### Введение

За основу представленной учебной модели компьютера взята модель принстонской (фон Неймана) архитектуры, описанная в книге Дж., Г. Брукшира Введение в компьютерные науки. Общий обзор, 6-е издание. : Пер. с англ. — М. : Издательский дом "Вильямс", 2001. — 688 с. : ил. — Парал.тит. англ.

Модель составляют процессор, память и устройства ввода-вывода.

Процессор включает АЛУ, представленное регистрами общего назначения (РОН, 16 регистров с номерами от 0 до 0хF в шестнадцатеричной системе счисления по 8 бит), и УУ, представленное регистрами: команд (длиной 16 бит) и счетчика команд (8 бит).

Основная память представлена последовательностью 256 ячеек с адресами от 0 до 0хFF, каждая из которых имеет длину 8 бит. Старшие адреса от 0хC0 до 0хFF формируют буферную видеопамять для прямого вывода результатов программы.

Консоль ввода-вывода представлена полем для ввода кода программы и цветным дисплеем для отображения результатов вычислений (4х8 знакомест, 256 цветов).

«Аппаратное» управление вычислениями осуществляется кнопками:

* запись введенного на клавиатуре кода в память,
* в учебных целях осуществляется только пошаговое (по командное) выполнение программы,
* очистка памяти (в том числе, регистров и дисплея).

Целые числа обрабатываются в двоичном дополнительном коде (two’s complement).

Примечание. Дополнительный код для отрицательного числа можно получить инвертированием его двоичного модуля (первое дополнение) и прибавлением к инверсии единицы (второе дополнение).

Поскольку длина РОН 8 бит, доступный числовой диапазон составляет от -128 (0х80) до 127 (0x7F).

Длина каждой машинной команды равна двум байтам. Первые четыре бита содержат код операции, последние 12 бит образуют поле операндов. В приведенной ниже таблице перечислены и кратко описаны команды, показанные в шестнадцатеричном представлении. Буквы R, S и Т используются для указания в поле операндов позиции шестнадцатеричных цифр, являющихся идентификаторами регистров, которые меняются в зависимости от конкретной команды. Буквы X и Y используются для указания в поле операндов позиций тех шестнадцатеричных цифр, которые не являются идентификаторами регистров.

Таблица 1. Машинные команды модели

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Опкод** | **Команда** | **Операнд** | **Описание** | **Пример** |
| 1 | LOAD1 | RXY | Загрузка в регистр R двоичного кода числа из ячейки памяти с адресом XY. | Команда 14A3 помещает в регистр 4 содержимое ячейки памяти с адресом A3 |
| 2 | LOAD2 | RXY | Загрузка в регистр R двоичного кода числа XY. | Команда 20АЗ помещает в регистр 0 значение A3 |
| 3 | STORE | RXY | Сохранение двоичного кода числа, хранящегося в регистре R, в ячейке памяти с адресом XY. | Команда 35В1 помещает содержимое регистра 5 в ячейку памяти с адресом В1 |
| 4 | MOV | 0RS | Перемещение двоичного кода числа из регистра R в регистр S. | Команда 4 0А4 копирует содержимое регистра А в регистр 4 |
| 5 | ADD1 | RST | Суммирование дополнительных двоичных кодов чисел, хранящихся в регистрах S и Т, с сохранением суммы в регистре R. | Команда 5726 суммирует дополнительные двоичные коды чисел, хранящиеся в регистрах 2 и 6, а сумму помещает в регистр 7 |
| 6 | DIV | RST | Целочисленное деление двоичных кодов чисел, хранящихся в регистрах S (делимое) и Т (делитель), с сохранением частного в регистре R. | Команда 6726 делит двоичный код числа из регистра 2 на число, хранящееся в регистре 6, а частное помещает в регистр 7 |
| 7 | OR | RST | Выполнение поразрядной операции OR над двоичными кодами чисел, хранящихся в регистрах S и Т, и помещение результата в регистр R. | Команда 7СВ4 помещает в регистр 7 результат операции OR над содержимым регистров В и 4 |
| 8 | AND | RST | Выполнение поразрядной операции AND над двоичными кодами чисел, хранящихся в регистрах S и Т, и помещение результата в регистр R. | Команда 8045 помещает в регистр 0 результат операции AND над содержимым регистров 4 и 5 |
| 9 | XOR | RST | Выполняется поразрядная операция XOR над двоичными кодами чисел, хранящихся в регистрах S и Т, и результат помещается в регистр R | Команда 95F3 помещает в регистр 5 результат операции XOR над содержимым регистров F и 4 |
| A | ROTATE | R0X | Выполняется операция циклического поразрядного сдвига вправо на X позиций над двоичным кодом числа, хранящегося в регистре R. При каждом одиночном сдвиге бит из младшего разряда перемещается в старший разряд. | Команда А403 выполняет циклический поразрядный сдвиг вправо на 3 бита в содержимом регистра 4 |
| В | JUMP | RXY | Выполняется переход к команде, размещенной в ячейке памяти по адресу XY, если двоичный код числа в регистре R совпадает с двоичным кодом числа в регистре 0. | Команда В43С сначала сравнивает содержимое регистра 4 с содержимым регистра 0. Если они равны, последовательность выполнения команд изменится так, что следующей будет выполнена команда, расположенная в памяти по адресу ЗС. В противном случае выполнение программы продолжится в обычной последовательности |
| С | HALT | 000 | Прекращение выполнения программы, отображение результатов на монитор. | Команда С000 останавливает выполнение программы и копирует содержимое ячеек видеопамяти в соответствующие знакоместа на дисплее консоли |

Код операции для каждой команды размещается в первых ее четырех битах и представляется одной шестнадцатеричной цифрой. Весь перечень команд включает только двенадцать базовых команд, коды операций которых представляются шестнадцатеричными цифрами от 1 до С. Таким образом, любой код команды, который начинается с шестнадцатеричной цифры 3 (битовая комбинация 0011), относится к команде сохранения STORE, а каждая команда, код которой начинается с шестнадцатеричного символа А, является командой циклического сдвига ROTATE.

Поле операндов состоит из трех шестнадцатеричных цифр (12 бит) и во всех случаях (кроме команды остановки HALT, для которой не требуется никаких уточнений) содержит дополнительные сведения о команде, заданной кодом операции. Например, если первая шестнадцатеричная цифра команды равна 1 (код операции считывания ячейки памяти LOAD), то следующая шестнадцатеричная цифра команды указывает общий регистр, в который требуется загрузить считанное из основной памяти значение, а последние две шестнадцатеричные цифры задают адрес ячейки памяти, из которой требуется считать данные. Например, команда 1347 (шестнадцатеричное число) воспринимается машиной как "Загрузить в регистр 3 содержимое ячейки памяти с адресом 47". Если код операции представлен шестнадцатеричной цифрой 7 (операция OR над содержимым двух регистров общего назначения), то следующая шестнадцатеричная цифра указывает номер регистра, в который следует поместить результат операции, а две последние шестнадцатеричные цифры поля операндов задают номера тех регистров, над содержимым которых необходимо выполнить операцию OR. В результате команда 70С5 понимается как инструкция "Выполнить операцию OR с содержимым регистров С и 5, а результат поместить в регистр 0".

Небольшое отличие существует и между двумя командами загрузки LOAD. Возможно, вы уже заметили, что код операции 1 (шестнадцатеричный) относится к команде загрузки регистра общего назначения содержимым ячейки основной памяти, тогда как код операции 2 (шестнадцатеричный) — к команде загрузки регистра общего назначения указанным числовым значением. Различие заключается в том, что поле операндов в команде первого типа содержит адрес, тогда как в команде второго типа поле операндов содержит ту битовую комбинацию, которую требуется загрузить в регистр.

В команде перехода JUMP (код операции — шестнадцатеричная цифра В) первая шестнадцатеричная цифра поля операндов указывает, какой регистр общего назначения следует сравнить с регистром 0. Если указанный регистр содержит ту же битовую комбинацию, что и регистр 0, то машина выполняет переход посредством выбора следующей команды по тому адресу,который указан в двух последних шестнадцатеричных цифрах поля операндов. В противном случае программа продолжает нормальную последовательность выполнения команд. По сути, это пример команды условного перехода. Однако если первая шестнадцатеричная цифра поля операндов будет равна 0, то данная команда запрашивает сравнить регистр 0 с регистром 0. Так как содержимое любого регистра всегда равно самому себе, то в этом случае переход всегда выполняется. Следовательно, команда, код которой начинается с 0хВ0, воспринимается как команда безусловного перехода.

### Описание констант и переменных модели

Константы

cw=2 длина машинного слова в байтах

ladr=2 длина адреса основной памяти в байтах

cron=16 количество РОН

cram=256 количество адресов основной памяти (включая видеопамять 0хС0-0хFF)

Переменные

cc - значение (содержимое регистра) счетчика команд (адрес текущей команды в основной памяти)

rc - содержимое регистра текущей команды

rona(cron) - массив содержимого регистров общего назначения

rama(cram) - массив ячеек основной памяти

opcod - код текущей операции (анализ rc)

op(3) - массив значений операндов текущей операции (анализ rc)

Процедура машинного цикла Macycle

1. чтение текущей команды в соответствующий регистр (из переменной rc в),
2. инкремент значения переменной ps (увеличение значения адреса на 1 и отображение изменений в позиции формы,
3. декодирование текущей команды (анализ значения переменной rc), определение значений переменных opcod и op(3)
4. активизация выполнения команды (изменение соответствующих текущей команде значений позиций регистров и основной памяти на форме)

Процедуры выполнения команд (соответственно табл.1)

Таблица 2. Процедуры имитации машинных команд

|  |  |
| --- | --- |
| **Имя процедуры** | **Параметры** |
| Mload1(rona(i),rama(j)) | номер РОН назначения, адрес источника в основной памяти |
| Mload2(rona(i),number) | номер РОН назначения, шестнадцатеричное число |
| Mstore(rona(i),rama(j)) | номер РОН источника, адрес назначения в основной памяти |
| Mmov(rona(i),rona(j)) | номер РОН источника, номер РОН назначения |
| Madd1(rona(i),rona(j),rona(k)) | номер РОН суммы, номера РОН слагаемых |
| Mdiv(rona(i),rona(j),rona(k)) | номер РОН частного, номера РОН делимого и делителя |
| Mor(rona(i),rona(j),rona(k)) | номер РОН результата, номера РОН участников |
| Mand(rona(i),rona(j),rona(k)) | номер РОН результата, номера РОН участников |
| Mxor(rona(i),rona(j),rona(k)) | номер РОН результата, номера РОН участников |
| Mrotate(rona(i),position) | номер РОН, шестнадцатеричное число |
| Mjump(rona(i),rona(j),rama(k)) | номера РОН сравнения, адрес в основной памяти копируемый в cc |
| Mhalt() |  |

Процедуры изменения элементов формы

* запуск записи программы пользователя в позиции ячеек основной памяти с кнопки формы (в каждую ячейку записывается по 2 последовательных символа программы)
* запуск машинного цикла с кнопки формы
* запись результатов выполнения текущей команды в позиции регистров и ячеек основной памяти
* формирование позиций таблицы ячеек основной памяти
* изменение позиций регистров cc и rc на форме
* вывод результатов программы на консоль (отображение ячеек видеопамяти бай-в-байт на 32 знакоместа консоли 4х8)
* обработка ошибок и сообщения пользователю

### Интерфейс приложения и порядок работы

Приложение реализовано в формате html-документа с автоматизацией на JavaScript. Следовательно его рабочей средой является любой Веб-браузер.

Интерфейс пользователя представлен на рис.1 и включает информационные поля (зоны), иллюстрирующие обработку данных моделью и отображающие содержание оперативной памяти (RAM), регистров процессора (RON, RC, CC) и устройств ввода-вывода (KEYBOARD, DISPLAY).

Приложение предоставляет пользователю возможность ввести последовательность машинных команд, реализующую алгоритм решения некоторой задачи обработки данных (программный код) и выполнить их в пошаговом режиме, визуализируя на соответствующих информационных полях происходящие при этом изменения состояния компонентов модели (запись данных в ячейки, регистры или знакоместа...).

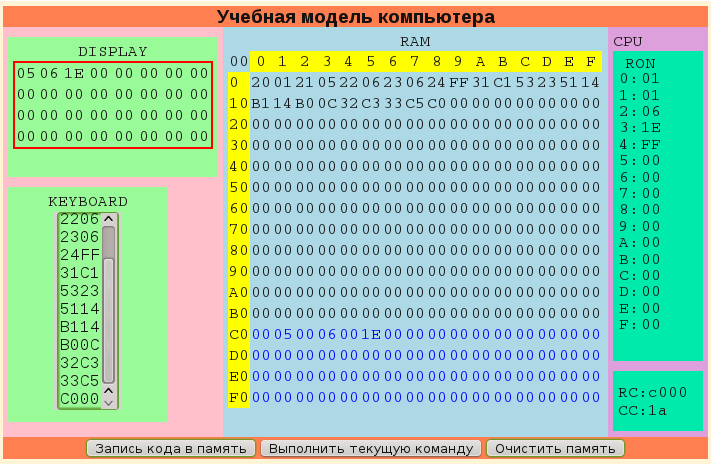
Примечание. На рис.1 представлено состояние компонентов модели на момент завершения вывода результатов выполнения машинного кода пользователя.

Поле KEYBOARD предоставляет необходимый минимум операций редактирования текста (копирование/вставка фрагментов, добавление/замена символов и др.) кода. Машинные команды последовательно одна за другой вводятся пользователем с клавиатуры в область KEYBOARD как подстроки из 4х шестнадцатеричных чисел в соответствии с табл. 1. Весь набор команд кода составляет одну строку.

Примечание. Нажатие клавиши «Enter» при вводе команд вставляет служебный символ (разрыва строки), обрабатываемый при дальнейшей работе модели аналогично пробелу. То есть под него будет выделена ячейка памяти, что приведет, возможно, к последующей ошибочной интерпретации команд.

Для выполнения введенного пользователем кода его необходимо предварительно записать в память модели посредством соответствующей кнопки. Последовательность команд записывается в память начиная с младшего адреса (0х00) бай за байтом и отображается в недоступной для редактирования информационной зоне RAM, которая скомпонована как таблица из 16 строк по 16 байтовых ячеек.

Адрес 0х00

Рис.1 Текущее состояние модели

Если визуально в области RAM обнаружены ошибки или требуется записать в нее другой код, следует выполнить очистку памяти модели с помощью соответствующей кнопки.

Для последовательного отображения изменений состояния компонентов модели машинными командами кода пользователя, записанного в ее память, используется кнопка «Выполнить текущую команду».

### Задачи практикума

1. Вычитание чисел (в РОН). 7-4=3 (2007210422ff9112230151135001c000).
2. Перезапись содержимого ячейки памяти (RAM) в следующую (по адресу).

Копия 0xE в 0xF (100e210e22015112310b3000c00007).

Здесь используется прием перезаписи команды в процессе вычислений. Изначально по адресу 0xB записываем 00, а зачем переписываем вычисленным значением из РОН1 (инкремент адреса исходной ячейки), которое в соответствующей команде будет восприниматься как необходимый адрес!

Вывод значения (перезапись в буферную память).

1. Копирование массива ячеек из N элементов в область буферной памяти (для вывода на дисплей). N=5 (20052101222023c12400b41e3211331315 35 5221533153315441b00ac00005320824ff)
2. Умножение двух целых чисел с выводом на дисплей исходных значений и результата.

6\*5=6+6+6+6+6=30 (2003210230c131c3220123014014b31653325441b00e34c5c000)

1. Сумма последовательности чисел от 0 до N.

5+4+3+2+1=15 (2000210531c1401422ff51125441b112b00a34c3c000)

1. Сумма элементов массива с выводом исходной последовательности чисел и результата

L=5, массив располагается сразу за кодом

(20052101222a23c12400b4203211331d15 5665522153315331544135 b00a33275331533136 c0001011121314)

1. Вычисление факториала числа < 5 с выводом на дисплей исходных значений и результата.

5!=5\*4\*3\*2=((5+5+5+5)+20+20))+60=120

(2000210231c122ff23fe541340454016b426b12656615552b51cb0145442b42640454061b01436c3c000)

не работает для 1.

1. Выбор max/min значений последовательности целых чисел с выводом на дисплей исходных значений и результата.

Наибольшее из двух чисел (РОН 1 и 2) помещается в РОН1 (2000212f220723ff2480952355518554b5144021c000)

Наибольший из элементов массива длиной L=5, который располагается сразу за кодом

(20002105220123ff243a25c1401627802800b6303419352d19 9a935aa88aa7ba244098544255525552566339 b01235375552555238 c0001013121114)

1. Выполнение на дисплее цветных символьных орнаментов (прямоугольник, треугольник, шахматка и др.).
2. Сортировка последовательности целых чисел с выводом на дисплей исходных значений и результата (в двух разных строках).