Лекция 12

**MSP430** — семейство 16-разрядных [микроконтроллеров](https://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D0%BA%D1%80%D0%BE%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D0%BB%D0%B5%D1%80) фирмы [Texas Instruments](https://ru.wikipedia.org/wiki/Texas_Instruments).

**Содержание**

* [1История создания и особенные характеристики](https://ru.wikipedia.org/wiki/MSP430#%D0%98%D1%81%D1%82%D0%BE%D1%80%D0%B8%D1%8F_%D1%81%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D1%8F_%D0%B8_%D0%BE%D1%81%D0%BE%D0%B1%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5_%D1%85%D0%B0%D1%80%D0%B0%D0%BA%D1%82%D0%B5%D1%80%D0%B8%D1%81%D1%82%D0%B8%D0%BA%D0%B8)
* [2Система команд MSP430](https://ru.wikipedia.org/wiki/MSP430#%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4_MSP430)
* [3Мнемоника эмулируемых команд](https://ru.wikipedia.org/wiki/MSP430#%D0%9C%D0%BD%D0%B5%D0%BC%D0%BE%D0%BD%D0%B8%D0%BA%D0%B0_%D1%8D%D0%BC%D1%83%D0%BB%D0%B8%D1%80%D1%83%D0%B5%D0%BC%D1%8B%D1%85_%D0%BA%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4)
* [4Примечания](https://ru.wikipedia.org/wiki/MSP430#%D0%9F%D1%80%D0%B8%D0%BC%D0%B5%D1%87%D0%B0%D0%BD%D0%B8%D1%8F)
* [5Ссылки](https://ru.wikipedia.org/wiki/MSP430#%D0%A1%D1%81%D1%8B%D0%BB%D0%BA%D0%B8)
  + [5.1Компиляторы и ассемблеры](https://ru.wikipedia.org/wiki/MSP430#%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80%D1%8B_%D0%B8_%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80%D1%8B)
  + [5.2Эмуляторы](https://ru.wikipedia.org/wiki/MSP430#%D0%AD%D0%BC%D1%83%D0%BB%D1%8F%D1%82%D0%BE%D1%80%D1%8B)
  + [5.3Полезные ссылки](https://ru.wikipedia.org/wiki/MSP430#%D0%9F%D0%BE%D0%BB%D0%B5%D0%B7%D0%BD%D1%8B%D0%B5_%D1%81%D1%81%D1%8B%D0%BB%D0%BA%D0%B8)

**История создания и особенные характеристики**

[[править](https://ru.wikipedia.org/w/index.php?title=MSP430&veaction=edit&section=1) | [править код](https://ru.wikipedia.org/w/index.php?title=MSP430&action=edit&section=1)]

Первый контроллер с аббревиатурой MSP430 появился в [1992 году](https://ru.wikipedia.org/wiki/1992_%D0%B3%D0%BE%D0%B4)[[1]](https://ru.wikipedia.org/wiki/MSP430#cite_note-1). [Texas Instruments](https://ru.wikipedia.org/wiki/Texas_Instruments) заявил о стремлении создать микроконтроллер с системой команд, близкой к популярной в 70-х—80-х годах архитектуре [PDP-11](https://ru.wikipedia.org/wiki/PDP-11).

Ядро MSP430 16-битное. Систему команд постарались сделать максимально [ортогональной](https://ru.wikipedia.org/w/index.php?title=%D0%9E%D1%80%D1%82%D0%BE%D0%B3%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%BA%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4&action=edit&redlink=1) с разнообразными способами адресации. Ортогональность системы команд означает, что в любой команде может использоваться любой способ адресации операнда: константа, прямой из регистра или памяти, косвенный и т. п.

В момент рождения семейства основной упор был сделан на снижение энергопотребления. Однако с тех пор экономия энергии стала идеей-фикс электронной техники, и MSP430 активно теснят на этом пьедестале другие производители со своими архитектурами.

Ключевым отличием и «визитной карточкой» семейства MSP430 является возможность тактировать любой модуль периферии асинхронно от ядра. В подавляющем большинстве однокристальных микроконтроллеров периферия синхронна с ядром (за исключением отдельных специальных узлов). Такая особенность позволяет гибко управлять скоростью (а значит, и потреблением) каждого модуля. Кроме того, уникальным является также модуль [USB](https://ru.wikipedia.org/wiki/USB) интерфейса: он имеет отдельный домен питания, что упрощает разработку батарейных приборов с USB-интерфейсом.

Сегодня семейство MSP430 предлагает широкую гамму однокристальных микроконтроллеров с объёмом [флеш-памяти](https://ru.wikipedia.org/wiki/%D0%A4%D0%BB%D0%B5%D1%88-%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C) от единиц до 512 килобайт и [ОЗУ](https://ru.wikipedia.org/wiki/%D0%9E%D0%97%D0%A3) до 64 килобайт. Тактовые частоты ядра — до 25 МГц.

**Система команд MSP430**

[[править](https://ru.wikipedia.org/w/index.php?title=MSP430&veaction=edit&section=2) | [править код](https://ru.wikipedia.org/w/index.php?title=MSP430&action=edit&section=2)]

MSP430 имеет [фоннеймановскую](https://ru.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D0%B8%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D1%84%D0%BE%D0%BD_%D0%9D%D0%B5%D0%B9%D0%BC%D0%B0%D0%BD%D0%B0) архитектуру, с единым адресным пространством для команд и данных. Память может адресоваться как побайтово, так и пословно. Порядок хранения 16-разрядных слов — [от младшего к старшему](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%80%D1%8F%D0%B4%D0%BE%D0%BA_%D0%B1%D0%B0%D0%B9%D1%82%D0%BE%D0%B2#%D0%9F%D0%BE%D1%80%D1%8F%D0%B4%D0%BE%D0%BA_%D0%BE%D1%82_%D0%BC%D0%BB%D0%B0%D0%B4%D1%88%D0%B5%D0%B3%D0%BE_%D0%BA_%D1%81%D1%82%D0%B0%D1%80%D1%88%D0%B5%D0%BC%D1%83) ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *little-endian*).

Процессор содержит 16 16-разрядных ортогональных регистров. Регистр R0 используется как программный счетчик ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Program Counter — PC*), регистр R1 как указатель стека ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Stack Pointer — SP*), регистр R2 как регистр статуса ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Status Register — SR*), а R3 как специальный регистр, именуемый «генератор констант» ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Constant Generator — CG*), R2 также может использоваться в качестве генератора констант. Генератор констант используется для сокращения общей длины команды вследствие неявного представления константы в коде операции. Регистры с R4 по R15 используются как регистры общего назначения.

Набор инструкций очень простой и представлен 27 инструкциями, 24 эмулированными инструкциями. Инструкции имеют как 8-битную (байт), так и 16-битную (слово) форму обработки операндов. Бит B/W управляет этим признаком.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Система команд** | | | | | | | | | | | | | | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** | **Команда** |
| 0 | 0 | 0 | 1 | 0 | 0 | opcode | | | B/W | As | | register | | | | Однооперандные команды |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | B/W | As | | register | | | | **RRC** Вращение вправо через перенос |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | As | | register | | | | **SWPB** Обмен байтов |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | B/W | As | | register | | | | **RRA** Вращение вправо арифметическое |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | As | | register | | | | **SXT** Расширение знака байта до слова |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | B/W | As | | register | | | | **PUSH** Опустить операнд в стек |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | As | | register | | | | **CALL** Вызов подпрограммы; сохранить PC в стеке и загрузить PC новым значением |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **RETI** Возврат из прерывания; Извлечь SR и PC из стека |
| 0 | 0 | 1 | условие | | | 10-бит знаковое смещение | | | | | | | | | | Условный переход; PC = PC + 2×offset |
| 0 | 0 | 1 | 0 | 0 | 0 | 10-бит знаковое смещение | | | | | | | | | | **JNE**/**JNZ** Переход, если не\_равно/не\_ноль |
| 0 | 0 | 1 | 0 | 0 | 1 | 10-бит знаковое смещение | | | | | | | | | | **JEQ**/**JZ** Переход, если равно/ноль |
| 0 | 0 | 1 | 0 | 1 | 0 | 10-бит знаковое смещение | | | | | | | | | | **JNC**/**JLO** Переход, если не\_перенос/ниже (беззнаковое сравнение) |
| 0 | 0 | 1 | 0 | 1 | 1 | 10-бит знаковое смещение | | | | | | | | | | **JC**/**JHS** Переход, если перенос/выше или то же (беззнаковое сравнение) |
| 0 | 0 | 1 | 1 | 0 | 0 | 10-бит знаковое смещение | | | | | | | | | | **JN** Переход, если отрицательный |
| 0 | 0 | 1 | 1 | 0 | 1 | 10-бит знаковое смещение | | | | | | | | | | **JGE** Переход, если больше\_или\_равно |
| 0 | 0 | 1 | 1 | 1 | 0 | 10-бит знаковое смещение | | | | | | | | | | **JL** Переход, если меньше |
| 0 | 0 | 1 | 1 | 1 | 1 | 10-бит знаковое смещение | | | | | | | | | | **JMP** Переход (непосредственный) |
| opcode | | | | source | | | | Ad | B/W | As | | destination | | | | Двухоперандная арифметика |
| 0 | 1 | 0 | 0 | source | | | | Ad | B/W | As | | destination | | | | **MOV** Переслать источник в приёмник |
| 0 | 1 | 0 | 1 | source | | | | Ad | B/W | As | | destination | | | | **ADD** Прибавить источник к приёмнику |
| 0 | 1 | 1 | 0 | source | | | | Ad | B/W | As | | destination | | | | **ADDC** Прибавить источник\_и\_перенос к приёмнику |
| 0 | 1 | 1 | 1 | source | | | | Ad | B/W | As | | destination | | | | **SUBC** Вычесть источник из приёмника (с переносом) |
| 1 | 0 | 0 | 0 | source | | | | Ad | B/W | As | | destination | | | | **SUB** Вычесть источник из приёмника |
| 1 | 0 | 0 | 1 | source | | | | Ad | B/W | As | | destination | | | | **CMP** Сравнить (операцией вычитания) источник с приёмником |
| 1 | 0 | 1 | 0 | source | | | | Ad | B/W | As | | destination | | | | **DADD** [Decimal](https://ru.wikipedia.org/w/index.php?title=Binary-coded_decimal&action=edit&redlink=1) Десятичное сложение источника и приёмника (с переносом) |
| 1 | 0 | 1 | 1 | source | | | | Ad | B/W | As | | destination | | | | **BIT** Проверка битов (операцией AND) источника и приёмника |
| 1 | 1 | 0 | 0 | source | | | | Ad | B/W | As | | destination | | | | **BIC** Битовая очистка (dest &= ~src) |
| 1 | 1 | 0 | 1 | source | | | | Ad | B/W | As | | destination | | | | **BIS** Битовая установка (logical OR) |
| 1 | 1 | 1 | 0 | source | | | | Ad | B/W | As | | destination | | | | **XOR** [Исключающее или](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D0%BA%D0%BB%D1%8E%D1%87%D0%B0%D1%8E%D1%89%D0%B5%D0%B5_%D0%B8%D0%BB%D0%B8) источника с приёмником |
| 1 | 1 | 1 | 1 | source | | | | Ad | B/W | As | | destination | | | | **AND** [Логический AND](https://ru.wikipedia.org/w/index.php?title=%D0%9B%D0%BE%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_AND&action=edit&redlink=1) источника с приёмником (dest &= src) |

Все инструкции 16-битные. 4 способа адресации операнда, определены в 2 битах как As поле.

Поле As — регистровый, индексный, косвенный-регистровый, косвенно-регистровый с постдекрементом. Поле Ad определяет два способа адресации — регистровый и индексный.

|  |  |  |  |
| --- | --- | --- | --- |
| **Режим адресации** | | | |
| **As** | **Регистр** | **Синтаксис** | **Описание** |
| 00 | *n* | **R*n*** | Регистровый. Операнд — содержимое одного из регистров из R*n* |
| 01 | *n* | ***x*(R*n*)** | Индексный. Операнд находится в памяти по адресу R*n*+*x*. X-слово находится после текущей команды |
| 10 | *n* | **@R*n*** | Косвенный регистровый. Операнд находится в памяти по адресу, который содержится в регистре R*n* |
| 11 | *n* | **@R*n*+** | Косвенный регистровый с автоинкрементом. В зависимости от значения разряда B/W значение регистра R*n* увеличивается после выполнения операции на 1 или 2 |
| Режимы адресации при использовании R0 (PC) | | | |
| 01 | 0 (PC) | ***LABEL*** | Относительный(символьный). Операнд x (PC) в памяти по адресу PC+*x* |
| 11 | 0 (PC) | **#*x*** | Непосредственный. @PC+ Адрес операнда из х-слова, находящегося после текущей команды |
| Использование R2 (SR) и R3 (CG), специальный способ декодирования | | | |
| 01 | 2 (SR) | **&*LABEL*** | Абсолютный. Операнд в памяти по адресу, взятому из *x* |
| 10 | 2 (SR) | **#4** | Константа 4 |
| 11 | 2 (SR) | **#8** | Константа 8 |
| 00 | 3 (CG) | **#0** | Константа 0 |
| 01 | 3 (CG) | **#1** | Константа 1 при байтовых операциях |
| 10 | 3 (CG) | **#2** | Константа 2 |
| 11 | 3 (CG) | **#-1** | Константа −1 или 0xFFFF |

**Мнемоника эмулируемых команд**

[[править](https://ru.wikipedia.org/w/index.php?title=MSP430&veaction=edit&section=3) | [править код](https://ru.wikipedia.org/w/index.php?title=MSP430&action=edit&section=3)]

Другие команды, поддерживаемые ассемблером MSP430, образуются из основных и именуются эмулируемыми (способ получения — в скобках). Общее число поддерживаемых ассемблером эмулируемых команд — 24:

1. **CLRZ** — очистка флага Z регистра состояния процессора (PSW) (BIC #2,SR).
2. **CLRN** — очистка флага N регистра состояния процессора (PSW) (BIC #4,SR).
3. **CLRC** — очистка флага C регистра состояния процессора (PSW) (BIC #1,SR).
4. **SETZ** — установка флага Z регистра состояния процессора (PSW) (BIS #2,SR).
5. **SETN** — установка флага N регистра состояния процессора (PSW) (BIS #4,SR).
6. **SETC** — установка флага C регистра состояния процессора (PSW) (BIS #1,SR).
7. **EINT** — разрешение прерываний (BIC #8,SR).
8. **DINT** — запрещение прерываний (BIS #8,SR).
9. **CLR** dst — очистка операнда (MOV #0,dst).
10. **TST** dst — проверка операнда на ноль (CMP #0,dst).
11. **INV** dst — инвертирование битов операнда (XOR #-1,dst).
12. **ADC** dst — прибавление переноса к операнду (ADDC #0,dst).
13. **DADC** dst — десятичное сложение переноса с получателем (DADD #0,dst).
14. **SBC** dst — вычитание переноса из операнда (SUBC #0,dst).
15. **INC** dst — инкремент операнда (ADD #1,dst).
16. **DEC** dst — декремент операнда (SUB #1,dst)
17. **INCD** dst — увеличение на 2 операнда (ADD #2,dst).
18. **DECD** dst — уменьшение на 2 операнда (SUB #2,dst).
19. **RLA** dst — сдвиг влево операнда, флаг переноса заполняется из старшего бита, а младший бит — результата −0 (ADD dst, dst).
20. **RLC** dst — сдвиг влево операнда с использованием переноса (ADDC dst, dst).
21. **RET** — возврат из подпрограммы (MOV @sp+,pc).
22. **POP** dst — извлечение операнд из стека (MOV @sp+,dst).
23. **BR** dst — переход в программе, используя операнд (MOV dst, pc).
24. **NOP** — нет операции (MOV r3,r3).

Примечание: приведена форма записи команд без указания на тип операндов байт/слово. Имеются и другие возможные операции для формирования задержки выполнения программного кода.

Поддерживаемый формат команд ассемблером в мнемонике имеет указание на тип обрабатываемых данных.

**Примечания**

[[править](https://ru.wikipedia.org/w/index.php?title=MSP430&veaction=edit&section=4) | [править код](https://ru.wikipedia.org/w/index.php?title=MSP430&action=edit&section=4)]

* 1. [↑](https://ru.wikipedia.org/wiki/MSP430#cite_ref-1) *BY STEPHEN EVANCZUK.* [The most-popular MCUs ever](https://www.edn.com/slideshow-the-most-popular-mcus-ever/) (англ.). [*https://www.edn.com/*](https://www.edn.com/) (20 августа 2013). Дата обращения: 15 января 2021. [Архивировано](https://web.archive.org/web/20210609202443/https:/www.edn.com/slideshow-the-most-popular-mcus-ever/) 9 июня 2021 года.

**Ссылки**

[[править](https://ru.wikipedia.org/w/index.php?title=MSP430&veaction=edit&section=5) | [править код](https://ru.wikipedia.org/w/index.php?title=MSP430&action=edit&section=5)]

* [Официальная страница MSP430 на сайте Texas Instruments](http://www.ti.com/msp430)
* [Форум по микроконтроллерам MSP430](https://web.archive.org/web/20091023050337/http:/caxapa.ru/msp.html)
* [Семейство микроконтроллеров MSP430x1xx: руководство пользователя.](http://www.gaw.ru/html.cgi/txt/doc/micros/msp430/arh/index.htm)
* [Семейство микроконтроллеров MSP430x4xx: руководство пользователя.](http://www.gaw.ru/html.cgi/txt/doc/micros/msp430/arh430x4xx/index.htm)
* [MSP430 примеры применения](http://www.gaw.ru/html.cgi/txt/app/micros/msp430/index.htm)

**Компиляторы и ассемблеры**

[[править](https://ru.wikipedia.org/w/index.php?title=MSP430&veaction=edit&section=6) | [править код](https://ru.wikipedia.org/w/index.php?title=MSP430&action=edit&section=6)]

* [VisSim/ECD](http://www.vissim.com/solutions/embedded_development.html) позволяет быстро создать прототип для контроля приложений управляется DSP от [Texas Instruments](https://ru.wikipedia.org/wiki/Texas_Instruments). Для MSP430, [VisSim](https://ru.wikipedia.org/wiki/VisSim) нужно только 740 байт FLASH и 64 байт [ЗУПВ](https://ru.wikipedia.org/wiki/%D0%97%D0%A3%D0%9F%D0%92) для небольших закрытых [модуляцией ширины импульса петля](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%80%D0%BE%D1%82%D0%BD%D0%BE-%D0%B8%D0%BC%D0%BF%D1%83%D0%BB%D1%8C%D1%81%D0%BD%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D1%83%D0%BB%D1%8F%D1%86%D0%B8%D1%8F) (PWM) системы.
* [AQ430 Среда разработки для MSP430Fxxxx микроконтроллеров](http://www.quadravox.com/AQ430.htm)
* [CrossWorks С-компилятор MSP430](http://www.rowley.co.uk/)
* [компилятор GCC для MSP430 (Свободный C-компилятор)](https://web.archive.org/web/20030426185442/http:/mspgcc.sourceforge.net/)
* [HI-TECH C-компилятор для MSP430](http://www.htsoft.com/)
* [IAR С-компилятор для TI MSP430 фирмы](http://www.iar.com/)[IAR Systems](https://ru.wikipedia.org/w/index.php?title=IAR_Systems&action=edit&redlink=1)
* [Среда разработки Code::Blocks](http://www.shelezyakin.ru/?p=816)
* [C-компилятор фирмы](http://www.imagecraft.com/)[ImageCraft](https://ru.wikipedia.org/w/index.php?title=ImageCraft&action=edit&redlink=1)
* [ForthInc Forth-компилятор](http://www.forth.com/embedded/swiftx-embedded-systems-15.html)
* [MPE Forth-компилятор](http://www.mpeltd.demon.co.uk/forth6.htm) [Архивная копия](https://web.archive.org/web/20200220053627/http:/www.mpeltd.demon.co.uk/forth6.htm) от 20 февраля 2020 на [Wayback Machine](https://ru.wikipedia.org/wiki/Wayback_Machine)
* [naken430asm — ассемблер, дизассемблер, симулятор для 430 серии, Windows, MacOS, Linux](http://www.mikekohn.net/micro/naken430asm_msp430_assembler.php)
* [energia — форк arduino IDE, позволяющие программировать в стиле arduino](https://github.com/energia/Energia/wiki/Getting-Started)