Каллисто

Алгоритмический язык программирования для «Электроники МК-161»

Руководство по эксплуатации

(бета-версия 0.8б)

Москва, 2016

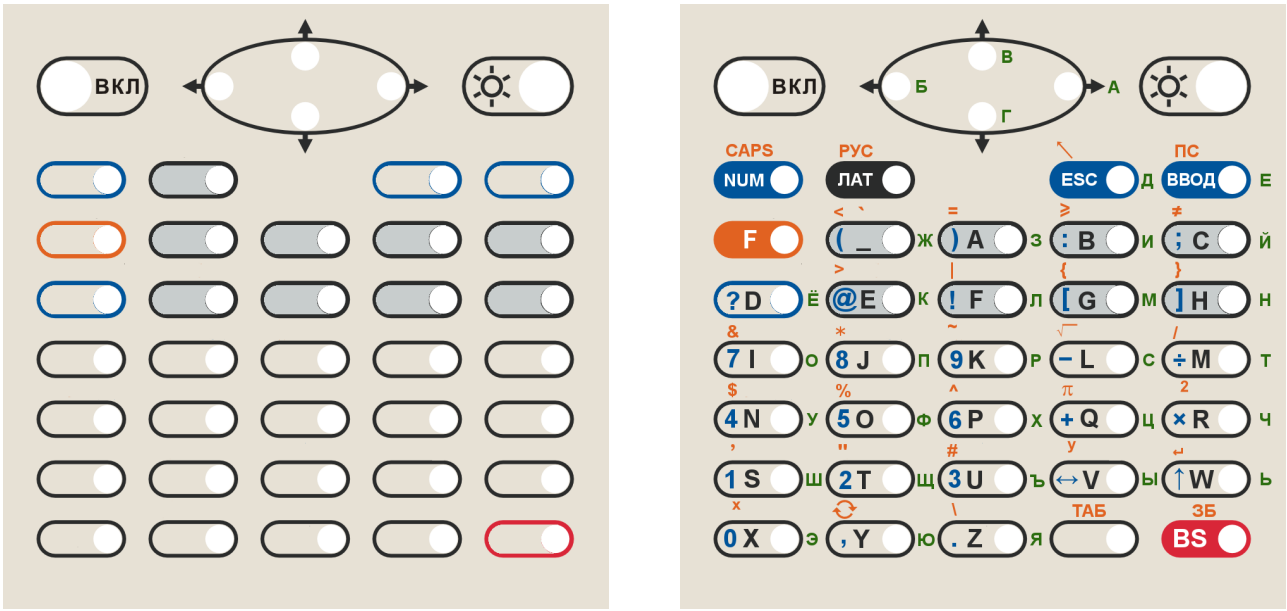

Рис.1 *Оверлей* — накладная клавиатура для Каллисто на «Электронике МК-161» в натуральную величину.

Каллисто

Каллисто — алгоритмический язык программирования для «Электроники МК-161». Каллисто создана как расширение *компактного входного языка калькулятора* (ЯМК) для удобства программирования *на борту* «Электроники МК-161», то есть без необходимости приобретения зарубежных компьютеров и использования внешних сред разработки.

Каллисто разработана в Москве на основе Форта и его наследника colorForth. Каллисто распространяется по свободной лицензии, которую можно найти в начале файла **`Callisto.mkl`**

Данное Руководство ещё в процессе написания и весьма «сырое», но я решил предоставить его в помощь тем смелым первопоселенцам, кто уже начал осваивать Каллисто.

Подробнее о Каллисто рассказано на вики по адресу pmk.the-hacker.ru

Краткая история

Каллисто названа в честь четвёртой луны системы Юпитера — самого дальнего из четырёх крупнейших спутников самой большой планеты нашей Солнечной системы. Иногда её даже называют Юпитер IV.

С Земли Юпитер (♃) хорошо виден невооружённым глазом, как яркая оранжевая звезда. В телескоп гигантский Юпитер виден как диск, по которому можно изучать характерные детали его атмосферы. Но Юпитер представляет собой газовый гигант, на котором даже нет поверхности, чтобы высадиться. Каллисто легко увидеть в хороший бинокль и на ней есть не только твёрдая поверхность, но и какая-никакая гравитация. Человек, когда туда доберётся, сможет ходить по ней. Каллисто побольше нашей Луны и незначительно меньше Меркурия. Каллисто — самый перспективный спутник Юпитера для колонизации, т.к. он единственный из четырёх крупных спутников находится за пределами смертельного радиационного пояса Юпитера.

Первые семь месяцев, с августа 2014 по март 2015 года, Каллисто разрабатывалась на «Электронике МК-161», как педантичная реализация готовящегося стандарта Форта-2012. После того, как Форт «задышал» на «Электронике» и изучения критики стандарта Форта-94 стала понятна утопичность подобного подхода.

Чрезмерное увлечение стандартами Форта возникло в результате избытка мощности процессоров и других доступных ресурсов. Форт нетребователен к ресурсам, появившиеся излишки стали вкладывать в обеспечение архитектурной независимости и поддержания *переносимых программ*. Написание переносимых программ отнимает больше времени, а исполнение потребляет больших вычислительных ресурсов, чем когда слова Форта и оборудование традиционно подстраивают друг под друга.

На «Электронике», особенно при реализации Форта на уровне входного языка микрокалькулятора (ЯМК), лишних ресурсов нет. Поэтому было решено поступить традиционным для Форта путём. Сейчас архитектура Каллисто отражает архитектуру «Электроники МК-161». Родной, разработанный специально для «Электроники» язык, позволяет удобней управлять своей машиной с максимальной скоростью, доступной Форту.

Каллисто — новый удобный входной язык, серьёзно расширяющий возможности ЯМК — входного языка советских микрокалькуляторов и новосибирских ЭКВМ. Разрабатывая Каллисто, при возможности и равноценности выбора я отдавал преимущество решениям, стандартным для Форта. Когда для обеспечения совместимости с зарубежным стандартом приходилось жертвовать быстродействием или памятью, я выбирал жертвовать совместимостью ради сохранения наших скромных ресурсов. В любом случае преимущество отдавалось совместимости с входным языком советских ПМК.

Сразу после своего появления Каллисто вобрала в себя лучшее из входного языка (ЯМК), набора символов и клавиш «Электроники». «Перелёт на Каллисто» осознанно сделан максимально безболезненным и даже удобным для владельцев «Электроники МК-161» и советских ПМК.

Перенос программ с Форта на Каллисто возможен, но для этого программисту придётся учесть особенности «Электроники МК-161». Учесть скромные размеры её индикатора и памяти, наборы клавиш и символов. Особенности Каллисто, входного языка, будут требовать от программиста внимательности при переводе программ, по мере работы над ними приобретающих характерный внешний вид программ для «Электроники». Такие проекты можно сравнить с доставкой земных товаров на небесное тело Каллисто. Впрочем, затраты это не такие уж космические и доступны как профессионалам, так и наиболее опытным любителям.

Добро пожаловать на Каллисто!

Установка Каллисто и комплект поставки

Установка Каллисто подобна установке операционной системы на «голую» ЭВМ. После успешной установки включение МК-161 будет автоматически запускать Каллисто. Внутри Каллисто вы будете писать программы, из неё же их запускать. Снимать накладную клавиатуру вам придётся редко, только при выходе из Каллисто для составления и запуска программ на языке МК (команда `BYE`).

К счастью, процесс установки Каллисто проще и быстрее, чем установка операционной системы. Если вы передавали на МК-161 с компьютера хотя бы одну программу, проблем с установкой Каллисто у вас не возникнет.

Главное отличие — после установки Каллисто вам захочется также передать в МК-161 редактор и другие программы, написанные на Каллисто. Их придётся передавать в МК-161 по одному блоку, записывая эти блоки в отдельный каталог. Но давайте всё по порядку.

Каллисто распространяется в архиве `ZIP`, название которого содержит номер версии.

Запускаемым файлом Каллисто является файл `Callisto.mkp` — он занимает 100 страниц памяти программ «Электроники МК-161», начиная со страницы №0. Именно его и требуется передать в «Электронику». Желательно сразу же записать Каллисто на электронный диск, можно под русским именем.

Запускается Каллисто после загрузки `Callisto.mkp` в память программ МК-161 с компьютера или электронного диска обычным образом: `В/О С/П`.

Исходный текст Каллисто располагается в файле `Callisto.mkl` в кодировке Windows. При желании вы можете самостоятельно скомпилировать Каллисто стандартными средствами в уже известный вам исполняемый файл `Callisto.mkp`. Пока документация в процессе разработки, исходный текст транслятора крайне полезен для изучения Каллисто.

В каталоге `Blocks` содержатся блоки — полезные инструменты и примеры, написанные для «Электроники МК-161» на языке Каллисто. Например, файл `Blocks/B011.mkt` содержит исходный код *строкового редактора EDIT* — стандартное средство Каллисто для написания программ. Эти блоки передаются в «Электронику МК-161» и должны быть сразу же записаны как файлы типа T (текст) с таким же именем в каталог `Каллисто1`, располагающийся в корневом каталоге электронного диска А. Например, редактор EDIT должен быть записан на внутреннем диске «Электроники» в файл `Каллисто1/B011`.

Сразу после установки Каллисто рекомендуется откомпилировать редактор командой `11 LOAD` (может занять порядка 15 минут) и сохранить редактор на диске командой `SAVE" EDIT`. Это же относится ко всем программам, которые вы намерены часто использовать на своей «Электронике». Загрузка скомпилированных программ происходит значительно быстрее, чем их интерпретация из блоков! Можно загрузить сразу несколько программ и сохранить получившийся комплект под одним именем, если на это хватит памяти.

Этот же каталог `Blocks` содержит исходный код блоков в формате TXT, удобном для прочтения человеком. Например, исходный код редактора EDIT расположен в файле `Blocks/B011.txt` Кодировка символов Unicode.

Каталог `compiled` содержит эти же блоки, заранее откомпилированные поставляемой версией Каллисто. Каждая из четырёх откомпилированных программ состоит из двух файлов с расширениями MKB и MKD, которые надо записать на электронный диск ЭКВМ в виде файлов типа B и D под одинаковым именем, под которым они загружаются из Каллисто словом `LOAD"`

Если вы самостоятельно скомпилируете `Callisto.mkp`, вам придётся компилировать заново эти файлы получившейся версией Каллисто. Номер версии Каллисто при правке исходного кода транслятора также имеет смысл изменить.

В каталоге `docs` располагается это скромное Руководство в формате PDF, в процессе написания.

В каталоге `tools` располагаются полезный инструмент для компьютера — исходный код программы `listmkt` на языке Си, которая переводит блоки из формата MKT в удобный для чтения человеком формат TXT. Чтобы запустить эту необязательную, но полезную программу, вам потребуется компилятор с языка Си.

Введение для владельцев ПМК

Каллисто переопределяет все клавиши МК-161 (*см.* Рис. 1). Для ввода английских букв нажмите чёрную клавишу ЛАТ (Р–ГРД–Г). Буквы расположены на клавишах в алфавитном порядке. Для возврата на цифровую клавиатуру нажмите синюю клавишу NUM (Р).

Чтобы набирать русские буквы, перед ЛАТ (Р–ГРД–Г) нажмите оранжевую клавишу F. В русском языке 33 буквы, поэтому на русской клавиатуре клавиши ESC (ВЫХОД), ВВОД и четыре «стрелки» работают на ввод русских букв Д, Е, А, Б, В и Г. Когда вам потребуется ввести ESC или перевести строку, переключайтесь на цифровую клавиатуру нажатием клавиши NUM (Р).

Распечатайте Рис.1 на листе А4 и прорежьте дырочки под клавиши — работать с накладкой намного удобней, чем угадывать клавишу или вычислять её по рисунку.

Для вырезания дырок под клавиши ВКЛ и подсветки лучше всего использовать пробойник на 6мм. Для остальных клавиш можно использовать пробойник на 4мм — или даже на 3мм, если умеете пробивать дырки очень точно. При отсутствии инструмента можно использовать обычные маникюрные ножницы и терпение. Именно такими ножницами я сделал свою первую накладку.

В Каллисто регистр букв роли не играет, заглавные или строчные буквы в большинстве случаев взаимозаменяемы.

Каллисто нельзя остановить клавишей С/П. Для выхода из Каллисто служит команда BYE. Также можно выключить МК-161, отключить его от зарядного устройства и включить обратно, удерживая нажатой клавишу С (С/П).

Красная клавиша BS (Cx) удаляет последний введённый символ, комбинация F Cx удаляет всю введённую строку и позволяет начать ввод заново. Клавиша ВВОД (недоступна на русской клавиатуре) передаёт введённую строку Каллисто — на исполнение или компиляцию. Клавиша ВП вводит пробел.

Для ввода порядка вместо ВП используется английская буква E, например 1E50 вместо 1 ВП 50. Чтобы её набрать, переключитесь на английскую клавиатуру, а потом обратно.

Стек бесконечный, точнее может хранить многие сотни значений. При правильно составленной программе переполнение стека маловероятно.

При вводе в стек последовательности чисел команда ↑ не требуется. Вместо 3 ↑ 15 нужно писать просто 3 15. Числа отделяются пробелом, который вводит клавиша ВП. Если вы допустили ошибку, Каллисто выводит сообщение с номером ошибки *слово*? MSG# и очищает стек. Если ошибок не было, Каллисто выводит ok и готова принять от вас следующие команды.

Команды ПМК Fsin K{x} и т.д. работают в бесконечном стеке данных и вводятся по буквам без префиксной клавиши, например sin {x} и т.д. Для переключения градусной меры угла служат слова DEGREES RADIANS. Команда y^x возводит число в степень, хранящуюся на вершине стека. Операции AND OR XOR работают так, словно в стеке хранятся 32-битные числа. Команда INVERT выполняет побитовое логическое отрицание.

В Каллисто нет регистра X1. Если вы хотите сохранить аргумент, чтобы использовать его после операции, позаботьтесь об этом самостоятельно. Чтобы вывести на индикатор число с вершины стека, используйте команду . (точка). Соответствующая клавиша располагается в нижнем ряду на месте клавиши /–/. Слово . убирает выводимое число из стека. Если вы хотите сохранить число в стеке, используйте комбинацию ↑ .

При выводе целых чисел на индикатор и вводе их с клавиатуры можно использовать разные системы счисления, придавая значения переменной BASE. Слова HEX DECIMAL устанавливают шестнадцатеричную и десятичную системы счисления.

Подпрограммы (*слова*) *определяются* с помощью конструкции `: имя тело ;`. Двоеточие осуществляет переход в режим программирования (здесь его называют *режимом компиляции*) и даёт имя вашему слову, а точка с запятой осуществляет возврат в режим автоматической работы (*исполнения*). Досрочный выход из подпрограммы осуществляет слово `EXIT`. Вложенных *вызовов слов* может быть сотни, ограничиваясь лишь размером стека возвратов. Хранить в своей памяти адреса не обязательно, каждому слову вы сами даёте нужное имя, указывая его после двоеточия.

Для вывода программой на индикатор текста используйте конструкцию `." текст"`, например:

```
: СУММА ." 2+2=" 2 2 + . ;
```

Если вы определили слово, его можно запустить на выполнения — просто набрать данное слову имя в командной строке, например `СУММА`, и нажать ВВОД. Скорость работы вашей программы будет примерно в 50 раз медленней программы на входном языке МК-161 и в 15 раз быстрее программ для советских ПМК.

Вместо номеров регистров можно использовать переменные, которым давать имена. При определении десятичной переменной `XYZ` надо задать начальное значение, например `60 VALUE XYZ`. Когда `XYZ` встречается в коде, происходит считывание (ИП) числа из регистра, в котором хранится `XYZ`. Чтобы записать число с вершины стека в переменную (П) надо употребить слово `TO`, например `TO XYZ`. При этом число не только записывается в переменную, но ещё и исчезает со стека. Если вам этого не требуется, используйте команду `↑`.

Циклы со счётчиком `FL0 FL1 FL2 FL3` записываются, как `FOR` *команды* `NEXT`. Число повторений берётся с вершины стека. Если оно равно 0 или отрицательно, цикл не выполняется ни разу. Внутри цикла узнать значение счётчика цикла можно с помощью слова `I`. Если вы используете цикл в цикле, значение счётчика внешнего цикла даст слово `J`. Количество вложений цикла ограничено только размером стека возвратов, то есть вложений могут быть сотни.

Оба условных оператора `IF` *команды* `THEN` и `IF` *команды* `ELSE` *команды-иначе* `THEN` берут число со стека. Если взятое число «истина» (отлично от нуля), выполняются *команды*. Если оно «ложь» (равно нулю), выполняются *команды-иначе*.

Чтобы сформировать число перед оператором `IF`, предусмотрены команды сравнения чисел на вершине стека. Например:

```
12 VALUE a   99 VALUE b
: ТЕСТ a b < IF ." a<b" ELSE ." a≥b" THEN ;
```

Циклы и условные операторы, как и в ЯМК, работают только в режиме программирования, точнее внутри определений через двоеточие.

Слово `KEY` отображает курсор, ожидает ввода символа с клавиатуры и кладёт на стек код введённого символа, число от 0 до 255. Слово `EMIT` берёт код символа со стека и выводит символ на индикатор. Слово `TYPE` берёт из стека адрес и длину строки, после чего выводит её на индикатор. Это всё — стандартные и давно устоявшиеся слова Форта.

С «сырой» клавиатурой МК-161 работают два слова `EKEY INKEY`. `EKEY` ожидает нажатия на клавишу МК-161 и кладёт на стек код этой клавиши, число от 0 до 37. `INKEY` кладёт на стек код нажатой клавиши или –1, если никакая клавиша в данный момент не нажата. `KEY EKEY` последовательно используют буфер клавиатуры МК-161, а `INKEY` может не заметить нажатие клавиши, если пользователь уже отпустил её к моменту проверки.

Для ввода с клавиатуры строки символов служит слово `ACCEPT`. Параметры у него такие же, как у слова `TYPE`, а возвращает оно на стеке число введённых символов, которое может оказаться меньше запрошенного. При вводе можно использовать клавиши редактирования `BS (Cx)` и `F Cx`.

МК-161 может хранить слова Каллисто не только в уже скомпилированном виде, но также их исходный код. Исходный код слов хранится на диске в *блоках* в виде *экранов,* каждый из которых содержит 48 строк по 64 символа. Экран, находящийся в блоке номер *n*, просматриваются командой *n* `LIST`, а редактируется *редактором экранов*. Текст, хранящийся в блоке номер *n*, начинает выполняться и компилироваться при выполнении команды *n* `LOAD`. На экранах можно использовать комментарии, закрывая их в круглые скобки: `(` *комментарий* `)`.

Ошибки, деление на ноль или вычисление квадратного корня из отрицательного числа, останавливают Каллисто. Чтобы начать заново, нажмите `В/О  С/П` — при этом Каллисто запустится заново, не сохранённый на диск результат работы пропадёт.

Чтобы хорошо программировать на Каллисто, необходимо достать хороший учебник по Форту и изучить его.

Выполняя упражнения из учебника, заглядывайте на вики pmk.the-hacker.ru и в исходный текст транслятора Каллисто, который находится в файле `Callisto.mkl`

Главные отличия Каллисто от стандартного Форта

Вместо циклов DO LOOP используется взятый из colorForth цикл FOR NEXT.

Для команд DUP SWAP INVERT и NEGATE используются традиционные для ПМК имена ↑ ↔ NOT и /-/.

Стек содержит десятичные числа с плавающей запятой «Электроники МК-161». Если целые числа по модулю меньше 1000 миллиардов (10^{12}), потерь значащих цифр в стеке из-за этого не происходит.

Для ввода дробных чисел используется десятичная запятая. Набор слов для целых чисел двойной точности сильно ограничен, ведь в одной ячейке стека с лёгкостью помещаются 32-битные двоичные числа и 12-разрядные десятичные числа.

При вводе знака числа или порядка нельзя использовать +, только – либо ничего.

При компиляции можно использовать целые числа только в диапазоне от –32768 до 32767. Числа за пределами этого диапазона должны содержать десятичную запятую, при программировании их надо оформлять переменными VALUE.

Для умножения и деления с плавающей запятой используются символы ПМК × и ÷. Символ / используется для целочисленного «фортовского» деления с остатком, символ * в стандартных словах не используется.

Слово NUMBER оперирует с простой ячейкой стека, а не двойной и самостоятельно обращается к слову WORD при обработке символьного литерала 'c'.

Слово . (точка) умеет работать с плавающей запятой, если активна десятичная система счисления. Слова Форта-2012 . и .R переименованы в I. и I.R.

Слова поиска 'N FINDN возвращают NFA, а не CFA. Используйте слово NAME> для получения CFA. Слово FINDN пока сохраняет отпечаток Форта ИТЭФ.

Вместо CREATE DOES> из Форта-2012 используется <BUILDS DOES> из FIG-FORTH и дополнения к FORTH-79. Различия минимальны, просто используйте <BUILDS вместо CREATE, когда компилируете определяющее слово.

Длина строки, возвращаемая (LINE), указана с учётом убранных с помощью –TRAILING пробелов в конце строки.

Стандартные слова Каллисто, которые могут быть плохо знакомы новичкам

ACCEPT из Форта-2012 используется вместо устаревшего EXPECT.

VARIABLE не требует числа на стеке для инициализации переменной, начиная с Форта-79.

Слово FOR берёт n с вершины стека и исполняет цикл FOR NEXT ровно n раз. Если n≤0, управление сразу передаётся на код за словом NEXT. Внутри цикла слово I показывает, сколько ещё раз циклу выполняться, включая текущий раз. Слово LEAVE покидает цикл сразу же.

Рекомендуется использовать порождающее слово VALUE вместо VARIABLE. Когда вам нужно хранить переменные с плавающей точкой или целые числа, большие по модулю, оно необходимо.

Для ввода чисел в других системах счисления (отличных от BASE) можно использовать префиксы Форта-2012: $1a #99 %1101 'c'.

Слова, существующие только на Каллисто

Для прямого доступа к регистрам «Электроники МК-161» служат слова П ИП. Трансляции адресов не происходит, например 13 ИП считывает на стек R13, который ещё называют RД.

Десятичные регистры организованы в *десятичный словарь (Д)*, которым управляют слова Дн Днere Дallot Д,.

Целый ряд слов, таких как sin cos lg ln 10ˣ eˣ √ x² π СЧ ЗН [x] {x} и др. взят из ЯМК.

Слово ГРФ обновляет графический экран, отображая нарисованную картинку.

Слово BEEP издаёт гудок заданной частоты и продолжительности.

Слова U# U#S U#> оперируют с простой ячейкой стека, а не двойной. Технически, слова <# SIGN уже соответствует словам Форта <# SIGN. Указанные три легко переопределить до стандарта, но лучше наоборот — переписывать определения, обращающиеся к этим удобным словам.

Слова <F# F# F#> осуществляют форматное преобразование не только целых чисел, но также десятичных дробей и чисел с плавающей запятой.

Сообщения об ошибках

При ошибке Каллисто выводит на индикатор номер ошибки и переходит в пультовый режим. Предупреждения также имеют номер ошибки, выводимый на индикатор, но остановки выполнения программы не происходит.

Некоторые слова Форта, отсутствующие в Каллисто

>MOVE<	(F79)
>NAME	(F83)
>TYPE	
;CODE	(F79)
+BLOCK	(F79)
+BUF	
+LOOP	(F79)
?DO	
!BITS	(F79)
#TIB	(F83)
.LINE	
.VOC	
2!	(F79)
2@	(F79)
2CONSTANT	(F79)
2OVER	(F79)
2ROT	(F79)
2VARIABLE	(F79)
79-STANDARD	(F79)
ABORT"	(F83)
ASCII	(F79)
ASHIFT	(F79)
AT-XY	
CLEAR	
CVARIABLE	
D+	(F79)
D-	(F79)
D.	(F79)
D.R	(F79)
D<	(F79)
DABS	(F79)
DMAX	(F79)
DMIN	(F79)
DNEGATE	(F79)
DO	(F79)
DU<	(F79)
END-CODE	(F79)
EXPECT	
FENCE	
FORGET	(F79)
I'	(F79)
IFEND	(F79)
IFTRUE	(F79)
L>NAME	(F83)
LIMIT	
LOOP	(F79)
M*	(F79)
M/	(F79)
M/MOD	(F83)
MASK	(F79)
MOVE	(F79)
MU/MOD	
O.	(F79)
OFFSET	(F79)
OTHERWISE	(F79)
REMEMBER	(F79)
ROLL	(F79 F83)
ROTATE	(F79)
SET	(F79)
SHIFT	(F79)
SPAN	(F83)
TRIAD	
U*	(F79)
U/	
U/MOD	(F79)
U.	(F79)
U.R	(F79)
U<	(F79)
UM*	(F83)
UM/MOD	(F83)
USER	(F79)
WARNING	
WHERE	(F79)
WIDTH	(FIG)

Альтернативные имена слов Каллисто в других Фортах

Каллисто	Другие реализации Форта
.	F.
?DUP	–DUP
!RP	≈ RP!
!SP	≈ SP!
>IN	IN
><	SWAB
1/x	F1/X
10ˣ	F10^X
2×	2*
2÷	F2/
/-/	NEGATE MINUS
ACCEPT	≈ EXPECT
ARCCOS	FACOS
ARCSIN	FASIN
ARCTG	FATAN
BLANK	BLANKS
BODY>	CFA
COLD	INI INIT
CONVERT	(NUMBER)
COS	FCOS
eˣ	FEXP
H	DP
I	R R@
I.	.
I.R	.R
U#	≈ #
U#>	≈ #>
U#S	≈ #S
NOT	INVERT
LAST	LATEST
LG	FLOG
LN	FLN
N>BODY	PFA
S>D	S->D
SIN	FSIN
TG	FTAN
THEN	ENDIF
UNTIL	END
VALUE	QUAN
WORDS	VLIST (F79)
x²	FX^2
yˣ	** F**
[x]	INT
|x|	ABS
\S	;S (F79)
↑	DUP
↔	SWAP
{x}	FRAC
×	* (F79)
×/	*/ (F79)
×/MOD	*/MOD
÷	F/
↵	CR (F79)
↖	PAGE (F79)
√	FSQRT
π	FPI
ЗН	FSGN
СЧ	FRAN

Слова Каллисто, не имеющие аналогов в Форте

(PLAY)	(a u ––)	Проиграть мелодию из u нот по адресу a
AUTOEXEC	(–– a)	Переменная автозапуска после LOAD"
INKEY	(–– c)	Опрос клавиатуры
LOAD"	(––)	Загрузить словари с диска
SAVE"	(––)	Сохранить словари на диск
TYPE1	(a u ––)	Вывести в одну строку индикатора не более u символов, начиная с адреса a
ГРФ	(––)	Обновить графический экран
Д,	(x ––)	Записать число в очередной десятичный регистр
ДALLOT	(u ––)	Зарезервировать u десятичных регистров
ДН	(–– a)	Переменная, хранящая вершину десятичного словаря
ДHERE	(–– u)	Номер первого свободного десятичного регистра
ИП	(u –– x)	Прочитать регистр номер u
П	(x u ––)	Записать x в регистр номер u
(ИП)	(–– r)	Считать значение регистра, зашитого в шитом коде

Определения слов, отсутствующих в ядре Каллисто

```
: <>  ( n1 n2 -- f)   = 0= ;                          ( F79)
: >LINK  ( CFA -- LFA)   2- ;                         ( F83)
: ?  ( a --)   @ . ;                                  ( F79)
: R@  ( -- 16b)   R> I ↔ >R ;                         ( F79 F83)
; 0!  ( a --)   0 ↔ ! ;
: 1+!  ( a --)   1 ↔ +! ;                             ( F79)
: 1-!  ( a --)   -1 ↔ +! ;                            ( F79)
: -ROT  ( a b c -- c b a)   ROT ROT ;
: BINARY  ( --)   2 BASE ! ;
: COMPILE,  ( a --) , ;                               ( F94)
: D=  ( d1 d2 --)   ROT XOR -ROT XOR - 0= ;           ( F79)
: D0=  ( d --)   OR 0= ;                              ( F79)
: HOME  ( --)   0 ↑ AT ;
: LINK>  ( LFA -- CFA)   2 + ;                        ( F83)
: OCTAL  ( --)   8 BASE ! ;                           ( F79)
: RECURSE  ( --)   LATEST NAME> , ; IMMEDIATE         ( F83 aka MYSELF )
: STRING  ( c --)   WORD C@ 1+ ALLOT ;                ( Броуди)
: РЕГ?  ( u --)   ИП . ;

2 CONSTANT CELL
0 CONSTANT FALSE
-1 CONSTANT TRUE
```

Слова ядра Каллисто, которые можно определить самостоятельно

```
: 2DROP  ( dd --)   DROP DROP ;
: 2SWAP  ( dd1 dd2 -- dd2 dd1)   ROT >R ROT R> ;
: 2DUP  ( dd -- dd dd)   OVER OVER ;
```

Машинная графика

AT	(x y ––)	начальная координата
+BOX	(dx dy ––)	закрашенный прямоугольник
+FRAME	(dx dy ––)	рамка
BAR	(x y ––)	отрезок от AT до (x,y)
DOT!	(x y ––)	установка точки
DARK	(––)	устанавливает чёрный цвет (рисует)
LIGHT	(––)	устанавливает белый цвет (стирает)
ATR	(a ––)	единый адрес регистра, где хранится атрибут вывода

Определения синонимов Форта через слова Каллисто

```
: ALIAS ( "<пр>новое_слово <пр>старое_слово" --) CREATE 'N NAME> U@ LAST
NAME> ! ;
```

Каллисто	FORTH
--------	-----

Как Каллисто использует регистры MK-161

R0		временный регистр слова-примитива
R1		временный регистр слова-примитива
R2	RP	указатель целочисленного стека возвратов (RS) — растёт вниз от 5092
R3	SP	указатель десятичного стека (DS с плавающей запятой) — растёт вниз от 998
R4		временный регистр слова-примитива
R5		временный регистр слова-примитива, может быть изменён NEXT, CALL, EXIT
R6	RI	указатель шитого кода, он же IP указатель инструкций, указывает на следующий CFA (когда R9==NEXTD)
R7	W	указатель слов Форта (WP), содержит CFA исполняемого слова, слово-примитив может менять R7
R8	JP	указатель/регистр передачи управления NEXT, временный регистр слова-примитива
R9	=NEXTP/NEXTD, адрес кода NEXT	
RA		временный регистр, его может изменить BIOS
RB		временный регистр слова-примитива
RC	=RPUSHRIP/RPUSHRID	
		адрес кода сохранения указателя шитого кода RI в стеке возвратов
RD		
RE	=256, литерал	
R19		при загрузке/сбросе словаря содержит номер версии Каллисто
R9042	RI	указатель шитого кода (когда R9==NEXTP)
R9044		читает байт шитого кода, RI++

Единое адресное пространство

Единая адресация Каллисто объединяет пространства регистров и памяти программ так:

0.. 9999	байты памяти программ, только для чтения -- система ввода-вывода, словарь Форта (начало)	
10000..10999	десятичные регистры (R0…R999):	
	10000..10049	регистры Форта, служебные и зарезервированные
	10020..ДHERE-1	десятичный словарь
	SP..10997	стек данных (параметров), должен вмещать минимум 32 регистра по Форт-79
11000..15095	двоичные регистры (4 Кб, R1000…R5095):	
	11000..11095	входной буфер терминала (TIB) и признак конца ввода (2 байта),
	11096..11155	область переменных Форта (USER),
	11156..HERE-1	словарь Форта (продолжение), должен вмещать минимум 2000 байт по Форт-79
	HERE..HERE+84	буфер форматированного вывода (PAD), должен вмещать минимум 84 байта по Форту-83
	RP..15089	целочисленный стек возвратов, должен вмещать минимум 48 байт по Форт-79
	15094..15095	номер блока, загруженного в буфер и его флаг сохранения (2 байта)
15096..18167	область текста, буфер с исходным текстом для компиляции (3 Кб, R5096…R8167)	
18168…18999	не используется (всегда 0)	
19000..19999	регистры функций «Электроники МК-161»	

Дно стека записано в переменной `S0`, а дно стека возвратов — в переменной `R0`.

Хранение чисел в памяти

Стандарт размещения многобайтных слов: в двоичном виде, старшее по младшему адресу.

Операции с беззнаковыми 16-битными числами добавляют к отрицательным аргументам 65536.

Операции с беззнаковыми 32-битными числами добавляют к отрицательным аргументам 65536^2 = 4 294 967 296.

Словарная статья

Каллисто использует косвенный шитый код. Определения через двоеточие состоят из последовательности скомпилированных CFA — указателей на поля кода используемых слов. Токен исполнения xt представляет из себя CFA — адрес поля кода соответствующей статьи.

Адрес в самом поле кода может указывать только на ЯМК в памяти программ.

Словарная статья состоит из 4 полей: *Name Link Code Parameter*

Заголовок статьи это первые два поля: *Name Link*

После заголовка идёт тело статьи: *Code Parameter*

где *Name* это поле имени: *флаг-счётчик имя*

Link это поле связи (2 байта) — NFA предыдущего слова

Code это поле кода (2 байта) — ссылка на интерпретатор слова, то есть исполняемый код ЯМК в памяти программ

Parameter (размер не фиксирован) это поле параметров: реализация слова — шитый код, код на ЯМК, область данных и т.д.

Адреса этих полей называются NFA LFA CFA PFA

Все поисковые слова Каллисто дают NFA, откуда путём преобразований получаем адреса любых других полей. Обратная операция, получение NFA слова по адресу других полей, в Каллисто невозможна.

Флаги в первом байте поля имени: 80H IMMEDIATE, 20H SMUDGE (HIDDEN), 40H зарезервирован и пока не используется.

Блоки, хранение исходного кода и его компиляция

Наши блоки занимают 3 Кбайта и обычно хранят экраны с исходным текстом, 48 строк по 64 символа. Нумерация строк осуществляется с 0 до 47. В строке номер 0 принято хранить *индекс* (заголовок) — комментарий с кратким содержанием экрана. Примерно первые два десятка символов нулевой строки — самые важные и именно они выводятся из разных блоков с помощью слова *нач кон* `INDEX`.

По стандарту Форта-79 должно быть минимум 32 блока по 1 Кбайту. На диск МК-161 влезает чуть больше 150 блоков. Слово `BLOCK` (n –– a) загружает блок в память и возвращает адрес его первого байта.

Блоки размещаются на диске A в подкаталогах Каллисто1 Каллисто2 Каллисто3 и Каллисто4 главного каталога. Каждый подкаталог содержит до 60 файлов типа T (текст) с именами B001..B999. Если блок или подкаталог не существуют, Каллисто их создаёт.

Вывести на индикатор блок можно с помощью слова *n* `LIST`. Загрузить блок и выполнить оттуда команды, в том числе компиляцию новых слов, можно с помощью слова *n* `LOAD`. При этом блок не должен опустошать стек или класть туда новые слова, это помешает восстановлению работы Каллисто после компиляции блока.

Сохранить на диск объектный код (словарь и переменные, всё состояние Каллисто) можно с помощью слова `SAVE" `*имя-файла*`"`., а считать обратно с помощью `LOAD" `*имя-файла*`"`. Если эта команда — последняя во вводимой строке, закрывающую кавычку можно опустить.

Сохранения между разными версиями Каллисто несовместимы, но в пределах одной версии должны работать на любом МК-161. Сохранения могут использоваться для распространения вашей программы, если поставляются с соответствующей версией Каллисто.

Каталог блоков

* Блоки №1-10 зарезервированы под будущие версии Каллисто. Для совместимости с ними их использование не рекомендуется.

* Блок №11 содержит строковый редактор `EDIT`

* Блок №41 содержит пример «Подмосковные вечера», позволяющий создавать на Каллисто простейшие одноголосые мелодии

* Блок №42 содержит инструмент «Секундомер» для измерения скорости работы фрагментов программ

* Блок №43 содержит пример «Музыка Pacman»

В каталоге compiled дистрибутива хранятся эти четыре, уже откомпилированные программы.

Строковый редактор EDIT

Строковый редактор EDIT предназначен для редактирования 3Кб экранов Каллисто, состоящих из 48 строчек по 64 символов каждая. Строки нумеруются с нулевой (где обычно располагается индекс, справочный комментарий ко всему экрану) по 47-ую.

Стандартное расположение исходного кода редактора EDIT — блок номер 11. Компиляция редактора по команде `11 LOAD` может занять 15 минут. Чтобы в дальнейшем загружать редактор быстрее, запишите скиньте его на диск сразу после компиляции с помощью команды `SAVE" EDIT`. Также можно перенести на электронный диск уже откомпилированный редактор из поставки Каллисто, состоящий из двух файлов.

Если в описании после команды идёт слово «текст», это означает, что любой текст, набранный после команды, будет скопирован в текстовый буфер, который данная команда использует. Содержимое буфера будет использовано во время выполнения команды. Если пользователь не ввёл никакого текста, во время исполнения команды будет использовано прошлое содержимое буфера, оставшееся от предыдущих команд.

Номер редактируемого экрана берётся из переменной `SCR` — туда его заносят команды `LIST` и `EDIT`, но можно занести и вручную. Все слова редактора располагаются в лексиконе `EDITOR`. Команды редактирования изменяют экран в памяти МК-161, но не на диске. Чтобы сохранить результат редактирования, используйте команду `SV`.

`EDIT` EDIT (*#блока* ––)

Приступить к редактированию экрана, расположенного в блоке *#блока*. #блока записывается в переменную `SCR` и текущим контекстом объявляется `EDITOR`.

`SV` SaVe (––)

Записывает отредактированный экран на диск.

`X` eXtract (––)

Копирует текущую строку в буфер INSERT и удаляет её с экрана. Все последующие строчки сдвигаются вверх, строка 47 остаётся пустой.

`T` Type (*n* ––)

Вывести строку *n* текущего экрана. Поставить курсор на начало этой строки. Часто используется для задании номера строки перед командами `P X U` и др.

`L` List (––)

Выводит редактируемый экран на индикатор. Как команда Каллисто `LIST`, только номер экрана берётся из `SCR` — её не приходится набирать. Не меняет позиции курсора.

`LS` LineS (*нач кон* ––)

Вывести строчки текущего экрана с *нач* по *кон*. Команда `LS` не меняет позиции курсора и не выводит его позиции.

N Next (––)

Увеличивает номер текущего экрана на 1. Если старый экран не сохранён по команде SV, его содержимое пропадает.

B Back (––)

Уменьшает номер текущего экрана на 1. Если старый экран не сохранён по команде SV, его содержимое пропадает.

P Put (––)

P *текст*

Любой идущий следом *текст* будет скопирован в буфер INSERT. Буфер INSERT будет скопирован в текущую строку, заменяя её предыдущее содержимое. Если текст состоит из одного или более пробелов, текущая строка будет стёрта (заменена пробелами).

WIPE Wipe (––)

Стирает текущий экран, забивая его пробелами. Эквивалент команды CLEAR (у нас такой пока нет — AtH), только пользователю не нужно вводить номер экрана.

COPY Copy (*откуда куда* ––)

Копирует один экран на другой. У нас это команда ядра Каллисто и доступна всегда, даже без загрузки редактора.

I Insert (––)

I *текст*

Любой идущий следом *текст* будет скопирован в буфер INSERT. Команда I копирует содержимое буфера INSERT в текущую строчку, начиная с текущей позиции курсора. Любой текст справа от курсора будет сдвинут вправо, за пределы строки и потерян, если общая длина строки превысит 64 символа.

U Under (––)

U *текст*

Любой последующий *текст* будет скопирован в буфер INSERT. Раздвигает экран на одну линию, непосредственно ПОД текущей, вставляя пустую строку. Все последующие линии сдвигаются вниз. Любое содержимое строки 47 будет потеряно. Содержимое буфера INSERT будет скопировано на пустую строку, и эта линия станет текущей строкой.

M Move (*#блока #строки* ––)

Копирует текущую строку в буфер INSERT, затем копирует содержимое буфера INSERT в блок, указанный *#блока*, ПОД строкой, указанной в *#строки*. Исходный номер блока восстанавливается, а следующая линия в блоке становится текущей. Это позволяет копировать последовательные линии минимальным количеством нажатий на клавиши. К сожалению, побочный эффект этой команды — чтобы скопировать что-то в строку 0 на другом экране, вам надо вначале скопировать это ПОД строку 0, используя команду *xxx* 0 M, сделать экран *xxx* текущим и затем извлечь (X) старую строку 0, пододвинув всё вверх.

^ (––)

Используется как разделитель для любых команд, позволяющих ввод текста — таких как P и U. Позволяет вводить более одной команды на той же самой строчке, например:

```
3 T P  Это строчка 3^ L  (ввод)
```

Хотя эта особенность полезна, она не позволяет использовать «^» как символ в любом тексте, который записывается на экран.

<<<EOF>>>