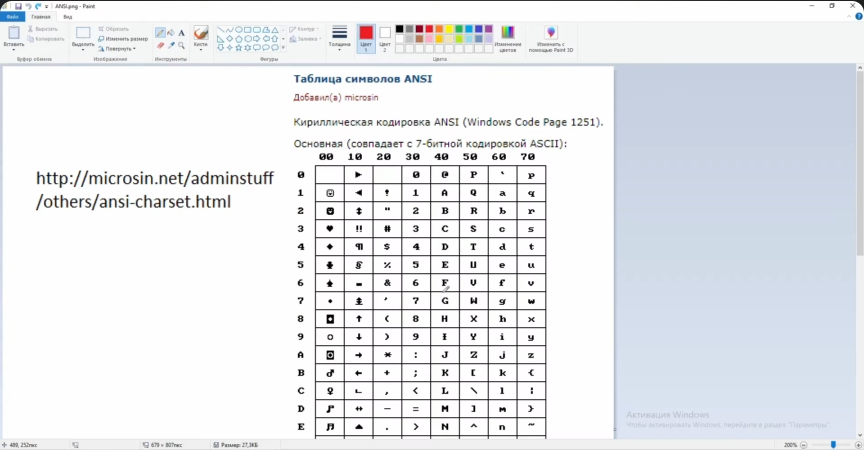
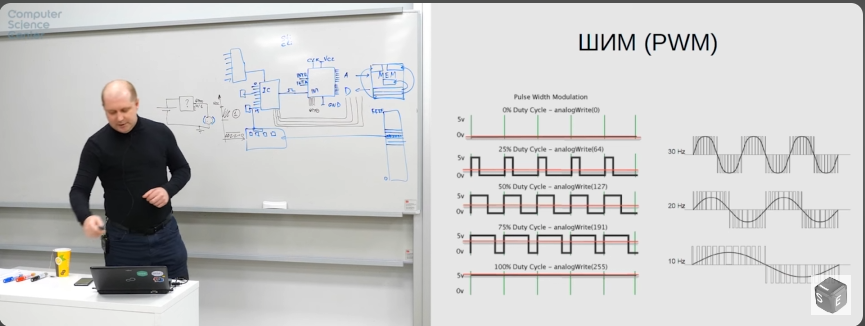
|  |  |  |
| --- | --- | --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РЕСПУБЛИКИ КАЗАХСТАН  СЕВЕРО-КАЗАХСТАНСКИЙ УНИВЕРСИТЕТ ИМ. М. КОЗЫБАЕВА  ФАКУЛЬТЕТ ИНЖЕНЕРИИ И ЦИФРОВЫХ ТЕХНОЛОГИЙ  КАФЕДРА «ИНФОРМАЦИОННО-КОММУНИКАЦИОННЫЕ ТЕХНОЛОГИИ» | | |
|  | | |
| Отчет по CРО № 12 | | |
| Выполнил студент  группы ВТиПО-22: |  | Белокопытов Я.Ю |
| Проверил профессор: |  | Куликов В.П |

Петропавловск, 2024

06. Низкоуровневое программирование. Ассемблер. Пример программы. [Универсальный программист]

Введение в программирование на языке ассемблера

• Видео начинается с объяснения, что программирование на языке ассемблера называется низкоуровневым программированием, так как оно работает на уровне, максимально приближенном к процессору.

• Язык ассемблера используется для разработки операционных систем, драйверов и системного софта, а также для программирования микроконтроллеров.

Пример программы на языке ассемблера

• В видео демонстрируется пример программы на языке ассемблера, написанной на языке программирования C++.

• Программа создается в текстовом редакторе и преобразуется в машинный код с помощью ассемблера.

Работа с ассемблером

• В видео объясняется, как использовать ассемблер для преобразования программы из исходного кода в машинный код.

• Ассемблер также используется для работы с данными, которые могут быть встроены в исполняемый файл.

Работа с строками и данными

• В видео обсуждаются особенности работы со строками и данными в ассемблере.

• Строки и данные могут быть представлены в виде кодов символов или кодов чисел.

• В ассемблере для Intel-процессоров байты идут в обратном порядке, сначала младший байт, потом старший.

Работа с именами и адресами в асемблере

• В асемблере используются имена для адресации данных и кода.

• Имена могут быть даны как адресам данных, так и адресам кода.

• Имена в асемблере называются символами, а символы в строковых данных называются характерами.

Создание исполняемого файла

• Асемблер преобразует имена в адреса и создает исполняемый файл.

• Исполняемый файл содержит машинный код и данные, но не содержит имен.

Восстановление исходного кода

• Исходный код можно восстановить с помощью дизасемблера, но имена будут потеряны.

• Восстановление исходного кода затруднено из-за отсутствия имен.

Основы устройства компьютера. Ввод-вывод.

Ввод-вывод через память и порты

• Ввод-вывод в компьютерах может быть организован через память или порты.

• Ввод-вывод через порты использует специальные команды и прерывания для обработки информации.

Прерывания и обработчики прерываний

• Прерывания позволяют процессору автоматически проверять наличие информации на портах и обрабатывать ее.

• Обработчики прерываний выполняют действия с информацией на портах.

Прямой доступ к памяти

• Прямой доступ к памяти позволяет устройствам напрямую обращаться к памяти без участия процессора.

• Освобождает процессор от рутинной работы и двойной пересылки данных.

Архитектура ЭВМ Лекция 11: Организация ввода вывода. Интерфейсы передачи данных.

Введение в прерывания

• Ввод-вывод - это взаимодействие компьютера с внешним миром, включая устройства ввода (клавиатура, монитор, принтер) и вывода (жесткий диск).

• Прерывания - это механизм, позволяющий процессору обрабатывать сигналы от устройств.

Физическая реализация прерываний

• Прерывание - это интерфейс физического прерывания, который позволяет процессору обрабатывать сигналы от устройств.

• Контроллер прерываний - это устройство, которое обрабатывает прерывания и передает информацию процессору.

Обработка прерываний

• Контроллер прерываний может писать на шину данных, чтобы процессор мог прочитать состояние устройства.

• В процессоре есть инструкции для обработки прерываний, такие как "сти" и "клои".

Завершение работы с прерываниями

• Устройства могут ждать, пока процессор закончит обработку предыдущего обработчика прерывания, используя "инт-р" и "инт-а".

• Контроллер прерываний используется для управления шинами и другими устройствами.

Шина и прерывания

• В видео обсуждается шина и ее аппаратный интерфейс, который включает контроллер шины и набор проводов.

• Шина используется для координации устройств и предоставления возможности захвата шины для взаимодействия с процессором.

Клавиатура и прерывания

• Приводится пример клавиатуры, где прерывание возникает, когда пользователь нажимает клавишу.

• Контроллер прерывания передает это прерывание процессору, который считывает состояние клавиатуры из таблицы векторов прерываний.

Мемори маппинг и драйверы

• Мемори маппинг - это процесс, когда физическое адресное пространство связывается с устройствами ввода-вывода.

• Обработчик прерывания получает управление и знает конфигурацию устройства, чтобы читать или писать в память.

• Драйвер - это программа, которая обслуживает устройство и позволяет обработчику прерывания работать с ним.

Управление внешними устройствами

• В процессоре есть специальные адреса для управления памятью, ввода-вывода и другими процессами.

• Для управления внешними устройствами, такими как клавиатура и мышь, используются специальные контакты, называемые "джойп айо".

Использование процессора для управления внешними устройствами

• Процессор может управлять своим сопротивлением, что позволяет ему быть проводником или непроводником.

• Это позволяет использовать процессор для управления внешними устройствами, такими как лампочки и моторы.

Протокол передачи данных между устройствами

• Для передачи данных между двумя устройствами используется протокол "у-арт", который позволяет двум устройствам обмениваться данными.

• Для этого используются специальные контакты, называемые "трансмиттер" и "ресивер".

Скорость передачи данных

• Скорость передачи данных измеряется в битах в секунду (бит/с).

• Для понимания, где начинается и заканчивается байт, используются дополнительные инструменты, такие как контроль четности и топовый бит.

Дискретные сигналы и аналоговые сигналы

• Дискретные сигналы, такие как логические единицы и нули, могут быть мгновенными или иметь некоторую инерцию.

• Аналоговые сигналы, такие как лампочки, имеют определенную инерцию и могут быть смоделированы с помощью генерации горбатых сигналов.

• Для сложных случаев, таких как синусоида, размер загрузки интервала зависит от производной сигнала.

Интерфейс и протокол

• В видео обсуждается интерфейс и протокол, используемый в устройствах для обмена данными.

• В отличие от предыдущего протокола, где требовалось договариваться о чистоте между устройствами, в этом протоколе чистота задается мастером, а устройства только передают данные.

Работа интерфейса

• В интерфейсе есть мастер и слейф, которые обмениваются данными через два провода.

• Мастер генерирует чистоту, а слейф передает данные мастеру.

• Мастер может выбирать устройство, с которым он работает, и устанавливать значение бита данных.

• Если устройство медленное, оно может придержать клок до тех пор, пока не обработает бит.

Выбор устройств и протокол

• Устройства выбираются по номеру, который указывает на их положение в протоколе.

• Протокол общения с устройством запакован в более высокий уровень.

• Мастер может поднять клок только после того, как слейф опустит его.

Аналого-цифровой преобразователь

• Объясняется принцип работы аналого-цифрового преобразователя, который позволяет оцифровывать аналоговые сигналы и работать с ними в цифровом виде.

• Упоминается идея использования операционного усилителя для сравнения двух сигналов и определения, какой из них больше.

Множественные шины и протоколы обмена

• Обсуждается идея разделения устройств на быстрые и медленные шины, а также использование протоколов обмена для взаимодействия между ними.

• Упоминаются популярные шины, такие как USB, SATA и PCI Express.

Кэш-память и протокол MESI

• Объясняется проблема устаревания данных в кэш-памяти из-за одновременного использования разными процессорами.

• Упоминается протокол MESI, который позволяет контролировать состояние кэш-памяти и предотвращать конфликты между процессорами.

Различия между процессорами и микроконтроллерами

• В видео обсуждается разница между процессорами и микроконтроллерами.

• Процессоры, такие как Intel Core i7, имеют свою собственную память, оперативную память и другие компоненты, в то время как микроконтроллеры, такие как Arduino, являются законченными железками, которые можно подключить к плате и они будут работать.

• Микроконтроллеры могут иметь различные ядра, такие как ядра реального времени, которые используются для обработки данных в реальном времени.

• В целом, процессоры и микроконтроллеры имеют разные подходы к дизайну и использованию, и важно понимать их различия, чтобы правильно использовать их в своих проектах.

Использованная литература:

1. [https://online.osll.ru/useful](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbU5tcldsQThZb1RjMnJJVlRwMWJFN1NRNTJsd3xBQ3Jtc0tsaFVUQ3BTR3p2TDg5aGxYYnpoMWZRbFVSTklGNEF0V3ZNZkFLY2VQbngxLW9IWW5GRE9CbWFlc2dDcG85VEpLcl9MNkgweHpVeWFhZTBIajljeHFNR1hENXJjRm5EZ1FMaXJCZGNKWmlQNkZZQjlCOA&q=https%3A%2F%2Fonline.osll.ru%2Fuseful&v=WLj1R2YlueY)   
2. abglazov.rfpgu.ru

3. Плейлист Универсальный программист:

• Универсальный программист