IE6, OSL – Lab Test 4

Task 1: Synchronization

Complete the project Control:

- Add events and event handling.
- Events should be named uniquely.
- This process should do some output, e. g. to inform the user about a start of the calculation.
- This process recieves events, if a trigger signal has been recognized or if the calculation has been finished.

control.cpp

```
#include <iostream>
#include <windows.h>
#include <stdio.h>
#include <conio.h>
#include <cstdint>
#include <thread>

#define MYNAME "Control"

HANDLE ghInputEvent;
HANDLE ghStartProcessingEvent;
HANDLE ghProcessingFinishedEvent;
HANDLE ghOutputEvent;
HANDLE ghExitEvent;

DWORD WINAPI Control(LPVOID);

int main()
{
        // handles for process and threads
        HANDLE hProcess = GetCurrentProcess();
        HANDLE hThread = GetCurrentThread();

        // set priority and processors
        int threadPriorityMask = THREAD_PRIORITY_NORMAL;
        DWORD processAffinityMask = 0x00000001;

        if (!SetThreadPriority(hThread, threadPriorityMask))
        {
                printf("SetThreadPriority failed process name: %s, error: %d\n", MYNAME, GetLast
                return EXIT_FAILURE;
        }
        if (!SetThreadAffinityMask(hThread, processAffinityMask))
        {
                printf("SetThreadAffinityMask failed process name: %s, error: %d\n", MYNAME, Get
                return EXIT_FAILURE;
        }
        // get info
        int processid = GetProcessId(hProcess);
        int threadid = GetThreadId(hThread);
        int processornum = GetCurrentProcessorNumber();
        DWORD ppriority = GetPriorityClass(hProcess);
        int tpriority = GetThreadPriority(hThread);

        std::cout << "Process ID: " << processid << "\nThread ID: " << threadid << "\nNow runnir
                          << "\nWith process priority class 0x" << std::hex << ppriority << "\nW

        BOOL bContinue = TRUE;
```

```
DWORD dwEvent;

// Create events

// create input event:
ghInputEvent = CreateEvent(
        NULL,                           // default security attributes
        FALSE,                          // Auto Reset! Only one process / thread gets t
        FALSE,                          // initial state: nonsignaled
        TEXT("InputEvent")); // name of the event to adress from other processes
// check if event was created successfully:
if (ghInputEvent == NULL)
{
        printf("%s: Create InputEvent error: %d\n", MYNAME, GetLastError());
        ExitProcess(0);
}

// create start processing event:
ghStartProcessingEvent = CreateEvent(
        NULL,                                           // default security attribute
        FALSE,                                          // Auto Reset! Only one proce
        FALSE,                                          // initial state: nonsignalec
        TEXT("StartProcessingEvent")); // name of the event to adress from other process
if (ghStartProcessingEvent == NULL)
{
        printf("%s: Create StartProcessingEvent error: %d\n", MYNAME, GetLastError());
        ExitProcess(0);
}

// create processing finished event:
ghProcessingFinishedEvent = CreateEvent(
        NULL,                                               // default security at
        FALSE,                                              // Auto Reset! Only or
        FALSE,                                              // initial state: nons
        TEXT("ProcessingFinishedEvent")); // name of the event to adress from other proc
if (ghProcessingFinishedEvent == NULL)
{
        printf("%s: Create ProcessingFinishedEvent error: %d\n", MYNAME, GetLastError())
        ExitProcess(0);
}

// create output event:
ghOutputEvent = CreateEvent(
        NULL,                           // default security attributes
        FALSE,                          // Auto Reset! Only one process / thread gets
        FALSE,                          // initial state: nonsignaled
        TEXT("OutputEvent")); // name of the event to adress from other processes
if (ghOutputEvent == NULL)
{
        printf("%s: Create OutputEvent error: %d\n", MYNAME, GetLastError());
        ExitProcess(0);
}

// create exit event:
ghExitEvent = CreateEvent(
        NULL,                           // default security attributes
        TRUE,                           // Manual Reset! All processes / threads wait fc
        FALSE,                          // initial state: nonsignaled
        TEXT("ExitEvent")); // name of the event to adress from other processes

if (ghExitEvent == NULL)
{
        printf("%s: Create ExitEvent error: %d\n", MYNAME, GetLastError());
        ExitProcess(0);
}

HANDLE ghEvents[] = {ghInputEvent, ghProcessingFinishedEvent, ghExitEvent};
int nNumEvents = sizeof(ghEvents) / sizeof(ghEvents[0]);

printf("%s process is running.\n", MYNAME);

while (bContinue)
{
        dwEvent = WaitForMultipleObjects(
                nNumEvents, // number of objects to wait for
                ghEvents,       // array of objects to wait for
                FALSE,          // wait for all objects to be signaled
                INFINITE);      // wait infinite time
        printf("%s: I got the event %d!\n\n", MYNAME, dwEvent);
        switch (dwEvent)
        {
        case WAIT_OBJECT_0 + 0: // InputEvent
                if (!SetEvent(ghStartProcessingEvent))
                {
                        printf("%s: SetEvent error: %d\n", MYNAME, GetLastError());
                        ExitProcess(0);
                }
                else
                {
                        printf("%s: SetEvent %s\n", MYNAME, "StartProcessingEvent");
                }
                break;
        case WAIT_OBJECT_0 + 1: // ProcessingFinishedEvent
                if (!SetEvent(ghOutputEvent))
                {
                        printf("%s: SetEvent error: %d\n", MYNAME, GetLastError());
                        ExitProcess(0);
                }
                else
                {
                        printf("%s: SetEvent %s\n", MYNAME, "OutputEvent");
                }
                break;
        case WAIT_OBJECT_0 + 2: // ExitEvent
                bContinue = FALSE;
                break;
        case WAIT_TIMEOUT:
                break;

        default:
                printf("%s: Wait error: %d\n", MYNAME, GetLastError());
                ExitProcess(0);
        }

        printf("\n%s: Process stopped. Hit any key to close Window.\n", MYNAME);
        (void)_getch();

        for (int i = 0; i < nNumEvents; i++)
        {
                CloseHandle(ghEvents[i]);
        }
        ExitProcess(0);
}
}
```

Complete the project Processing:

- Add events and event handling.
- Events should be named uniquely.
- This process has to inform the control process about a completed calculation.
- This process recieves a command to start the calculation.

Processing.cpp

```
// Processing.cpp : This file contains the 'main' function. Program execution begins and ends th
//

#include <iostream>
#include <windows.h>
#include <stdio.h>
#include <conio.h>
#include <cstdint>
#include <thread>

#define MYNAME "Processing"

// Create handles for each event
HANDLE ghStartProcessingEvent = NULL, ghProcessingFinishedEvent = NULL, ghExitEvent = NULL;
DWORD WINAPI Processing(LPVOID);


int main()
{
        //handles for process and threads
        HANDLE hProcess = GetCurrentProcess();
        HANDLE hThread = GetCurrentThread();
        //set priority and processors
        int threadPriorityMask = THREAD_PRIORITY_NORMAL;
        DWORD processAffinityMask = 0x00000001;
        if (!SetThreadPriority(hThread, threadPriorityMask))
        {
                printf("SetThreadPriority failed process name: %s, error: %d\n", MYNAME, GetLast
                return EXIT_FAILURE;
        }
        if (!SetThreadAffinityMask(hThread, processAffinityMask))
        {
                printf("SetThreadAffinityMask failed process name: %s, error: %d\n", MYNAME, Get
                return EXIT_FAILURE;
        }



        //get info
        int processid = GetProcessId(hProcess);
        int threadid = GetThreadId(hThread);
        int processornum = GetCurrentProcessorNumber();
        DWORD ppriority = GetPriorityClass(hProcess);
        int tpriority = GetThreadPriority(hThread);

        std::cout << "Process ID: " << processid << "\nThread ID: " << threadid << "\nNow runnir
                << "\nWith process priority class 0x" << std::hex << ppriority << "\nWith threac
```

```
DWORD dwEvent;
BOOL bContinue = TRUE;
volatile int vi, i;

// open StartProcessingEvent:
ghStartProcessingEvent = OpenEvent(EVENT_ALL_ACCESS | EVENT_MODIFY_STATE, FALSE, TEXT("S


if (ghStartProcessingEvent == NULL) {
        printf("%s: Error opening event %s! Are you sure Control is running? Hit any key
        (void)_getch();
        return EXIT_FAILURE;
}
// open ProcessingFinishedEvent:
ghProcessingFinishedEvent = OpenEvent(EVENT_ALL_ACCESS | EVENT_MODIFY_STATE, FALSE, TEXT
if (ghProcessingFinishedEvent == NULL) {
        printf("%s: Error opening event %s! Are you sure Control is running? Hit any key
        (void)_getch();
        return EXIT_FAILURE;
}
// open quit event:
ghExitEvent = OpenEvent(EVENT_ALL_ACCESS | EVENT_MODIFY_STATE, FALSE, TEXT("ExitEvent"))

if (ghExitEvent == NULL) {
        printf("%s: Error opening event %s! Are you sure Control is running? Hit any key
        (void)_getch();
        return EXIT_FAILURE;
}

// wait for events raised by Control) or the quit event (may be set by every participant
HANDLE ghEvents[] = { ghStartProcessingEvent, ghExitEvent , ghProcessingFinishedEvent };
int nNumEvents = sizeof(ghEvents) / sizeof(ghEvents[0]);

printf("%s process is running.\n", MYNAME);

while (bContinue) {
                dwEvent = WaitForMultipleObjects(
                nNumEvents,     // number of objects to wait for
                ghEvents,       // array of objects to wait for
                FALSE,          // wait for any object ...
                INFINITE);      // ...and for a very long time

        switch (dwEvent) // return value indicates, which event is signalled now
        {
        case WAIT_OBJECT_0 + 0: // StartProcessingEvent
                        // Some processing..
                        for (i = 0; i < 9000000; i++) { vi = i % 2; }

                        // Set ProcessingFinishedEvent
                        if (!SetEvent(ghProcessingFinishedEvent)) {
```

```
                    printf("%s: Error setting event %s! Hit any key...\n\n",
                    (void)_getch();
                    return 1;
                }
                break;
        case WAIT_OBJECT_0 + 1: // ExitEvent
            bContinue = FALSE;
            break;
        case WAIT_TIMEOUT:
            // no event but timeout
            break;
        default:
            // invalid return value: Error
            printf("%s: Wait error: %d\n", MYNAME, GetLastError());
            ExitProcess(0);
        }

    }

    printf("\n%s: Process stopped. Hit any key to close Window.\n", MYNAME);
    (void)_getch();
    ExitProcess(0);
}
```

Complete the project InputOutput

- The Meilhaus IO card is already included in the start project.
- Create events and event handling.
- Events should be named uniquely.
- This process already includes the Meilhaus driver: IRQ-trigger (Pin48) and digital Output (Pin 29) of the Meilhaus ME-4660 IO card.
- This process has to inform the control process about a new trigger. As soon, as a trigger is detected at the digital input of the IO card, the interrupt function IrqCallback() will be called. Here you have to signal an event.
- This process recieves a command to do a digital output.

InputOutput.cpp

```
#include <iostream>
#include <windows.h>
#include <stdio.h>
#include <conio.h>
#include <cstdint>
#include <thread>

//for threads
#include <cstdint>
#include <thread>

extern "C"
{
#include "medriver.h"
}

#ifdef _WIN64

#pragma comment (lib, "meIDSmain64.lib")

#else

#pragma comment (lib, "meIDSmain.lib")

#endif


#define MYNAME "InputOutput"

// Create handles for each event
HANDLE ghInputEvent = NULL, ghOutputEvent = NULL, ghExitEvent = NULL;

BOOL bContinue = TRUE;

// Forward declaration for callback function on MEids
int _stdcall IrqCallback(int iDevice, int iSubdevice, int iChannel,
        int iIrqCount, int iValue, void* pvContext, int iErrorCode);

const int gen_Device = 0;
const int out_SubDevice = 0;
const int out_Channel = 0;
const int in_IrqSubDevice = 10;
const int in_IrqChannel = 6;
const int in_IrqPin = 48;

int i_config_flags = ME_IO_SINGLE_CONFIG_DIO_BYTE;
int i_write_flags = ME_IO_SINGLE_TYPE_DIO_BYTE;
int i_me_error;

int main()
{
	//handles for process and threads
	HANDLE hProcess = GetCurrentProcess();
	HANDLE hThread = GetCurrentThread();

	//set priority and processors
	int threadPriorityMask = THREAD_PRIORITY_NORMAL;
	DWORD processAffinityMask = 0x00000001;
	if (!SetThreadPriority(hThread, threadPriorityMask))
	{
		printf("SetThreadPriority failed process name: %s, error: %d\n", MYNAME, GetLast
		return EXIT_FAILURE;
	}
	if (!SetThreadAffinityMask(hThread, processAffinityMask))
	{
		printf("SetThreadAffinityMask failed process name: %s, error: %d\n", MYNAME, Get
		return EXIT_FAILURE;
	}


	//get info
	int processid = GetCurrentProcessId();
	int threadid = GetCurrentThreadId();
	int processornum = GetCurrentProcessorNumber();
	DWORD ppriority = GetPriorityClass(hProcess);
	int tpriority = GetThreadPriority(hThread);

	std::cout << "Process ID: " << processid << "thread ID: " << threadid << "processor numb
	DWORD dwEvent;


	// open InputEvent:
	ghInputEvent = OpenEvent(EVENT_ALL_ACCESS | EVENT_MODIFY_STATE, FALSE, TEXT("InputEvent"
	if (ghInputEvent == NULL) {
		printf("%s: Error opening event %s! Are you sure Control is running? Hit any key
		(void)_getch();
		return EXIT_FAILURE;
	}
	ghOutputEvent = OpenEvent(EVENT_ALL_ACCESS | EVENT_MODIFY_STATE, FALSE, TEXT("OutputEver
	if (ghOutputEvent == NULL) {
		printf("%s: Error opening event %s! Are you sure Control is running? Hit any key
		(void)_getch();
		return EXIT_FAILURE;
	}
	ghExitEvent = OpenEvent(EVENT_ALL_ACCESS | EVENT_MODIFY_STATE, FALSE, TEXT("ExitEvent"))

	if (ghExitEvent == NULL) {
		printf("%s: Error opening event %s! Are you sure Control is running? Hit any key
        (void)_getch();
        return EXIT_FAILURE;
}

//open ME driver system
i_me_error = meOpen(0);

if (i_me_error != ME_ERRNO_SUCCESS)
{
        if (!SetEvent(ghExitEvent))
        {
                printf("%s: Error %d SetEvent()\n", MYNAME, GetLastError());
                bContinue = FALSE;
        }
        printf("****    meOpen - Error: %d    ****\n\n", i_me_error);

        printf("Press any key to terminate\n\n");

        (void)_getch();

        return -1;
}

//reset device (to reset IRQ Counter)
i_me_error = meIOResetDevice(gen_Device,                        //Device 0
        ME_IO_RESET_DEVICE_NO_FLAGS);                   // Flags

//declare callback funktion with fixed parameters
i_me_error = meIOIrqSetCallback(gen_Device,
        in_IrqSubDevice,                                        // Subdevice inc
        IrqCallback,                    // Callback routine
        NULL,                                   // Callback contextz
        ME_IO_IRQ_SET_CALLBACK_NO_FLAGS);               // Flags

if (i_me_error != ME_ERRNO_SUCCESS)
{
        if (!SetEvent(ghExitEvent))
        {
                printf("%s: Error %d SetEvent()\n", MYNAME, GetLastError());
                bContinue = FALSE;
        }
        printf("****    meIOIrqSetCallback - Error: %d    ****\n\n", i_me_error);

        printf("Press any key to terminate\n");

        (void)_getch();

        meClose(0);

        return -1;
}
```

```
//start interrupt function PIN48
i_me_error = meIOIrqStart(gen_Device,
        in_IrqSubDevice,                                                        // Subde
        in_IrqChannel,                                                  // Channel index
        ME_IRQ_SOURCE_DIO_LINE,         // IRQ source
        ME_IRQ_EDGE_RISING,                     // Type of signal edge
        ME_VALUE_NOT_USED,                      // Additional argument - Not required he
        ME_IO_IRQ_START_NO_FLAGS);      // Flags

if (i_me_error != ME_ERRNO_SUCCESS)
{
        if (!SetEvent(ghExitEvent))
        {
                printf("%s: Error %d SetEvent()\n", MYNAME, GetLastError());
                bContinue = FALSE;
        }
        printf("****    meIOIrqStart - Error: %d    ****\n\n", i_me_error);

        printf("Press any key to terminate\n");

        (void)_getch();

        meClose(0);

        return 1;
}

//  Configure the output device

i_me_error = meIOSingleConfig(gen_Device,                               // Device index
        out_SubDevice,
        out_Channel,
        ME_SINGLE_CONFIG_DIO_OUTPUT,                                            // Singl
        ME_VALUE_NOT_USED,
        ME_TRIG_CHAN_DEFAULT,
        ME_TRIG_TYPE_NONE,
        ME_VALUE_NOT_USED,
        i_config_flags);        // Flags


if (i_me_error != ME_ERRNO_SUCCESS)
{
        if (!SetEvent(ghExitEvent))
        {
                printf("%s: Error %d SetEvent()\n", MYNAME, GetLastError());
                bContinue = FALSE;
        }
        printf("****    meIOSingleConfig - Error (output config): %d    ****\n\n", i_me_

        printf("Press any key to terminate\n");
        (void)_getch();

        meClose(0);

        return -1;
}

//Create output array
meIOSingle_t io_single[1];

io_single[0].iDevice = gen_Device;
io_single[0].iSubdevice = out_SubDevice;
io_single[0].iChannel = out_Channel;
io_single[0].iDir = ME_DIR_OUTPUT;
io_single[0].iValue = 0;
io_single[0].iTimeOut = 0;                                              // No ti
io_single[0].iFlags = i_write_flags;


// wait for StartProcessing command (raised by the ececution control) or the quit event
HANDLE ghEvents[] = { ghOutputEvent, ghExitEvent };
int nNumEvents = sizeof(ghEvents) / sizeof(ghEvents[0]);

printf("%s process is running.\n", MYNAME);
printf("Press the ESC key at any time to stop this process\n");

while (bContinue) {
        dwEvent = WaitForMultipleObjects(
                nNumEvents,     // number of objects to wait for
                ghEvents,       // array of objects to wait for
                FALSE,          // wait for any object ...
                2);     // ... timeout after 2 ms

        switch (dwEvent) // return value indicates, which event is signalled now
        {
        case WAIT_OBJECT_0 + 0: // OutputEvent
                //Set Output to high level
                io_single[0].iValue |= (1 << 0);
                i_me_error = meIOSingle(&io_single[0],                  // Outpu
                        1,                                              // Numbe
                        ME_IO_SINGLE_NO_FLAGS); // Flags

                if (i_me_error != ME_ERRNO_SUCCESS)
                {
                        //printf("****    meIOSingle (output) - Error: %d    ****\n\n",

                        //printf("Press any key to terminate\n");

                        (void)_getch();
                        meClose(0);

                        return -1;
                }
                break;

        case WAIT_OBJECT_0 + 1: // ExitEvent
                printf("%s: I got the event %s!\n\n", MYNAME, "ExitEvent");
                bContinue = FALSE;
                break;

        case WAIT_TIMEOUT:
                // timeout after Set output low

                io_single[0].iValue &= ~(1 << 0); //Clear Bit 0 , for Bit 2 (1 << 2) etc
                i_me_error = meIOSingle(&io_single[0],                           // Outpu
                        1,                                                       // Numbe
                        ME_IO_SINGLE_NO_FLAGS); // Flags

                if (i_me_error != ME_ERRNO_SUCCESS)
                {
                        if (!SetEvent(ghExitEvent))
                        {
                                printf("%s: Error %d SetEvent()\n", MYNAME, GetLastError
                                bContinue = FALSE;
                        }
                        printf("****    meIOSingle (output) - Error: %d    ****\n\n", i_

                        printf("Press any key to terminate\n");

                        (void)_getch();

                        meClose(0);

                        return -1;
                }
                break;
        default:
                // invalid return value: Error
                printf("%s: Wait error: %d\n", MYNAME, GetLastError());
                ExitProcess(0);
        }

        if (_kbhit() != 0)
        {
                if (_getch() == 27)
                {
                        if (!SetEvent(ghExitEvent))
                        {
                                printf("%s: Error %d SetEvent()\n", MYNAME, GetLastError
                                        bContinue = FALSE;
                                }
                                break;
                        }
                }
        }

        printf("\n%s: Process stopped. Hit any key to close Window.\n", MYNAME);
        (void)_getch();

        //close callback funktion
        i_me_error = meIOIrqStop(0,                                                     // Devic
                10,                                             // Subdevice index
                0,                                                      // Channel index
                ME_IO_IRQ_STOP_NO_FLAGS);       // Flags

        if (i_me_error != ME_ERRNO_SUCCESS)
        {
                //printf("****    meIOIrqStop - Error: %d    ****\n\n", i_me_error);

                //printf("Press any key to terminate\n");

                (void)_getch();

                meClose(0);

                return(-1);
        }

        //Close ME driver System
        meClose(0);

        ExitProcess(0);
}


//Callback function from ME driver

int _stdcall IrqCallback(int iDevice, int iSubdevice, int iChannel,
        int iIrqCount, int iValue, void* pvContext, int iErrorCode)
{
        if (iErrorCode == ME_ERRNO_SUCCESS)
        {
                //printf("Interrupt: IRQ Count: %d  Value: %d\n", iIrqCount, iValue);
                if (!SetEvent(ghInputEvent))
                {
                        printf("%s: Error %d SetEvent()\n", MYNAME, GetLastError());
                        bContinue = FALSE;
                }

        }
        else
        {
                if (!SetEvent(ghExitEvent))
                {
                        printf("%s: Error %d SetEvent()\n", MYNAME, GetLastError());
                        bContinue = FALSE;
                }
                printf("****    IrqCallback - Error: %d    ****\n\n", iErrorCode);
        }

        return(0);
}
```

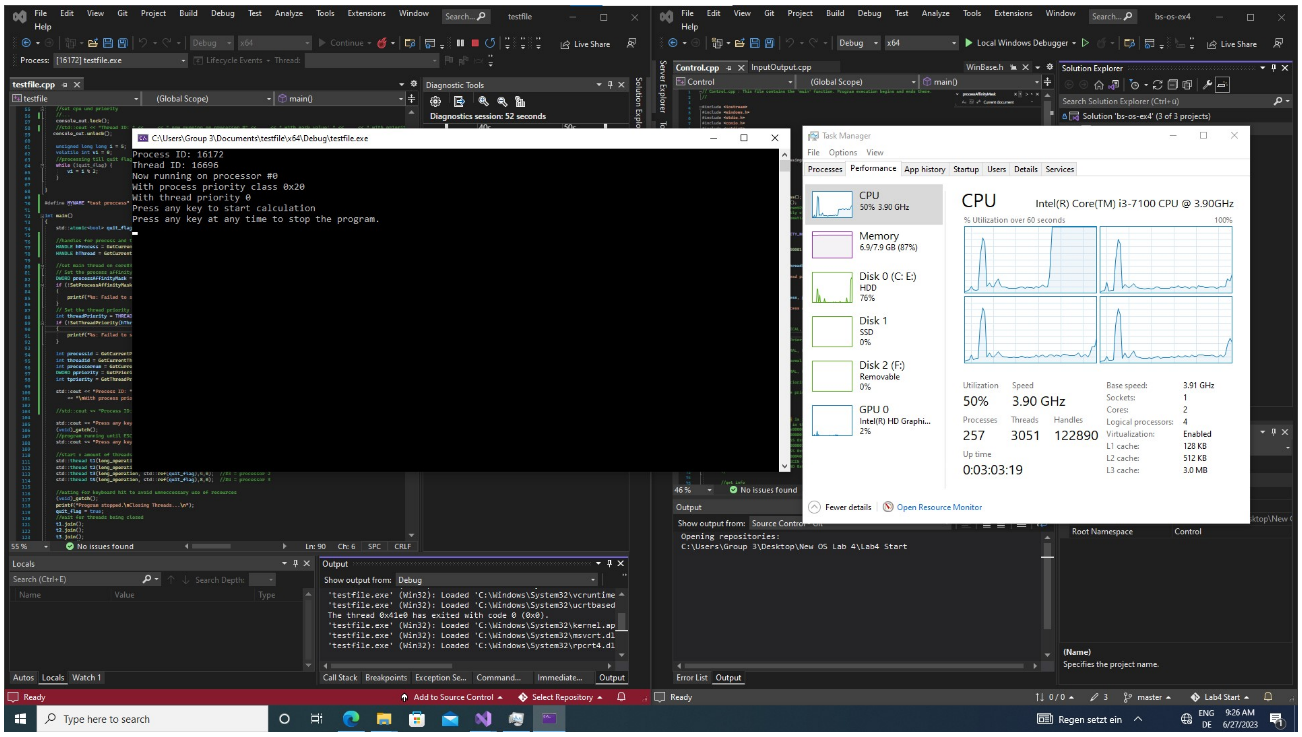
Process priorities and processor assignment

1. Add some code to show the current priority and the assigned processor core of your processes.

testfile.cpp

```
#include <iostream>
#include <stdio.h>
#include <conio.h>
#include <chrono>
#include <cstdint>
#include <thread>
#include <Windows.h>
#include <WinBase.h>
#include <mutex>
#include <processthreadsapi.h>


std::mutex console_out;

void long_operation(std::atomic<bool>& quit_flag, int core, int nPriority)
{
    //handles for process and threads
    HANDLE hProcess = GetCurrentProcess();
    HANDLE hThread = GetCurrentThread();
    console_out.lock();
    //std::cout << "Thread ID: " << ... << " created. Running on processor #" << ... << " with p
    std::cout << "Thread ID: " << GetCurrentThreadId() << " created. Running on processor #" <<
    console_out.unlock();
    //set cpu und priority
    // Set the process affinity mask
    DWORD processAffinityMask = 0x00000001 << core; // Run on the first processor by default
    if (!SetProcessAffinityMask(hProcess, processAffinityMask))
    {
        printf("Failed to set process affinity mask. Error code: %d\n", GetLastError());
    }
    // Set the thread priority
    int threadPriority = nPriority;
    if (!SetThreadPriority(hThread, threadPriority))
    {
        printf("Failed to set thread priority. Error code: %d\n", GetLastError());
    }
    //get current settings
    int processid = GetCurrentProcessId();
    int threadid = GetCurrentThreadId();
    int processornum = GetCurrentProcessorNumber();
    DWORD ppriority = GetPriorityClass(hProcess);
    int tpriority = GetThreadPriority(hThread);

    console_out.lock();
    //std::cout << "Thread ID: " << ... << " now running on processor #" << ... << " with mask v
    std::cout << "Thread ID: " << threadid << " now running on processor #" << processornum << "
    console_out.unlock();
    unsigned long long i = 5;
    volatile int vi = 0;
    //processing till quit flag is raised...
    while (!quit_flag) {
        vi = i % 2;
    }

}

#define MYNAME "test proccess"

int main()
{
    std::atomic<bool> quit_flag(false);

    //handles for process and threads
    HANDLE hProcess = GetCurrentProcess();
    HANDLE hThread = GetCurrentThread();

    //set main thread on core 3 with priority 0
    // Set the process affinity mask
    DWORD processAffinityMask = 0x00000001; // Run on the first processor by default
    if (!SetProcessAffinityMask(hProcess, processAffinityMask))
    {
        printf("%s: Failed to set process affinity mask. Error code: %d\n", MYNAME, GetLastError
    }
    // Set the thread priority
    int threadPriority = THREAD_PRIORITY_ABOVE_NORMAL;
    if (!SetThreadPriority(hThread, threadPriority))
    {
        printf("%s: Failed to set thread priority. Error code: %d\n", MYNAME, GetLastError());
    }

    int processid = GetCurrentProcessId();
    int threadid = GetCurrentThreadId();
    int processornum = GetCurrentProcessorNumber();
    DWORD ppriority = GetPriorityClass(hProcess);
    int tpriority = GetThreadPriority(hThread);

    std::cout << "Process ID: " << processid << "\nThread ID: " << threadid << "\nNow running or
        << "\nWith process priority class 0x" << std::hex << ppriority << "\nWith thread priorit


    std::cout << "Press any key to start calculation" << std::endl;
    (void)_getch();
    //program running until ESC
    std::cout << "Press any key at any time to stop the program." << std::endl;

    //start x amount of threads
    std::thread t1(long_operation, std::ref(quit_flag),1,0);  //#1 = processor 0
    std::thread t2(long_operation, std::ref(quit_flag),2,0);  //#2 = processor 1
    std::thread t3(long_operation, std::ref(quit_flag),4,0);  //#3 = processor 2
    std::thread t4(long_operation, std::ref(quit_flag),8,0);  //#4 = processor 3

    //wating for keyboard hit to avoid unneccessary use of recources
    (void)_getch();
    printf("Program stopped.\nClosing Threads...\n");
    quit_flag = true;
    //wait for threads being closed
    t1.join();
    t2.join();
    t3.join();
    t4.join();
    return 0;
}
```

2. You will find a computational intensive program: Studio project testfile.sln.
   Add some code to get 3 threads, each assigned to a different processor core.
   Note: You can observe the workload of each processor core by running the windows resource monitor:
   a. Start your program via batch file start.bat first (which processor hosts your processes?) and your test program testfile.prg.
   Observe the program's behavior.
   b. Start testfile.prg first and then your program via start.bat.
   Which processor hosts your processes?



```
C:\Users\Group 3\Documents\testfile\x64\Debug\testfile.exe
Process ID: 16172
Thread ID: 16696
Now running on processor #0
With process priority class 0x20
With thread priority 0
Press any key to start calculation
Press any key at any time to stop the program.
```

```
'testfile.exe' (Win32): Loaded 'C:\Windows\System32\vcruntime
'testfile.exe' (Win32): Loaded 'C:\Windows\System32\ucrtbased
The thread 0x41e0 has exited with code 0 (0x0).
'testfile.exe' (Win32): Loaded 'C:\Windows\System32\kernel.ap
'testfile.exe' (Win32): Loaded 'C:\Windows\System32\msvcrt.dl
'testfile.exe' (Win32): Loaded 'C:\Windows\System32\rpcrt4.dl
```

```
Opening repositories:
C:\Users\Group 3\Desktop\New OS Lab 4\Lab4 Start
```

CPU Intel(R) Core(TM) i3-7100 CPU @ 3.90GHz

Utilization	Speed		Base speed:	3.91 GHz
50%	3.90 GHz		Sockets:	1
Processes	Threads	Handles	Cores:	2
257	3051	122890	Logical processors:	4
Up time			Virtualization:	Enabled
0:03:03:19			L1 cache:	128 KB
			L2 cache:	512 KB
			L3 cache:	3.0 MB

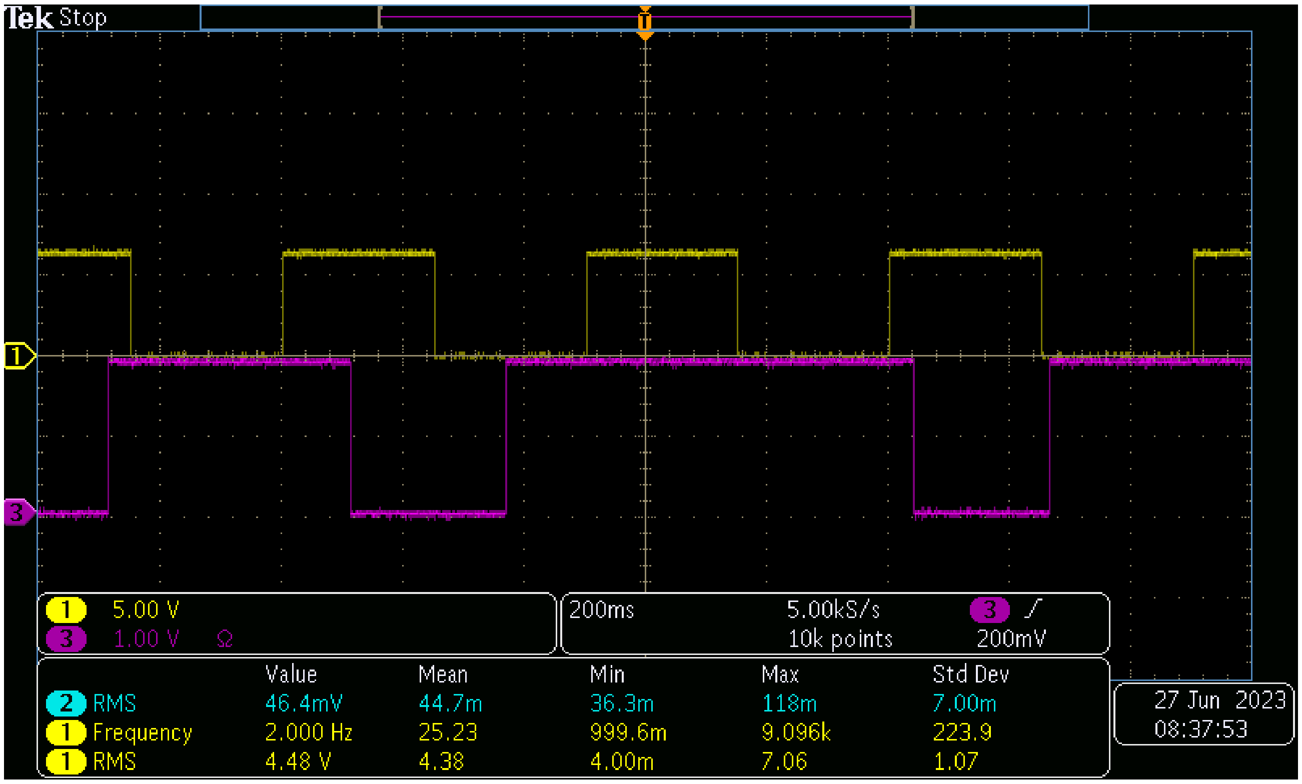
Fig.1 - Task Manager Core Usage

Observe the program’s behavior. c. Switch between the 4 console windows by using the mouse. Describe the effect to the signal output.
the output is being affected by the switching between the consoles, we presume that windows is giving the console with the focus more recources than the others as said on the lecture.

1. Repeat the tasks a) b) and c); now with 4 threads in your test program.

a. Start your processes with increased priority (ABOVE_NORMAL_PRIORITY_CLASS) via start.bat first and then start the test program (same priority as above).
Observe the program’s behavior. Switch between the 4 console windows by using the mouse. Describe the effect to the signal output.

Above normal priority class: 0x00008000

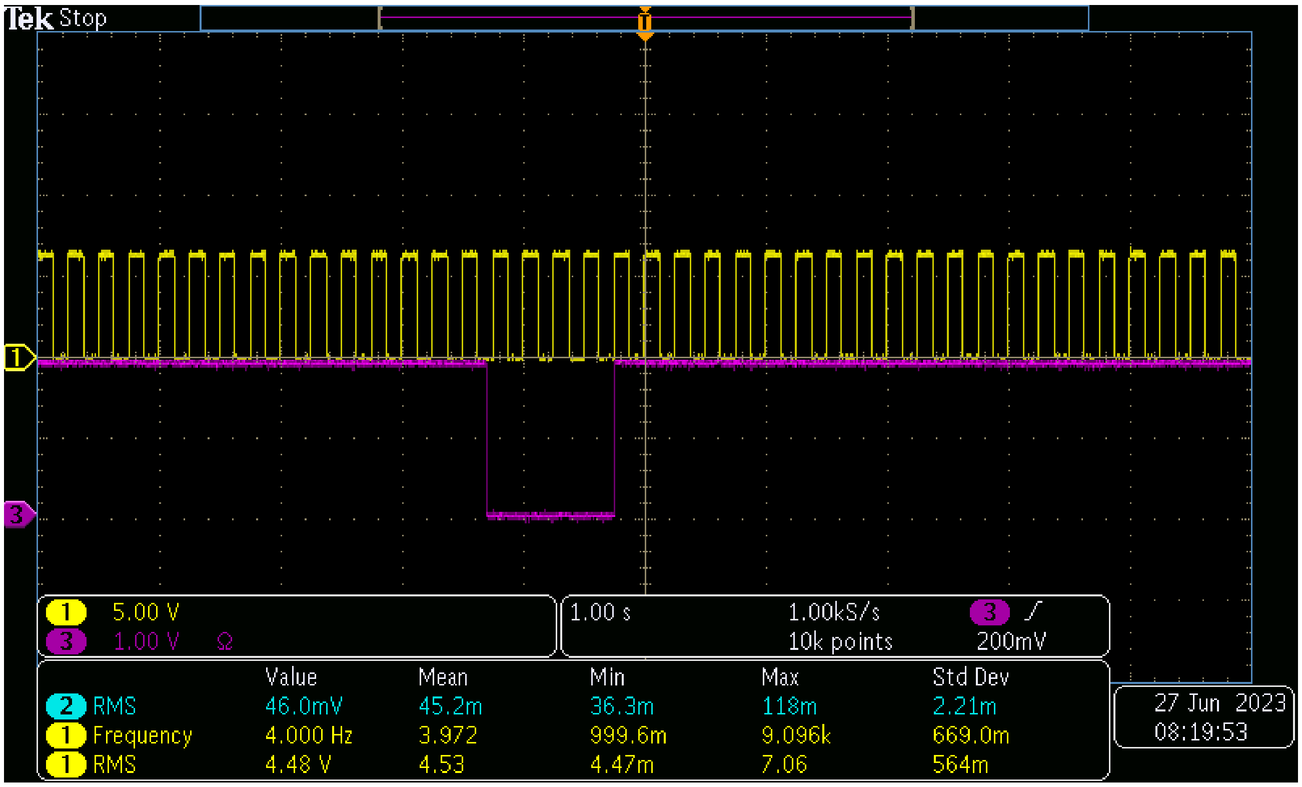


	Value	Mean	Min	Max	Std Dev
2 RMS	46.4mV	44.7m	36.3m	118m	7.00m
1 Frequency	2.000 Hz	25.23	999.6m	9.096k	223.9
1 RMS	4.48 V	4.38	4.00m	7.06	1.07

Fig.1 - Oscilloscope output

b. Start the test program with lower priority (BELOW_NORMAL_PRIORITY_CLASS) first and then your program via start.bat. Observe the program’s behavior. Switch between the 4 console

LOWEST_PRIORITY_CLASS: 0x00004000

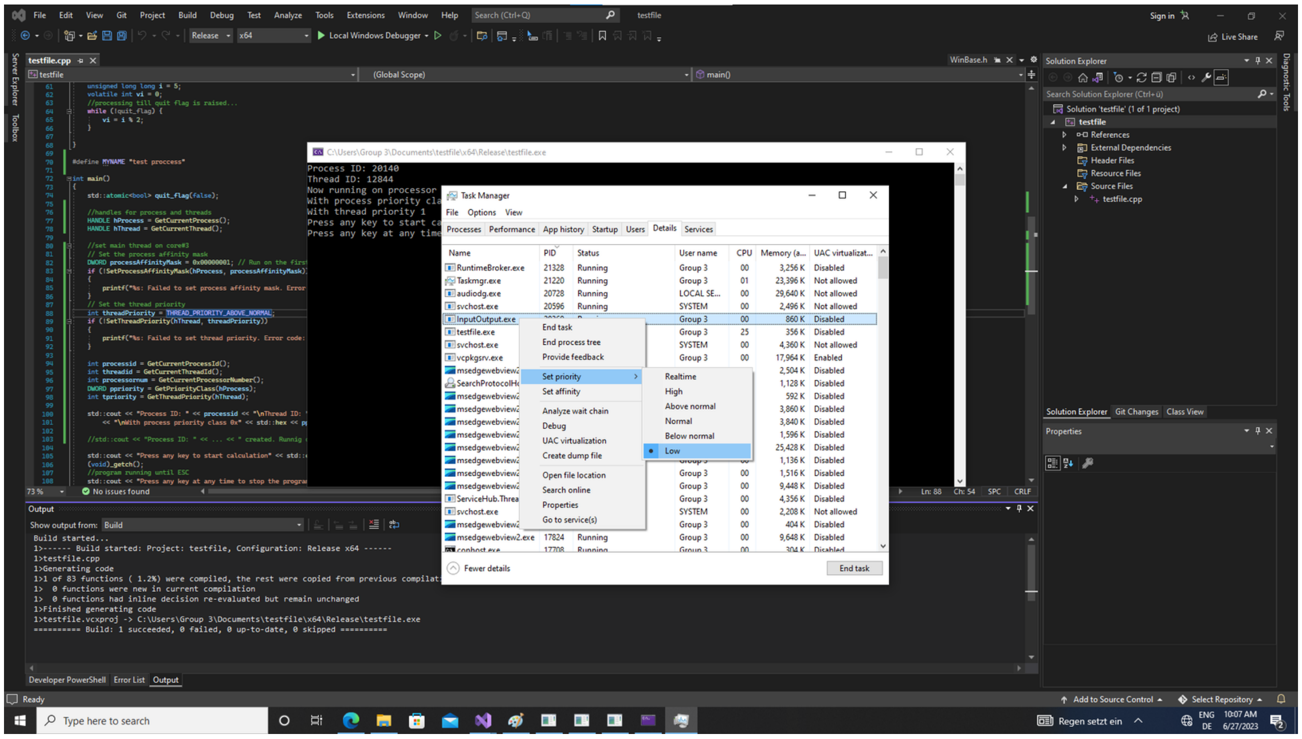


		Value	Mean	Min	Max	Std Dev
2	RMS	46.0mV	45.2m	36.3m	118m	2.21m
1	Frequency	4.000 Hz	3.972	999.6m	9.096k	669.0m
1	RMS	4.48 V	4.53	4.47m	7.06	564m

Fig.1 - Oscilloscope output

1. You can also change priorities and processor assignment with the Windows task manager. Test different priorities and assignments.

Task Manager Setting Priority to low for Input Output/ HIGH for testfile



```
    unsigned long long i = 5;
    volatile int vi = 0;
    //processing till quit flag is raised...
    while (!quit_flag) {
        vi = i % 2;
    }

}

#define MYNAME "test proccess"

int main()
{
    std::atomic<bool> quit_flag(false);

    //handles for process and threads
    HANDLE hProcess = GetCurrentProcess();
    HANDLE hThread = GetCurrentThread();

    //set main thread on core#3
    // Set the process affinity mask
    DWORD processAffinityMask = 0x00000001; // Run on the first
    if (!SetProcessAffinityMask(hProcess, processAffinityMask))
    {
        printf("%s: Failed to set process affinity mask. Error
    }
    // Set the thread priority
    int threadPriority = THREAD_PRIORITY_ABOVE_NORMAL;
    if (!SetThreadPriority(hThread, threadPriority))
    {
        printf("%s: Failed to set thread priority. Error code:
    }

    int processid = GetCurrentProcessId();
    int threadid = GetCurrentThreadId();
    int processornum = GetCurrentProcessorNumber();
    DWORD ppriority = GetPriorityClass(hProcess);
    int tpriority = GetThreadPriority(hThread);

    std::cout << "Process ID: " << processid << "\nThread ID: "
        << "\nWith process priority class 0x" << std::hex << pp

    //std::cout << "Process ID: " << ... << " created. Runnig

    std::cout << "Press any key to start calculation" << std::
    (void)_getch();
    //program running until ESC
    std::cout << "Press any key at any time to stop the program
```

C:\Users\Group 3\Documents\testfile\x64\Release\testfile.exe

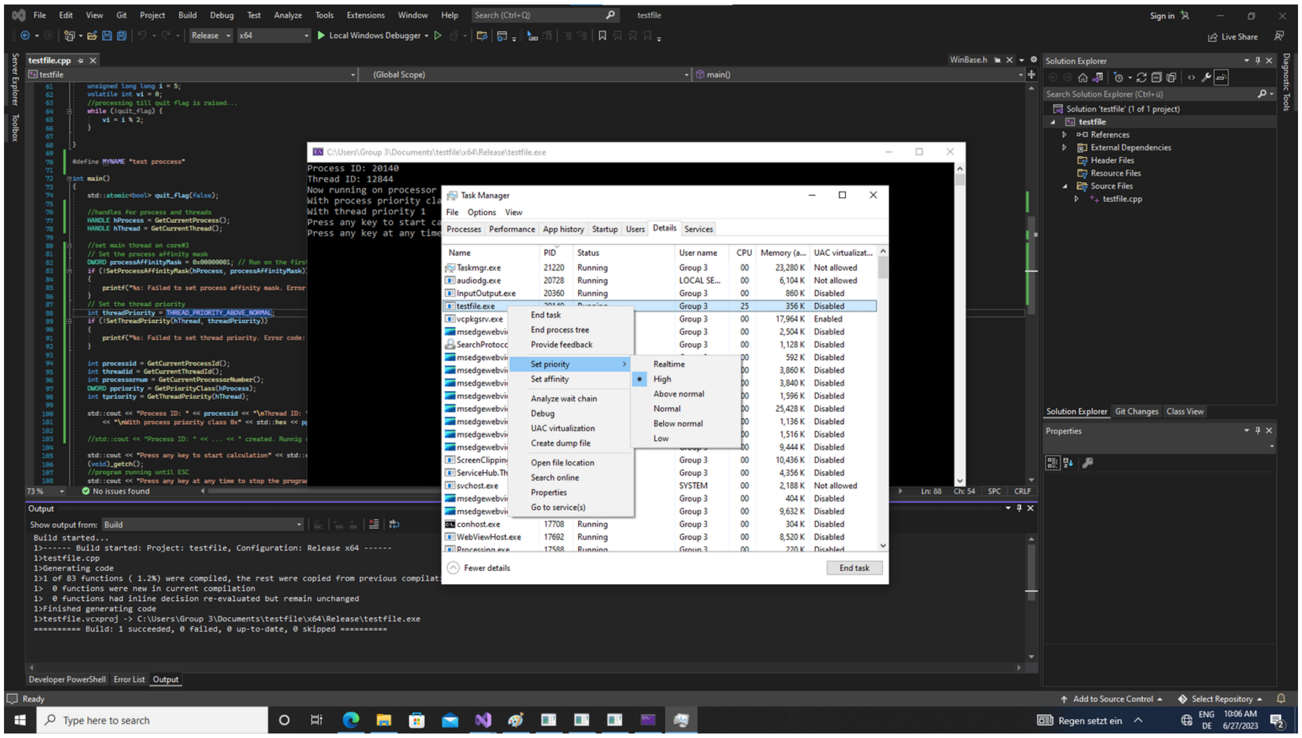
```
Process ID: 20140
Thread ID: 12844
Now running on processor
With process priority cla
With thread priority 1
Press any key to start ca
Press any key at any time
```

Task Manager

Name	PID	Status	User name	CPU	Memory (a...	UAC virtualizat...
RuntimeBroker.exe	21328	Running	Group 3	00	3,256 K	Disabled
Taskmgr.exe	21220	Running	Group 3	01	23,396 K	Not allowed
audiodg.exe	20728	Running	LOCAL SE...	00	29,640 K	Not allowed
svchost.exe	20596	Running	SYSTEM	00	2,496 K	Not allowed
InputOutput.exe			Group 3	00	860 K	Disabled
testfile.exe			Group 3	25	356 K	Disabled
svchost.exe			SYSTEM	00	4,360 K	Not allowed
vcpkgsrv.exe			Group 3	00	17,964 K	Enabled
msedgewebview2					2,504 K	Disabled
SearchProtocolHo					1,128 K	Disabled
msedgewebview2					592 K	Disabled
msedgewebview2					3,860 K	Disabled
msedgewebview2					3,840 K	Disabled
msedgewebview2					1,596 K	Disabled
msedgewebview2					25,428 K	Disabled
msedgewebview2					1,136 K	Disabled
msedgewebview2			Group 3	00	1,516 K	Disabled
msedgewebview2			Group 3	00	9,448 K	Disabled
ServiceHub.Threa			Group 3	00	4,356 K	Disabled
svchost.exe			SYSTEM	00	2,208 K	Not allowed
msedgewebview2			Group 3	00	404 K	Disabled
msedgewebview2.exe	17824	Running	Group 3	00	9,648 K	Disabled
conhost.exe	17708	Running	Group 3	00	304 K	Disabled

```
Build started...
1>------ Build started: Project: testfile, Configuration: Release x64 ------
1>testfile.cpp
1>Generating code
1>1 of 83 functions ( 1.2%) were compiled, the rest were copied from previous compilat
1>  0 functions were new in current compilation
1>  0 functions had inline decision re-evaluated but remain unchanged
1>Finished generating code
1>testfile.vcxproj -> C:\Users\Group 3\Documents\testfile\x64\Release\testfile.exe
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

Fig.1 - Task Manager Setting Priority to low for Input Output



```cpp
    unsigned long long i = 5;
    volatile int vi = 0;
    //processing till quit flag is raised...
    while (!quit_flag) {
        vi = i % 2;
    }

}

#define MYNAME "test proccess"

int main()
{
    std::atomic<bool> quit_flag(false);

    //handles for process and threads
    HANDLE hProcess = GetCurrentProcess();
    HANDLE hThread = GetCurrentThread();

    //set main thread on core#3
    // Set the process affinity mask
    DWORD processAffinityMask = 0x00000001; // Run on the first
    if (!SetProcessAffinityMask(hProcess, processAffinityMask))
    {
        printf("%s: Failed to set process affinity mask. Error
    }
    // Set the thread priority
    int threadPriority = THREAD_PRIORITY_ABOVE_NORMAL;
    if (!SetThreadPriority(hThread, threadPriority))
    {
        printf("%s: Failed to set thread priority. Error code:
    }

    int processid = GetCurrentProcessId();
    int threadid = GetCurrentThreadId();
    int processornum = GetCurrentProcessorNumber();
    DWORD ppriority = GetPriorityClass(hProcess);
    int tpriority = GetThreadPriority(hThread);

    std::cout << "Process ID: " << processid << "\nThread ID: 
        << "\nWith process priority class 0x" << std::hex << pp

    //std::cout << "Process ID: " << ... << " created. Runnig 

    std::cout << "Press any key to start calculation" << std::
    (void)_getch();
    //program running until ESC
    std::cout << "Press any key at any time to stop the progra
```

C:\Users\Group 3\Documents\testfile\x64\Release\testfile.exe

```
Process ID: 20140
Thread ID: 12844
Now running on processor
With process priority cla
With thread priority 1
Press any key to start ca
Press any key at any time
```

Task Manager

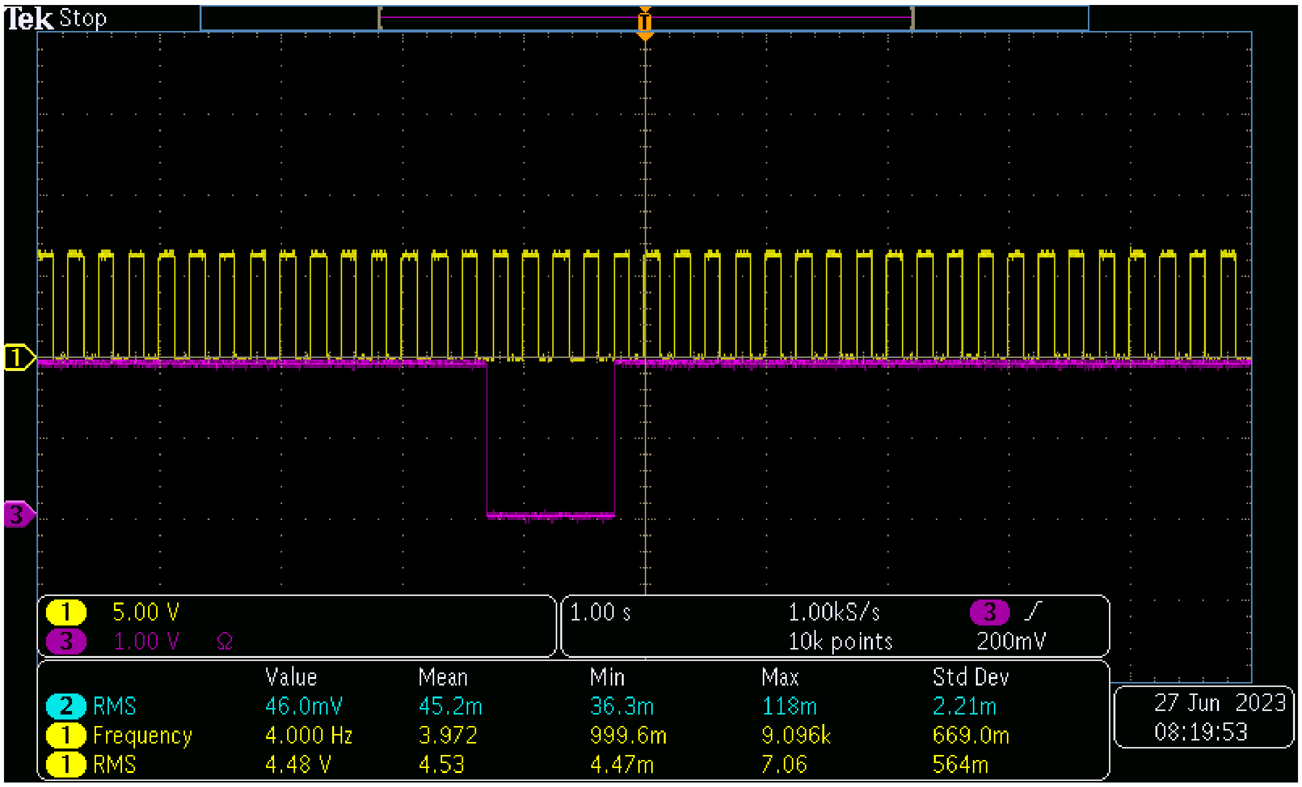
Name	PID	Status	User name	CPU	Memory (a...	UAC virtualizat...
Taskmgr.exe	21220	Running	Group 3	00	23,280 K	Not allowed
audiodg.exe	20728	Running	LOCAL SE...	00	6,104 K	Not allowed
InputOutput.exe	20360	Running	Group 3	00	860 K	Disabled
testfile.exe	[illegible]	[illegible]	Group 3	25	356 K	Disabled
vcpkgsrv.exe			Group 3	00	17,964 K	Enabled
msedgewebvi			Group 3	00	2,504 K	Disabled
SearchProtoc			Group 3	00	1,128 K	Disabled
msedgewebvi					592 K	Disabled
msedgewebvi					3,860 K	Disabled
msedgewebvi					3,840 K	Disabled
msedgewebvi					1,596 K	Disabled
msedgewebvi					25,428 K	Disabled
msedgewebvi					1,136 K	Disabled
msedgewebvi					1,516 K	Disabled
msedgewebvi				00	9,444 K	Disabled
ScreenClippin			Group 3	00	10,436 K	Disabled
ServiceHub.Th			Group 3	00	4,356 K	Disabled
svchost.exe			SYSTEM	00	2,188 K	Not allowed
msedgewebvi			Group 3	00	404 K	Disabled
msedgewebvi			Group 3	00	9,632 K	Disabled
conhost.exe	17708	Running	Group 3	00	304 K	Disabled
WebViewHost.exe	17692	Running	Group 3	00	8,520 K	Disabled
Processing.exe	17588	Running	Group 3	00	220 K	Disabled

Set priority: Realtime, [✓ High], Above normal, Normal, Below normal, Low

Output — Show output from: Build

```
Build started...
1>------ Build started: Project: testfile, Configuration: Release x64 ------
1>testfile.cpp
1>Generating code
1>1 of 83 functions ( 1.2%) were compiled, the rest were copied from previous compilat
1>  0 functions were new in current compilation
1>  0 functions had inline decision re-evaluated but remain unchanged
1>Finished generating code
1>testfile.vcxproj -> C:\Users\Group 3\Documents\testfile\x64\Release\testfile.exe
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

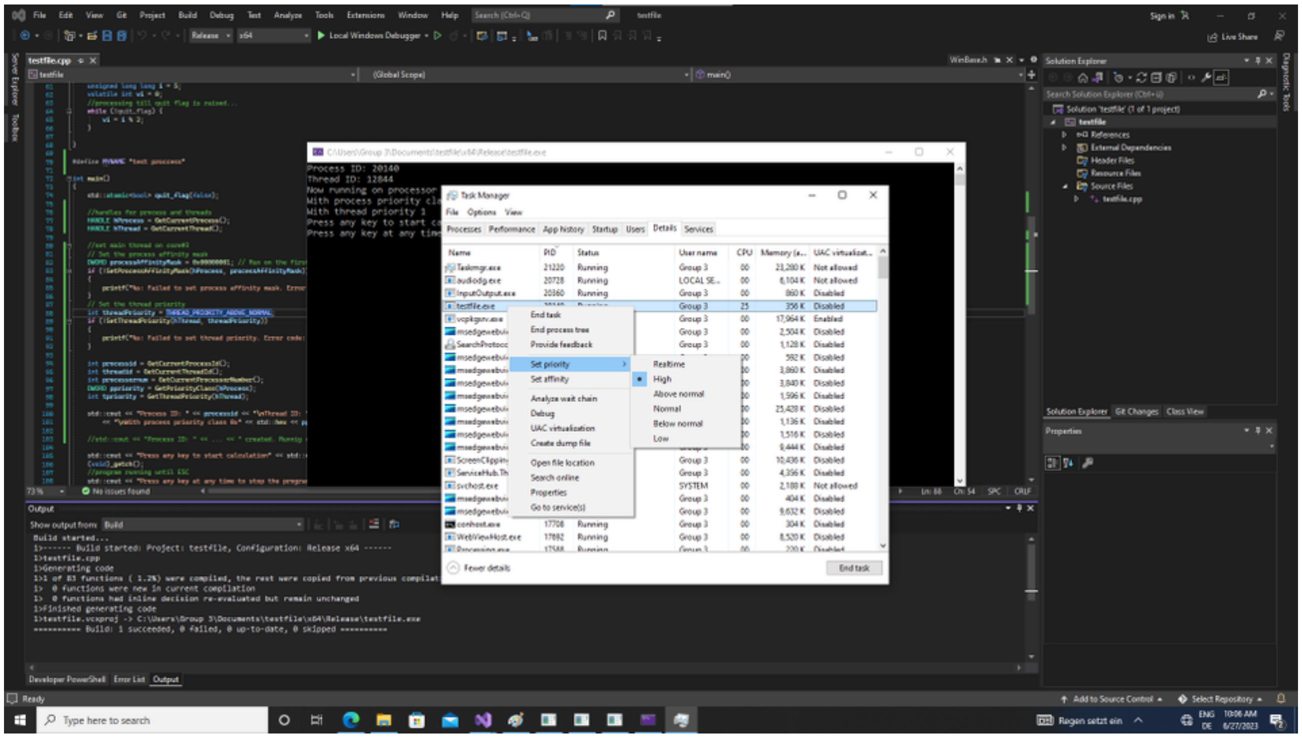
Fig.1 - Task Manager Setting Priority to high for testfile



		Value	Mean	Min	Max	Std Dev
2	RMS	46.0mV	45.2m	36.3m	118m	2.21m
1	Frequency	4.000 Hz	3.972	999.6m	9.096k	669.0m
1	RMS	4.48 V	4.53	4.47m	7.06	564m

Fig.1 - Oscilloscope output

Task Manager Setting Priority to high for Input Output/ LOW for testfile



```
    unsigned long long i = 5;
    volatile int vi = 0;
    //processing till quit flag is raised...
    while (!quit_flag) {
        vi = i % 2;
    }

}

#define MYNAME "test process"

int main()
{
    std::atomic<bool> quit_flag(false);

    //handles for process and threads
    HANDLE hProcess = GetCurrentProcess();
    HANDLE hThread = GetCurrentThread();

    //set main thread on core#1
    // Set the process affinity mask
    DWORD processAffinityMask = 0x00000001; // Run on the first
    if (!SetProcessAffinityMask(hProcess, processAffinityMask))
    {
        printf("%s: Failed to set process affinity mask. Error
    }
    // Set the thread priority
    int threadPriority = THREAD_PRIORITY_ABOVE_NORMAL;
    if (!SetThreadPriority(hThread, threadPriority))
    {
        printf("%s: Failed to set thread priority. Error code:
    }

    int processid = GetCurrentProcessId();
    int threadid = GetCurrentThreadId();
    int processornum = GetCurrentProcessorNumber();
    DWORD ppriority = GetPriorityClass(hProcess);
    int tpriority = GetThreadPriority(hThread);

    std::cout << "Process ID: " << processid << "\nThread ID: "
        << "\nWith process priority class 0x" << std::hex << p

    //std::cout << "Process ID: " << ... << " created. Runnig

    std::cout << "Press any key to start calculation" << std::
    (void)_getch();
    //program running until ESC
    std::cout << "Press any key at any time to stop the progra
```

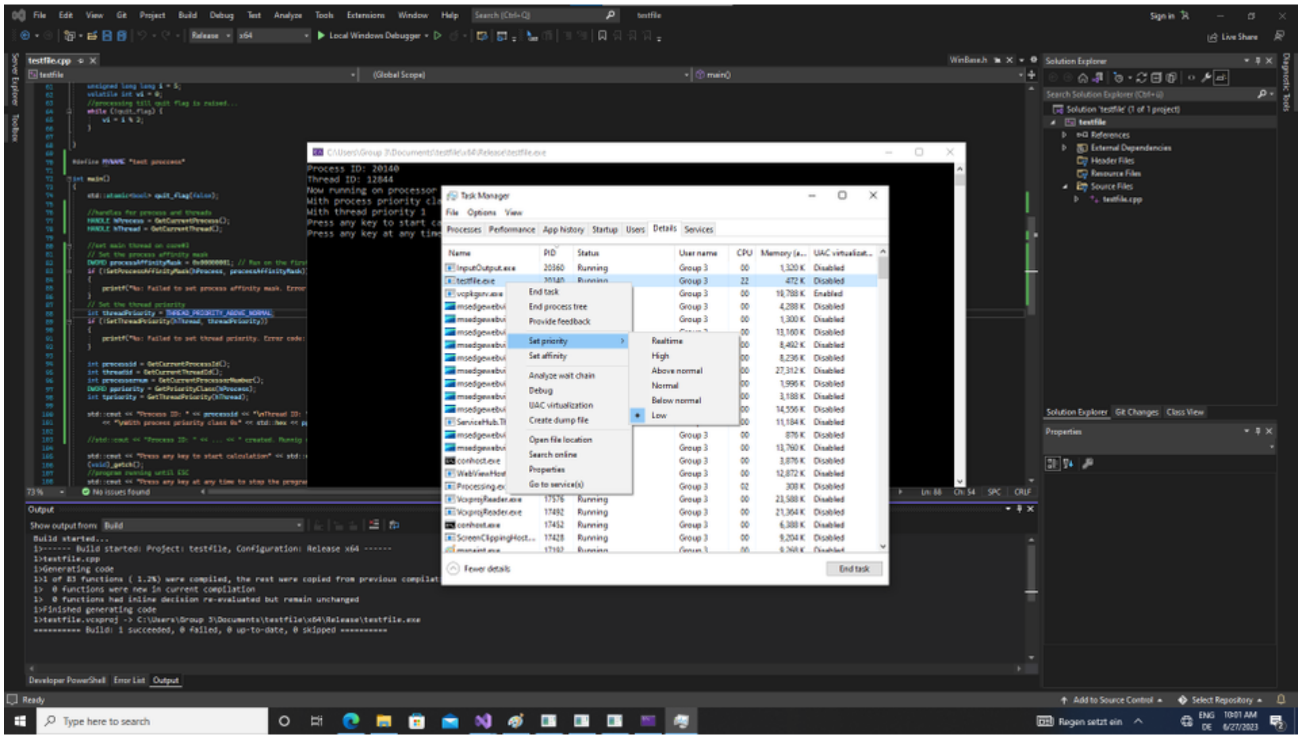
```
Process ID: 20140
Thread ID: 12844
Now running on processor
With process priority cla
With thread priority 1
Press any key to start ca
Press any key at any time
```

Name	PID	Status	User name	CPU	Memory (a...	UAC virtualizat...
Taskmgr.exe	21220	Running	Group 3	00	23,280 K	Not allowed
audiodg.exe	20728	Running	LOCAL SE...	00	6,104 K	Not allowed
InputOutput.exe	20360	Running	Group 3	00	860 K	Disabled
testfile.exe	20140	Running	Group 3	25	356 K	Disabled
vcpkgsrv.exe			Group 3	00	17,964 K	Enabled
msedgewebvi			Group 3	00	2,504 K	Disabled
SearchProtoco			Group 3	00	1,128 K	Disabled
msedgewebvi				00	592 K	Disabled
msedgewebvi				00	3,860 K	Disabled
msedgewebvi				00	3,840 K	Disabled
msedgewebvi				00	1,596 K	Disabled
msedgewebvi				00	25,428 K	Disabled
msedgewebvi				00	1,136 K	Disabled
msedgewebvi				00	1,516 K	Disabled
msedgewebvi				00	9,444 K	Disabled
ScreenClippin			Group 3	00	10,436 K	Disabled
ServiceHub.Th			Group 3	00	4,356 K	Disabled
svchost.exe			SYSTEM	00	2,188 K	Not allowed
msedgewebvi			Group 3	00	404 K	Disabled
msedgewebvi			Group 3	00	9,632 K	Disabled
conhost.exe	17708	Running	Group 3	00	304 K	Disabled
WebViewHost.exe	17892	Running	Group 3	00	8,520 K	Disabled
Processing.exe	17588	Running	Group 3	00	220 K	Disabled

Set priority: Realtime / [✓ High] / Above normal / Normal / Below normal / Low

```
Build started...
1>------ Build started: Project: testfile, Configuration: Release x64 ------
1>testfile.cpp
1>Generating code
1>1 of 83 functions ( 1.2%) were compiled, the rest were copied from previous compilat
1>  0 functions were new in current compilation
1>  0 functions had inline decision re-evaluated but remain unchanged
1>Finished generating code
1>testfile.vcxproj -> C:\Users\Group 3\Documents\testfile\x64\Release\testfile.exe
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

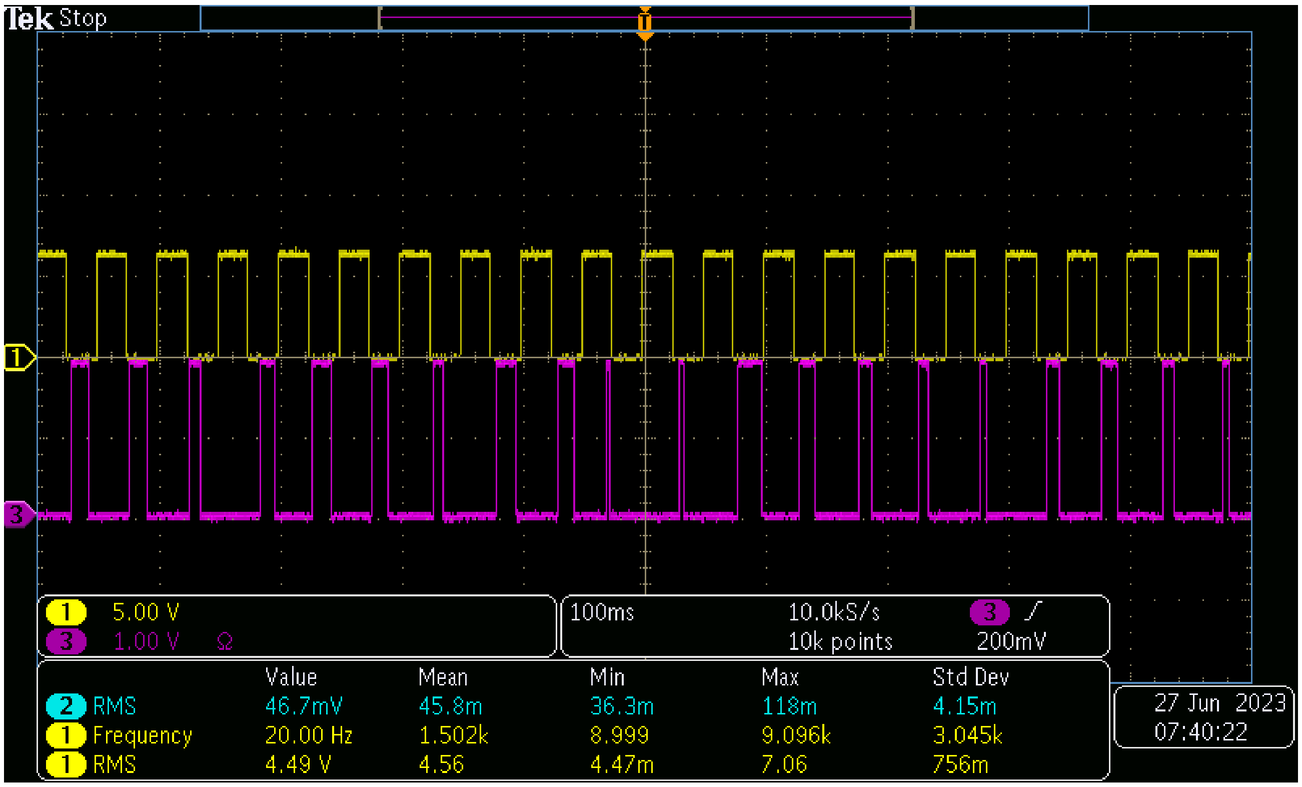
Fig.1 - Task Manager Setting Priority to high for Input Output



Name	PID	Status	User name	CPU	Memory (a...	UAC virtualizat...
InputOutput.exe	20360	Running	Group 3	00	1,320 K	Disabled
testfile.exe	20140	Running	Group 3	22	472 K	Disabled
vcpkgsrv.exe			Group 3	00	18,788 K	Enabled
msedgewebvi			Group 3	00	4,288 K	Disabled
msedgewebvi			Group 3	00	1,300 K	Disabled
msedgewebvi				00	13,160 K	Disabled
msedgewebvi				00	8,492 K	Disabled
msedgewebvi				00	8,236 K	Disabled
msedgewebvi				00	27,312 K	Disabled
msedgewebvi				00	1,996 K	Disabled
msedgewebvi				00	3,188 K	Disabled
msedgewebvi				00	14,556 K	Disabled
ServiceHub.Th				00	11,184 K	Disabled
msedgewebvi			Group 3	00	876 K	Disabled
msedgewebvi			Group 3	00	13,760 K	Disabled
conhost.exe			Group 3	00	3,876 K	Disabled
WebViewHost			Group 3	00	12,872 K	Disabled
Processing.ex			Group 3	02	308 K	Disabled
VcxprojReader.exe	17576	Running	Group 3	00	23,588 K	Disabled
VcxprojReader.exe	17492	Running	Group 3	00	21,364 K	Disabled
conhost.exe	17452	Running	Group 3	00	6,388 K	Disabled
ScreenClippingHost...	17428	Running	Group 3	00	9,204 K	Disabled
mspaint.exe	17192	Running	Group 3	00	9,268 K	Disabled

Set priority: Realtime / High / Above normal / Normal / Below normal / [✓ Low]

Fig.1 - Task Manager Setting Priority to low for testfile



```
1  5.00 V                100ms        10.0kS/s       3  ∫
3  1.00 V   Ω                         10k points      200mV

                 Value     Mean      Min       Max       Std Dev
2 RMS            46.7mV    45.8m     36.3m     118m      4.15m
1 Frequency      20.00 Hz  1.502k    8.999     9.096k    3.045k
1 RMS            4.49 V    4.56      4.47m     7.06      756m

27 Jun  2023
07:40:22
```

Fig.1 - Oscilloscope output