Министерство науки и высшего образования Российской Федерации

Новосибирский государственный технический университет

Индивидуальная работа по курсу

«Управление ресурсами в вычислительных системах»

Управления ресурсами в ОС Windows

Факультет: прикладной математики и информатики

Группа: ПМ-63

Студент: Кожекин М.В.

Вариант: 5

Преподаватели: Стасышин В. М.

Хайленко Е. А.

Новосибирск

2019

1. Задание

Разработка простейшего Windows-приложения определяющего максимальную ширину экрана в пикселях и поддержку процессором технологии MMX.

Выбран 4 уровень ИР. Данное приложении должно:

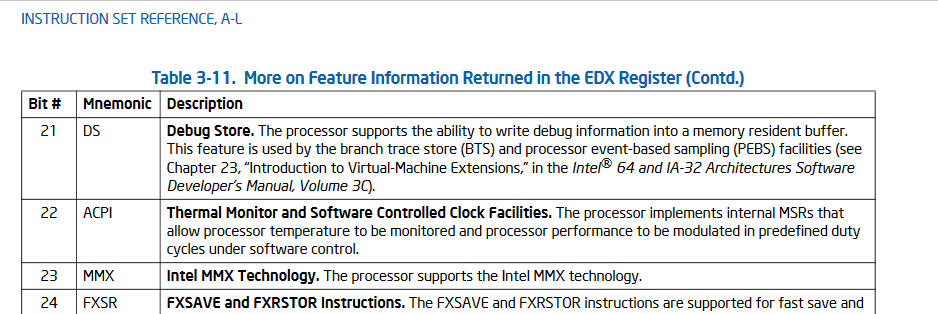
* создавать дочерний поток, в котором будет вызываться функция, реализующая требования варианта
* обладать графическим интерфейсом
* функция, реализующая требования выбранного варианта быть в составе динамической библиотеки
* Определение поддержки процессором технологии MMX реализована при помощи ассемблерной вставки

1. Анализ задачи

2.1. Определение максимальной ширины экрана в пикселях

Чтобы определить максимально возможную ширину экрана в пикселях нужно вызвать функцию **GetSystemMetrics** с параметром **SM\_CXFULLSCREEN**

2.2. Определение наличия поддержки MMX технологии



Исходя из документации по утилите **CPUID** для процессоров компании Intel нужно

* Приравнять регистр **EAX** единице
* вызвать утилиту **CPUID**
* **MMX** технология поддерживается 23 бит регистра **EDX** равен 1.

1. Общий алгоритм программы

Создание и отображении окна приложения. Запуск цикла обработки сообщений.

Создание дочернего потока, запуск процедуры в рамках созданного потока.

Загрузка динамической библиотеки. Вызов функции реализующей требования варианта:

- Определение максимальной ширины экрана в пикселях

- Определение наличия поддержки MMX технологии

Вывод полученных результатов при помощи графического интерфейса.

1. Описание разработанных функций

4.1. Функция динамической библиотеки

|  |  |
| --- | --- |
| Функция | Описание |
| void Information(int& maxWidth, bool& isMMXSupported) | Определение максимальной ширины экрана в пикселях и наличие поддержки MMX технологии процессором. |

4.2. Функции основного приложения

|  |  |
| --- | --- |
| Функция | Описание |
| DWORD WINAPI ThreadFunc(void \*) | Функция, выполняющаяся в отдельном потоке. Она включает в себя следующие этапы:  1. Подключение динамической библиотеки  2. Загрузка динамической библиотеки  3. Выполняем функцию из неё  4. Высвобождаем библиотеку  5. Интерпретируем полученные данные |
| LRESULT CALLBACK WindowFunc(HWND hWnd, UINT msg, WPARAM wParam, LPARAM lParam) | Процедура обработки сообщений. Шаги:  1. Создание шрифта  2. Создание потока в рамках которого исполняется функция **ThreadFunc**.  3. Инициализация контекста устройства, т.е. вывод строк **ScreenInfo** и **MMXInfo** |
| int WINAPI WinMain(HINSTANCE hThisInst, HINSTANCE hPrevInst, LPSTR str, int nWinMode) | Главная функция программы. Она включает в себя следующие этапы:  1. Регистрация класса окна  2. Создание главного окна  3. Цикл обработки сообщений:  - Получение сообщения  - Преобразование виртуальных кодов в ASCII-значения  - Посылка сообщения в нужную процедуру |

1. Спецификация

Программа находится в папке “ИР.ПМ-63.Кожекин”.

Чтобы собрать программу нужно в командной строке разработчика для VS 2017 нужно сначала перейти в папку с ИР, а затем ввести скомпилировать командой “compile.bat”.

Для запуска программы нужно запустить файл “driver.exe”.

1. Тесты

Процессор: Intel® Xeon® E5-2640 v4



Процессор: Intel® Core™ i7-3520m



1. Текст программы

Файл info.cpp

|  |
| --- |
| #define \_CRT\_SECURE\_NO\_WARNINGS  #define UNICODE  #include <stdio.h>  #include <Windows.h>  // Функция динамической библиотеки  extern "C" \_declspec(dllexport) void Information(int& maxWidth, bool& isMMXSupported)  {  // Определяем максимальную ширину окна (в пикселях)  maxWidth = GetSystemMetrics(SM\_CXFULLSCREEN);  // Определяем наличие поддержки MMX технологии  int fSupported;  \_\_asm  {  mov eax, 1 // CPUID уровня 1  cpuid  and edx, 0x800000 // проверяем 23 бит регистра EDX  mov fSupported, edx // если MMX поддерживается, то EDX[28] == 1  }  if (fSupported != 0)  isMMXSupported = true;  } |

Файл driver.cpp

|  |
| --- |
| #define \_CRT\_SECURE\_NO\_WARNINGS  #define UNICODE  #include <windows.h>  #include <string>  #include <stdio.h>  #include <Windows.h>  LPCWSTR szClassName = L"MyClass";  LPCWSTR szTitle = L"Individual work - resource managment";  TCHAR ScreenInfo[256];  TCHAR MMXInfo[256];  // Функция выполняющаяся в отдельном потоке  DWORD WINAPI ThreadFunc(void \*)  {  typedef int(\*ImportFunction)(int&, bool&);  ImportFunction DLLInfo; // Подключаем динамическую библиотеку  HINSTANCE hinstLib = LoadLibrary(TEXT("info.dll"));  DLLInfo = (ImportFunction)GetProcAddress(hinstLib, "Information");  int maxWidth = 0;  bool isMMXSupported = false;  DLLInfo(maxWidth, isMMXSupported); // Получаем данные  FreeLibrary(hinstLib); // Высвобождаем библиотеку  // Интерпретация выходных данных функции DLLInfo  wsprintf(ScreenInfo, L"Max screen width is: %d px", maxWidth);  if (isMMXSupported)  wsprintf(MMXInfo, L"MMX techology is supported");  else  wsprintf(MMXInfo, L"MMX techology isn't supported");  return 0;  }  // Процедура обработки сообщений  LRESULT CALLBACK WindowFunc(HWND hWnd, UINT msg, WPARAM wParam, LPARAM lParam)  {  PAINTSTRUCT ps;  HDC hDC;  // Создание шрифтра  HFONT hFont = CreateFont(20, 0, 0, 0, FW\_THIN, FALSE, FALSE, FALSE, DEFAULT\_CHARSET, OUT\_OUTLINE\_PRECIS,  CLIP\_DEFAULT\_PRECIS, CLEARTYPE\_NATURAL\_QUALITY, VARIABLE\_PITCH, TEXT("Calibri"));  switch (msg)  {  // Создание потока  case WM\_CREATE:  HANDLE hThread;  DWORD IDThread;  hThread = CreateThread(NULL, 0, ThreadFunc, NULL, 0, &IDThread);  WaitForSingleObject(hThread, INFINITE);  CloseHandle(hThread);  break;  // Закрытие окна  case WM\_DESTROY:  PostQuitMessage(0);  break;  // Инициализация контекста устройства  case WM\_PAINT:  hDC = BeginPaint(hWnd, &ps);  SelectObject(hDC, hFont); // Меняем шрифт  TextOut(hDC, 15, 10, ScreenInfo, 256);  TextOut(hDC, 15, 30, MMXInfo, 256);  EndPaint(hWnd, &ps);  break;  default:  return DefWindowProc(hWnd, msg, wParam, lParam);  }  return 0;  }  // Главная функция Windows-приложения  int WINAPI WinMain(HINSTANCE hThisInst,  HINSTANCE hPrevInst, LPSTR str, int nWinMode)  {  MSG msg;  WNDCLASS wcl;  HWND hWnd;  // Создание класса окна  wcl.hInstance = hThisInst;  wcl.lpszClassName = szClassName;  wcl.lpfnWndProc = WindowFunc;  wcl.style = CS\_HREDRAW | CS\_VREDRAW;  wcl.hIcon = LoadIcon(NULL, IDI\_APPLICATION);  wcl.hCursor = LoadCursor(NULL, IDC\_ARROW);  wcl.lpszMenuName = NULL;  wcl.cbClsExtra = 0;  wcl.cbWndExtra = 0;  wcl.hbrBackground = (HBRUSH)GetStockObject(WHITE\_BRUSH);  // Регистрация класса окна  RegisterClass(&wcl);  // Создание окна на базе его класса  hWnd = CreateWindow(szClassName, szTitle,  WS\_OVERLAPPEDWINDOW | WS\_CLIPCHILDREN |  WS\_CLIPSIBLINGS,  100, 50, 700, 120, HWND\_DESKTOP,  NULL, hThisInst, NULL);  // Отображение окна  ShowWindow(hWnd, nWinMode);  UpdateWindow(hWnd);  // Цикл обработки сообщений  while (GetMessage(&msg, NULL, 0, 0))  {  TranslateMessage(&msg);  DispatchMessage(&msg);  }  return msg.wParam;  } |

Файл compile.bat

|  |
| --- |
| if exist \*.exe del \*.exe  if exist \*.dll del \*.dll  cl /EHsc driver.cpp kernel32.lib user32.lib gdi32.lib advapi32.lib  cl /LD info.cpp kernel32.lib user32.lib gdi32.lib advapi32.lib  del \*.obj \*.lib \*.exp |