（2） 验证无符号整型变量能否直接赋值给有符号长整型变量**

在主程序中编写验证代码如下：



运行后，查看变量结果如图所示。则可知，无符号整型变量直接赋值给有符号长整型变量后，值不变，可以利用这种方式实现字长转换。

|  |
| --- |
|  |
| **（3）验证有符号长整型变量赋值给无符号整型变量的结果**  在主程序中编写验证代码如下：    结果如图所示。parameter\_unsigened\_int=0xEDF1。则可知，有符号长整型变量赋值给无符号整型变量后，低16位有效。 |
|  |
|  |

**5.以该FIR滤波器系统为例，总结分析系统实时性的取决因素。**

按照软硬件来区分，可以将实时性取决因素划分为运算量和芯片速度两个方面，而运算量又可以分为数据率和算法复杂程度。实时性的要求是指在下一个数据到达之前，前一个数据的处理必须完成。根据验证算法实时性的实验，采样频率越高，矩形波占空比越大，系统实时性越难以满足。另外，通过改变滤波器阶数N发现，N越大，即算法复杂度越高，实时性越难满足。

**6.观察各种输入信号通过数字滤波器的波形**

由于本滤波器是带通滤波器，因此2.5kHz的方波、三角波等波形在通过本滤波器后，会出现不同程度的失真，形状逐渐向基频的正弦波靠拢。这是因为这些波是由各种不同频率的正弦波叠加而成，因此在滤波后，只剩下了基频或基频和二次谐波部分，高次谐波部分被滤除。

**7.如何用该系统实现实时FFT**

可以调用TI提供相关dsp算法库中的fft库，如下图所示：



这段程序调用了“fft.h”，实现了一个基于512点Q31定点数FFT的频谱分析功能。首先调用函数 RFFT32\_brev() 对输入缓冲区 ipcbsrc 进行实数FFT的位反转操作，将数据重新排列为正确的顺序。然后将结构体对象 rfft 的成员变量指向相应的缓冲区和数组：rfft.ipcbptr 指向输入缓冲区 ipcb、rfft.magptr 指向幅度输出缓冲区 ipcbsrc、rfft.winptr 指向窗口系数数组 win。之后调用 rfft.init(&rfft) 来初始化结构体对象，主要是对旋转因子指针进行初始化。最后调用 rfft.calc(&rfft) 来计算FFT，rfft.split(&rfft) 进行后处理操作，以获取正确的频谱， rfft.mag(&rfft) 计算幅度，结果以Q31格式表示，即取绝对值并除以2^16后再平方。

我们可以将这段程序添加进已有的数据采集和输出程序，对不同频率成分的信号进行AD采集和转换，经过FFT处理，最后用DA输出信号频谱图。

**实验过程中我们遇到了以下问题：**

**1.示波器输出波形显示波形较粗并且抖动剧烈。**

经过进一步分析，我们发现问题出在系数定标上。我们的定标方法存在一定的问题，导致滤波器的系数范围没有正确地映射到合适的数值范围上。经过修正后的定标方法应该是在系数表中找出具有最大值的h系数，其值为0.15，并将其映射到16位有符号数的最大值。换言之，滤波器的系数之间的差异倍数并不大，这就导致了滤波效果的不佳。

**2.采样率过高，算法时间余裕不大**

我们使用40kHz作为采样频率，采样周期就是25us，通过算法实时性的验证实验可以看出，算法时间为20us，如果还想对数据进行一些其他处理的话，现有采样周期的余裕不大，就要被迫降低采样率。

# 12.9 实验总结

本次实验由于没有示例程序，因此需要自己对实验三的程序进行修改，以实现功能要求。因此难度相比前面的实验高了很多。回顾已做的实验，实际上都在为最后的实验奠定基础，确保电脑可以正确连接dsp仿真器并在线调试、DA模块输出正常、AD模块采样正常，这些都是滤波器实验高楼大厦的一砖一瓦，让我深刻体会到模块化工程调试的思想。

本次实验有了需要借助外部软件自行设计的部分，不同的滤波器参数得到的结果也会有所不同。这就要求我们准确掌握dsp相关原理知识，这样既可保证后续程序的可靠性，同时也可以在一定程度上降低验证结果的复杂度。

在验证仿真滤波器功能的过程中，如何编写实现代码成为了困扰我们多时的问题。因牵扯到数据移位和各种数制之间的转换，还要检查数组越界与否，我们前后多次调整程序，不断对猜想进行调试验证，尽量高效地修改程序，最终得到了理想的输出结果。但最后滤波输出波形有较大噪声，我们在现场没有深究滤波器系数定标的问题，而是选择先拍照记录，错失了发现问题之后再次调试的机会。

最后，感谢同组成员的相互配合。感谢助教和老师的指导！