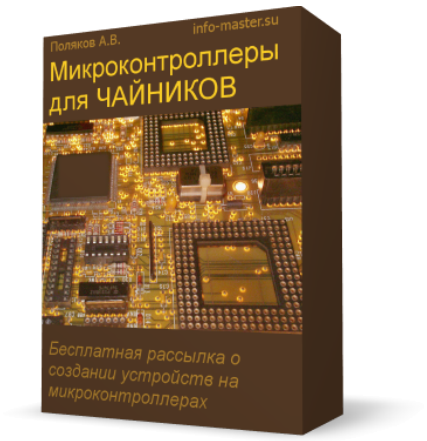
Как стать программистом

Бесплатная книга о программировании
для начинающих и бывалых.

Получить >>>

Подписаться:



Микроконтроллеры для ЧАЙНИКОВ
Изучать БЕСПЛАТНО

14.09.2020 г.
Добавлена статья Уменьшение энергопотребления.

05.09.2020 г.

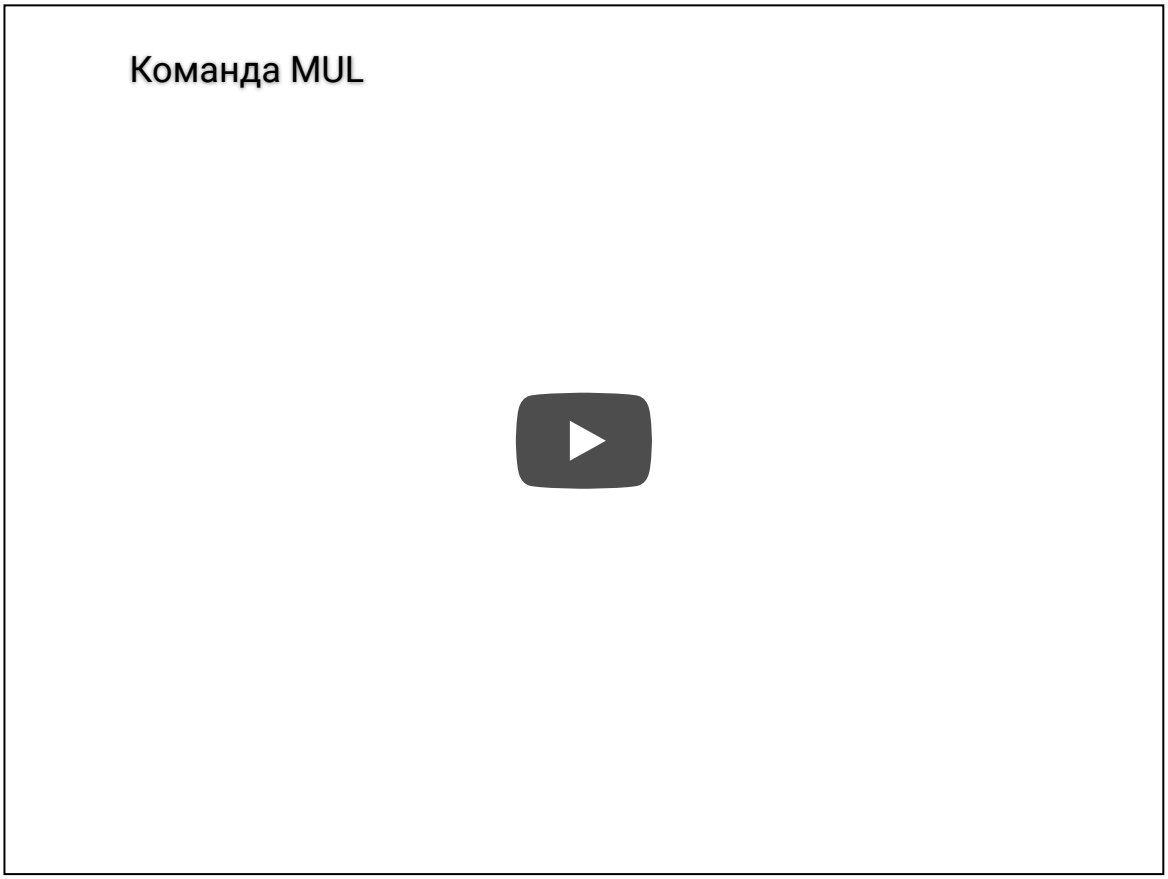
Добавлены видео и статья Самое простое устройство на микроконтроллере.

21.08.2020 г.
Добавлены видео и статья Инструкция CLI.

19.06.2020 г.
Добавлена статья Выводы ATtiny13A.

19.05.2020 г.
Добавлена статья Регистр PRR.

Команда MUL



Инструкция MUL в Ассемблере выполняет умножение без знака. Понять работу команды MUL несколько сложнее, чем это было для команд, рассмотренных ранее. Но, надеюсь, что я помогу вам в этом разобраться.

Итак, синтаксис команды MUL такой:

```
MUL ЧИСЛО
```

Выглядит всё очень просто. Однако эта простота обманчива.

Прежде чем разобраться в подробностях работы этой инструкции, давайте посмотрим, что может быть ЧИСЛОМ.

ЧИСЛОМ может быть один из следующих:

- Область памяти (MEM)
- Регистр общего назначения (REG)

Эта команда не работает с сегментными регистрами, а также не работает непосредственно с числами. То есть вот так

```
MUL 200 ; неправильно
```

делать нельзя.

А теперь алгоритм работы команды MUL:

- Если ЧИСЛО - это БАЙТ, то AX = AL * ЧИСЛО
- Если ЧИСЛО - это СЛОВО, то (DX AX) = AX * ЧИСЛО

Вот такая немного сложноватая команда. Хотя сложно это с непривычки. Сейчас мы разберём всё “по косточкам” и всё станет ясно.

Для начала обратите внимание, что инструкция MUL работает либо с регистром AX, либо с регистром AL. То есть перед выполнением этой команды нам надо записать в регистр AX или в регистр AL значение, которое будет участвовать в умножении. Сделать это можно, например, с помощью уже известной нам команды MOV.

Затем мы выполняем умножение, и получаем результат либо в регистр AX (если ЧИСЛО - это байт), либо в пару регистров DX и AX (если ЧИСЛО - это слово). Причём в последнем случае в регистре DX будет старшее слово, а в регистре AX - младшее.

А теперь, чтобы совсем всё стало понятно, разберём пару примеров - с байтом и словом.

Пример умножения в Ассемблере

Итак, например, нам надо умножить 150 на 250. Тогда мы делаем так:

```
MOV AL, 150    ; Первый множитель в регистр AL
MOV BL, 250    ; Второй множитель в регистр BL
MUL BL         ; Теперь AX = 150 * 250 = 37500
```

Обратите внимание, что нам приходится два раза использовать команду MOV, так как команда MUL не работает непосредственно с числами, а только с регистрами общего назначения или с памятью.

После выполнения этого кода в регистре AX будет результат умножения чисел 150 и 250, то есть число 37500 (927C в шестнадцатеричной системе).

Теперь попробуем умножить 10000 на 5000.

```
MOV AX, 10000  ; Первый множитель в регистр AX
MOV BX, 5000   ; Второй множитель в регистр BX
MUL BX         ; Теперь (DX AX) = 10000 * 5000 = 50000000
```

В результате мы получили довольно большое число, которое, конечно, не поместится в слово. Поэтому для результата используются два регистра - DX и AX. В нашем примере в регистре DX, будет число 762 (02FA - в шестнадцатеричной системе), а в регистре AX - число 61568 (F080 - в шестнадцатеричной системе). А если рассматривать их как одно число (двойное слово), где в старшем слове 762, а в младшем - 61568, то это и будет 50000000 (2FAF080 - в шестнадцатеричной системе).

Если не верите - может перевести всё это в двоичное число и проверить.

Теперь о флагах.

После выполнения команды MUL состояния флагов ZF, SF, PF, AF не определены и могут быть любыми.

А если старшая секция результата (регистр AH при умножении байтов или регистр DX при умножении слов) равна нулю, то

CF = OF = 0

Иначе эти флаги либо не равны, либо равны 1.

В конце как обычно расскажу, почему эта команда ассемблера называется **MUL**. Это сокращение от английского слова **MUL**TIPLY, которое можно перевести как “умножить, умножать”.