AVR. Учебный курс. Макроассемблер

AVR. Учебный курс | 5 Июль 2008 | DI HALT | 243 Comments

Перед изучением системы команд микроконтроллера надо бы разобраться в инструментарии. Плох тот плотник который не знает свой топор. Основным инструментом у нас будет компилятор. У компилятора есть свой язык — макроассемблер, с помощью которого жизнь программиста упрощается в разы. Ведь гораздо проще писать и оперировать в голове командами типа MOV Counter,Default_Count вместо

MOV R17,R16 и помнить что у нас R17 значит Counter, а R16 это Default_Count. Все подстановки с человеческого языка на машинный, а также многое другое делается средствами препроцессора компилятора. Его мы сейчас и рассмотрим.

Комментарии в тексте программы начинаются либо знаком «**;**«, либо двойными слешами «**//**«, а еще AVR Studio поддерживает Сишную нотацию комментариев, где коменты ограничены «колючей проволокой» **/* коммент */**.

Оператор .include позволяет подключать в тело твоей программы кусок кода из другого текстового файла. Что позволяет разбить большую исходник на кучу мелких, чтобы не загромождать и не мотать туда сюда огромную портянку кода. Считай куда ты воткнул **.include** туда и вставился кусок кода из другого файла. Если надо подключать не весь файл, а только его часть, то тебе поможет директива **.exit** дойдя до которой компилятор выйдет из файла.

Оператор .def позволяет привязать к любому слову любое значение из ресурсов контроллера — порт или регистр. Например сделал я счетчик, а считаемое значение находится в регистре **R0**, а в качестве регистра-помойки для промежуточных данных я заюзал **R16**. Чтобы не запутаться и помнить, что в каком регистре у меня задумано я присваиваю им через **.def** символические имена.

```
1
2
```

```
.def    schetchik = R0
.def    pomoika = R16
```

И теперь в коде могу смело использовать вместо официального имени **R0** неофицальную кличку **schetchik**
Одному и тому же регистру можно давать кучу имен одновременно и на все он будет честно откликаться.

Также есть оператор **.undef** после которого компилятор напрочь забывает, что данной переменной что либо соответствовало. Иногда бывает удобно. Когда одно и то же символическое имя хочется присвоить разным ресурсам.

```
1
```

```
.undef  pomoika
```

Оператор .equ это присвоение выражения или константы какой либо символической метке.
Например, у меня есть константа которая часто используется. Можно, конечно, каждый раз писать ее в коде, но вдруг окажется, что константа выбрана неверно, а значит придется весь код шерстить и везде править, а если где-нибудь забудешь, то получишь такую махровую багу, что задолбаешься потом ее вылавливать. Так что нафиг, все константы писать надо через **.equ**! Кроме того, можно же присвоить не константу, а целое выражение. Которое при компиляции посчитается препроцессором, а в код пойдет уже исходное значение. Надо только учитывать, что деление тут исключительно целочисленное. С отбрасыванием дробной части, без какого-либо округления, а значит 1/2 = 0, а 5/2 = 2

```
1
2
3
```

```
.equ    Time = 5
.equ    Accelerate = 4
.equ    Half_Speed = (Accelerate*Time)/2
```

Директивы сегментации. Как я уже рассказывал в посте про архитектуру контроллера AVR память контроллера разбита на независимые сегменты — данные (**ОЗУ**), код (**FLASH**), **EEPROM**

Чтобы указать компилятору, что где находится применяют директивы сегментации и адресации.

.CSEG сегмент кода, он же флеш. После этой директивы идет тело программы, комманды процессора. Тут же можно засунуть какие нибудь данные которые не меняются, например таблицу с заранее посчитаными значениями, статичный текст или таблицу символов для знакогенератора.

В сегменте кода уместны директивы:
Адресная метка. Любое слово, не содержащее пробелов и не начинающееся с цифры, главное, чтобы после него стояло **двоеточие**.

```
1
2
3
```

```
.CSEG
label:   LDI      R16,'A'
         RJMP   label
```

В итоге, после компиляции вместо label в код подставится адрес команды перед которой стоит эта самая метка, в данном случае адрес команды LDI R16,’A’
Адресными метками можно адресовать не только код, но и данные, записанные в любом сегменте памяти. Об этом чуть ниже.

.ORG address означет примерно следующее «копать отсюда и до обеда», т.е. до конца памяти. Данный оператор указывает **с какого адреса пойдет собственно программа**. Обычно используется для создания таблицы прерываний.

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
```

```
        .CSEG
        .ORG 0x0000
        RJMP Start  ;перепрыгиваем таблицу векторов.

        .ORG INT0addr   ; External Interrupt0 Vector Address
        RJMP INT0_expection

        .ORG INT1addr   ; External Interrupt1 Vector Address
        RETI

        .ORG OC2addr    ; Output Compare2 Interrupt Vector Address
        RJMP PWM_1

        .ORG OVF2addr   ; Overflow2 Interrupt Vector Address
        RETI
```

```
16
17          .ORG ICP1addr   ;Input Capture1 Interrupt Vector Address
18          RETI
19
20          .ORG 0x0032             ; Начало основной программы
21
22  Start:  LDI R16,0x54    ; и понеслась
```

Статичные данные пихаются в флеш посредством операторов

.db массив байтов.
.dw массив слов — два байта.
.dd массив двойных слов — четыре байта
.dq массив четверных слов — восем байт.

```
1  Constant:       .db     10     ; или 0xAh в шестнадцатеричном коде
2  Message:        .db     "Привет лунатикам"
3  Words:          .dw     10, 11, 12
```

В итоге, во флеше вначале будет лежать число 0A, затем побайтно будут хекскоды символов фразы «привет лунатикам», а дальше **000A, 000B, 000C**.
Последнии числа, хоть сами и невелики, но занимают по два байта каждое, так как обьявлены как **.dw**.

.DSEG сегмент данных, оперативка. Те самые жалкие считанные байты. Сюда не зазорно пихать перменные, делать тут буффера, тут же находится стек.

Тут действует оператор **.BYTE** позволяющий указать на расположение данных в памяти.

```
1
2
```

```
var1:           .BYTE 1
table:          .BYTE 10
```

В первом случае мы указали переменную **var1** состоящую из одного байта.
Во втором случае у нас есть цепочка из 10 байт и переменная **table** указывающая **на первый байт из цепочки**. Адрес остальных вычисляется смещением.
Указывать размеры перменных нужно для того, чтобы компилятор их правильно адресовал и они не налезали друг на друга.

.EESEG сегмент EEPROM, энергонезависимая память. Можно писать, можно считывать, а при пропаже питания данные не повреждаются. Тут действуют те же директивы что и в **flash — db, dw, dd, dq**.

MACRO — оператор макроподстановки. Вот уж реально чумовая вещь. Позволяет **присваивать имена целым кускам кода**, мало того, еще параметры задавать можно.

```
1
2
3
4
```

```
.MACRO SUBI16           ; Start macro definition
        subi @1,low(@0) ; Subtract low byte
        sbci @2,high(@0)        ; Subtract high byte
.ENDM                           ; End macro definition
```

@0, @1, @2 это параметры макроса, они нумеруются тупо по порядку. А при вызове **подставляются в код**.

Вызов выглядит как обычная команда:

```
1	SUBI16  0x1234,r16,r17
```

После имени через запятую передаются параметры, которые подставятся в код.
Макросы позволяют насоздавать себе **удобных команд** на все случаи жизни, по сути создать свой язык. Но надо помнить, что каждый макрос это тупо кусок кода, поэтому если макрос получается большой, то его лучше оформить в виде процедуры или функции — будет резкая экономия места в памяти, но выполняться будет чуток медленней.

Макроассемблер это мощнейшая штука. По ходу пьесы я буду вводить разные макросы и показывать примеры работы макроопределений.

Assembler AVR Макро язык

243 thoughts on “AVR. Учебный курс. Макроассемблер”

Stebanoid

25 Сентябрь 2008 в 4:38

1. Не понял про директиву .ORG Можно поподробнее?
2. В коде
Message: .db «Привет инопланетные придурки
по всей видимости упущена закрывающая кавычка.

3. Не понтно про
var1: .BYTE 1 — зачем это нужно? как можно применить?

4. Самое главное :). Рискуя показаться не в меру наглым, прошу сообщить мне свой e-mail, что бы я вам надоедал всякими вопросами, которые не получается приткнуть ни в одной теме на этом сайте. Просто мне не у кого больше спросить.
Если не хотите выкладывать свой адрес здесь, можно просто написать мне письмо. Мой адрес: мой_ник@gmail.com
Обещаю сильно не надоедать. :)

Stebanoid

25 Сентябрь 2008 в 5:28

Мыло твоё уже нашёл. Буду писать. :)

★ DI HALT

25 Сентябрь 2008 в 13:25

директива ORG говорит компилятору — Копать отсюда и до конца.
Т.е. у тебя память идет последовательно 0..1..2..3..4..5 и так до самого конца. Если ничего не указывать, то компилер команды забьет начиная с нуля. А если сказать, что у нас ORG 5 то уже будут записываться начиная с 5го адреса.

Да, кавычку забыл. Надо будет поправить.

Для того, чтобы у тебя в ОЗУ был зарезервирован байт и был адрес на него. А уж что с этим байтом ты будешь делать тебе решать. Я как переменные использую.

Stebanoid

25 Сентябрь 2008 в 13:37

А… с var1: .BYTE 1 понятно.
Просто в обычном ассемблере x86 я под переменные использовал
Name: .db 10, 156,34
а здесь так делать нельзя, т.к. память раздельная… Упустил этот момент.

---

skadi.exe

8 Январь 2009 в 1:09

отправь куданибудь, где можно толково почитать про обычный ассемблер, т.к. подразумевается, что читатель его уже знает. Желательно попроще, сам 2 курс и знаком только с приплюснутым…

★ DI HALT

8 Январь 2009 в 1:15

Обычный это какой?

---

skadi.exe

30 Январь 2009 в 22:10

Хм, глупость сморозил. На николаев.орг нашел все. Буду разбираться, не шибко страшно вроде. :[

---

skadi.exe

30 Январь 2009 в 22:11

Хм, глупость сморозил. На николаев.орг нашел все. Буду разбираться, не шибко страшно вроде. :[а имелось ввиду основы, регистры и прочее, команды…

---

skadi.exe

30 Январь 2009 в 22:15

WP Ajax Edit Comments — не работает почему-то (пишет «загружаем» и все…

---

ymn

31 Январь 2009 в 13:06

Купил давеча ATmega16-16PU и чёт нигде не могу найти что означает маркировка 16PU…

★ DI HALT

31 Январь 2009 в 14:11

16Мгц максимальная частота. Корпус PDip, коммерческий температурный диапазон (-5…40C)

---

ymn

31 Январь 2009 в 14:18

Благодарю!

---

testicq

10 Февраль 2009 в 2:54

не понятно с директивой .equ. вот кусок кода, который не работает должным образом:

```
;+Установка времени следующего срабатывания Timer0
; Параметры: @0 - число тиков, через которое сработает таймер
; Модифицирует: R16
.MACRO st0of
```

```
outi TCNT0, $FF-@0
.ENDMACRO

.equ XTAL = 8000000 ; Частота МК

.equ t0_divider = 256 ; Предделитель timer0
.equ impulse_time = 1200; ; Время в микросекундах
.equ t0_ticks = (XTAL*impulse_time)/(t0_divider*1000000) ; Время в тиках

st0of t0_ticks ; Задаем время до срабатывания (в тиках)
```

если поставить

```
st0of 37
```

то все работает на ура

★ DI HALT

10 Февраль 2009 в 3:15

Гхм, может слишком много вложенных макросов?

У тебя в макросе St0of макрос outi плюс сложная макроподстановка. Попробуй вынести FF-@0 в отдельный EQU

★ DI HALT

10 Февраль 2009 в 3:17

Да, кстати, результат вычисления какой получается? Целый?

testicq

10 Февраль 2009 в 3:48

неа. дробный, но мне целое число нужно. пофиг как округленное
В принципе задал значение 1152 вместо 1200 чтобы при любых (>=2MHz) частотах XTAL был целый результат (хочу сделать универсально) — все заработало, но вопрос остался — Как макросредствами делать целочисленное деление?

---

testicq

10 Февраль 2009 в 15:33

Странно, но вот такой код работает на ура даже с учетом того, что дробное значение получается

```
.equ    XTAL = 8000000          ; Частота МК

.equ    one_time = 1800;        ; Время проседания линии при логич. "1" (мс.)
.equ    zero_time = 600;        ; Время проседания линии при логич. "0" (мс.)
.equ    check_time = zero_time+(one_time-zero_time)/2;  ; Время проверки переданного бита (мс.)

.equ    t0_divider = 256        ; Предделитель timer0
.equ    t0_check_ticks = XTAL/1000000*check_time/t0_divider ; Время в тиках

outi    TCNT0, Low($FF-t0_check_ticks)
```

★ DI HALT

10 Февраль 2009 в 16:00

ХЗ. В Хелпе к студии есть описание макроязыка. По моему там получается как с типом int — т.е. дробная часть просто теряется.

А ну так тут ты явно привел все к однобайтному результату взяв сразу Low байт и все дела. А там ты пытался напрямую сложить в однобайтный регистр двубайтное (или более) число, вот компилятор и не понял, что ты пытался ему скормить.

Crystaly

12 Август 2013 в 23:02

Дело в том, что AVR Studio компилятор использует 32-разрядные вычисления со знаком. Когда вы «напрягли» компилятор вычислить выражение (XTAL*impulse_time)/(t0_divider*1000000) то получили (8000000*1200)/(256*1000000) в первой скобке 9600000000 не влезает в 32 разряда, и что получилось — знает только компилятор. Вот если бы вы взяли частоту сразу в мегагерцах то есть .equ XTAL = 8 и выкинули ненужный миллион в знаменателе: (XTAL*impulse_time)/t0_divider, то наверняка получили бы верный результат
Совет: можно посмотреть значения, которые вычислил компилятор для .equ и .set, значения меток, адреса переменных и др — в файле ***.map**, который лежит в папке проекта.

---

skywalker

11 Март 2009 в 16:30

Че то я не совсем понял, чем отличаются директивы .def и .equ

---

★ DI HALT

11 Март 2009 в 17:52

equ означает что «Теперь вот это слово равно/эквивалентно этому числу и вместо числа может быть слово»
def означает что «Теперь этот регистр можно обозвать еще и таким словом »

Регистру мы можем присвоить кучу разных имен и это не будет ошибкой (варнинг будет, но на него можно забить)

А вот заэквивалентить одному слову несколько чисел уже нельзя (наоборот, разным словам одно число — запросто, без проблем).

MorskoyZmey

30 Май 2010 в 19:00

т.е. как дефайн в си

.def OPA r16
.def OPA r17

А вот equ, это что-то вроде макро константы

Crystaly

12 Август 2013 в 23:24

На самом деле, есть еще директива .set про которую забыл упомянуть автор))
.def — это присвоение другого имени любому регистру из списка r0…r31
.set — это присвоение любому имени значения выражения. Выражением может быть константа (просто число), имя порта, метка, имя переменной (то же что адрес переменной), значение программного счетчика PC а также их комбинации арифметическими и логическими операциями. Важным свойством имени, заданного с помощью .set — его можно многократно переопределять, то есть это переменная компилятора! Важно понимать, что значение должно быть вычислено на этапе компиляции, а не на этапе выполнения программы.
.equ — это почти как .set, только определяется один раз и переопределить уже нельзя. То есть это константа компилятора!

Ridik911

29 Март 2009 в 0:35

а моджете прикрепить в конце статьи http://www.atmel.ru/Articles/Atmel11.htm для более глубоко ознакомления.. или не требуется?

★ DI HALT

29 Март 2009 в 1:15

В принципе, не проблема. Чуть позже добавлю.

Ridik911

13 Апрель 2009 в 22:45

жду инфу про арифметические (ну и про логические операции) на асм
и если можно то и комбинации арифметических операций например для вычисления синуса (макросом или процедурой\функцией?)
спасибо

★ DI HALT

13 Апрель 2009 в 22:51

Функции в подавляющем большинстве случаев, проще задавать таблицей. Вычислять их — дикий расход памяти и быстродействия.

Ridik911

14 Апрель 2009 в 21:23

Вычислять их — дикий расход памяти и быстродействия.
—-
ну да.. в ряд тейлора (циклом) до необходимой точность.. это мучительно для проца и памяти МК

задавать таблицей в томто и дело и неподходит…
ладно раскужу что хочу

хочеца сделать чтото типа мидиконтроллер — тоесть контроллер генерирующий звук при подключенной к нему некоторой клавиатуре… -но т.к. миди то нужно хранить семплы гдето например в eeprom- но памяти этой в МК будет очень мало (надо «допаивать» 24xx)
тогда яже свел задачу к генерации.. формулами — это экономит ROM память но предпожил что процессор потянет вычисления.. я

так понял из ваших слов непотянет..
да и самое сложное в моей идее — это подбор формул легче семплами

допустим выбрал метод семплы.. а вот нет идеи как их ээ микшировать\сумировать(когда нажаты две и более кнопки на клавиатуре)? (повидимум програмно но как.. прирывания — нет имхо, акакже)

★ DI HALT

14 Апрель 2009 в 21:44

Ну почему не потянет — потянет, только сожрет много ПЗУ.

А что из себя представляет сэмпл?

Ridik911

15 Апрель 2009 в 0:53

впринципе как ты и сказал «проще задавать таблицей»
впринципе семпл wav файл (а в реале миди семпл я незнаю, но возможно также)
тоесть мидиконтроллером управляем семплами: изменить скорость воспроизведения(нота), продолжительность, громкость… и т.д.
но семплы не мало занимают места как уже понятно…
например если делать семплы wav то каждый будет весить гдето по 4-8КБ

только сожрет много ПЗУ.
—-
может ОЗУ?

кстати очееень жду статейку про подключение к МК озу
для тех МК в которых это поддерживается (типа мега8515) и для тех кто не поддерживает но можно(если можно) реализовать

програмно
а также про dram (валяется планка сим.. думаю какбы примостырить к МК)

Ridik911

15 Апрель 2009 в 1:02

впринципе как ты и сказал «проще задавать таблицей»
—-
так и называют wavetable

Agrin

15 Май 2009 в 1:27

Можно использовать синтезаторы типа YM3812 или аналогичные. Синтезировать качественно многоканальный звук из сэмплов или функционально с помощью 8разрядного МК задача если и решимая, то с качеством звука как из китайских говорящих кукол.

RMiSe

13 Май 2009 в 14:07

Здравствуйте! DI HALT, не могли ли вы написать (может быть макрос) как из переменной table (table:.BYTE 10) считать побайтно символы?
Заранее благодарен за ответ!

★ DI HALT

13 Май 2009 в 16:29

ПОбайтно? Хм.

Проще вот так:
LDS R16,Table ; первый байт
LDS R16,Table+1 ; второй байт ну и так далее.

Если надо программно выбирать значение, то тут есть другой подход.

LDI XL,low(table)
LDI XH,High(table)

LD R16,X
; считать первый байт или

LD R16,X+
;считать первый байт и после увеличить X на 1

Ну или вручную складывать регистровую пару X с константой-смещением. Вариантов тьма. Что что, а инструментарий работы с ОЗУ у AVR очень богатый.

VladislavZ

22 Октябрь 2009 в 1:13

А можно по подробней об ОПЕРАТОРАХ .equ .def !? К примеру, возможно ли присвоить имя не регистру, а скажем порту или даже лучше конкретному пину порта!? Возможно, ли определенным ИМЕНАМ присвоить значение?!
Очень хочется статейки по различным интерфейсам, и самое главное про библиотеки, и по подробней!

★ DI HALT

22 Октябрь 2009 в 3:37

можно.

def работает ТОЛЬКО с регистрами. А вот имена портов это не более чем equ под имя порта, все они прописаны в inc файле соответствующего мк.

вот у меня обычно такая ботва:
.def COK = R26
.def SEDOK = R27
.def Catch = R28
.def Armed = R29

.equ R_ON = 6
.equ R_ON_P = PORTB
.equ R_ON_D = DDRB

.equ R_EN = 7
.equ R_EN_P = PORTB
.equ R_EN_D = DDRB

VladislavZ

22 Октябрь 2009 в 5:16

То есть получается (.equ R_ON = 6) R_ON 6-ой пин неизвестного пора ; (.equ R_ON_P = PORTB) R_ON_P все пины портаB. А совместить не как не получится? К примеру, я хочу управлять 4 пином ddrC, 5 пином ddrC по отдельности и каждому присвоить свое имя, допустим scl ddrC,4; sda ddrC,5 и потом уже в коде так

Sbi sda
Sbi scl
Cbi sda
Sbrc sda

★ DI HALT

22 Октябрь 2009 в 5:23

Не, не получится.

Только SBI IIC_PORT,SDA

Но никто не мешает сделать тебе макрос, например:

.MACRO SBI_SDA
SBI IIC_PORT,SDA
.ENDM

И юзать в своей программе
SBI_SDA как одну команду. Тока макросы тогда уже удобней звать, например так:

SDA_H
SDA_L

---

neon15

26 Февраль 2012 в 16:19

В основном тексте статьи:

Оператор .def позволяет привязать к любому слову любое значение из ресурсов контроллера — порт или регистр.

Нужно поправить на «ТОЛЬКО» с регистрами.

Alex_Megavolt_79

21 Ноябрь 2009 в 2:27

Di Halt, я не понял как кусок проги или всю её сделать как макрос и юзать. Например вот я не давно изучал динамическую индикацию, вот как всю её или часть забахать макрос и как задавать параметры то есть заюзать:

```
; Выполняемые функции: Динамическая индикация
;_____________________________________________________________________

.device ATtiny2313
.nolist
.include «d:\avr\def\tn2313def.inc»
.list

;=====================================================================
; Объявления:

.def temp1=r1
.def temp2=r2
.def n_digit=r3
.def counter=r4
.def digit_out=r5
.def n_point=r6
.def temp=r16

.equ crystal=6000000 ;частота кварца
.equ Sciler=1024 ;предделитель таймера
.equ N_TC0_2500uc=255-crystal/(Sciler*400);число тиков таймера TC0 для организации
;задержи при динамической индикации
```

```
.equ amount_digit=4 ;маска, нужно ввести количество разрядов индикатора
;в данном случае 4-х разрядный индикатор

;=====================================================================
; Начало программы:

.dseg

Digit: .byte 4
Point: .byte 1

.cseg
.org 0
rjmp RESET ;Reset Handler
nop ;rjmp INT0 ;External Interrupt0
nop ;rjmp INT1 ;External Interrupt1
nop ;rjmp ICP1 ;Input capture interrupt 1
nop ;rjmp OC1A ;Timer/Counter1 Compare Match A
nop ;rjmp OVF1 ;Overflow1 Interrupt
rjmp OVF0 ;Overflow0 Interrupt
nop ;rjmp URXC0 ;USART0 RX Complete Interrupt
nop ;rjmp UDRE0 ;USART0 Data Register Empty Interrupt
nop ;rjmp UTXC0 ;USART0 TX Complete Interrupt
nop ;rjmp ACI ;Analog Comparator Interrupt
nop ;rjmp PCINT ;Pin Change Interrupt
nop ;rjmp OC1B ;Timer/Counter1 Compare Match B
nop ;rjmp OC0A ;Timer/Counter0 Compare Match A
nop ;rjmp OC0B ;Timer/Counter0 Compare Match B
nop ;rjmp USI_START ;USI start interrupt
```

```
nop ;rjmp USI_OVF ;USI overflow interrupt
nop ;rjmp ERDY ;EEPROM write complete
nop ;rjmp WDT ;Watchdog Timer Interrupt
; for compatibility purpose

;=====================================================================
; Инициализация

RESET:
ldi temp,low (RAMEND) ;Определение начала стека
out SPL,temp

ldi temp,0b0001111 ;Конфигурирование портов
out DDRD,temp ;Порт D вход\выход
ser temp
out DDRB,temp ;Порт B выход
out PIND,temp ;Команда для отладчика
out PORTD,temp ;Включаем подтягивающие резисторы
out PORTB,temp ;Выключаем матрицу

ldi temp,15 ;Инициализация сторожевого таймера
out WDTCR,temp
rcall TNCT_0 ;Инициализация таймера TC0

;=====================================================================
; для проверки работы индикатора выводит на индикатор число 4321
; запятая во втором знакоместе
ldi Temp,4
sts Digit ,Temp ;загрузка начальных значений число 4321
```

```
ldi Temp,3
sts Digit+1,Temp
ldi Temp,2
sts Digit+2,Temp
ldi Temp,1
sts Digit+3,Temp

ldi Temp,1
sts Point,Temp ;загрузка номера разряда запятой 2-й разряд
clr counter

;=====================================================================
; Основной блок программы

main:

wdr ;»Бросим кость собаке»

rjmp main ;Начинаем всё сначала

;=====================================================================
; Инициализация таймеров

TNCT_0:
cli ;запрещаем прерывания
ldi temp,N_TC0_2500uc ; инициализируем таймер T\C0 на T=0,0025 c
out TCNT0,temp
ldi temp,0b00000000
out TCCR0A,temp
```

```
ldi temp,(1<<CS02)+(1<<CS00) ;прескайлер таймера 1024
;ldi temp,(1<<CS00) ;прескайлер таймера 1 для отладки
out TCCR0B,temp
ldi temp,(1<<TOIE0) ;прерывания T\C0 по переполнению
out TIMSK,temp
sei
reti

;=====================================================================
; Обработчик прерываний

OVF0:
rcall TNCT_0 ;Инициализация таймера TC0
ldi ZL,Low(Digit) ;инициализация массива
ldi ZH,High(Digit)
clr temp2
add ZL,counter ;выбираем нужный номер ячейки
adc ZH,temp2
ld digit_out,Z ;загружаем ячейку
lds n_point,Point
rcall Decoder_n_digit
rcall Decoder
out PORTD,n_digit
out PORTB,digit_out
inc counter ;увеличиваем номер разряда
mov temp,counter
andi temp,(amount_digit-1) ;отсекаем количество разрядов по маске
mov counter,temp
reti
```

```
;=====================================================================
; Подпрограммы
Decoder:
;преобразует десятичную цифру разряда в 7-ми сегментный код индикатора

clr temp
ldi ZL,Low(DcMatrix*2) ;инициализация массива перекодировки
ldi ZH,High(DcMatrix*2)
clr Temp2 ;прибавление переменной
add ZL,digit_out ;к 0-му адресу массива
adc ZH,Temp2
lpm ;загрузка значения 7-ми сегментного кода индикатора
mov digit_out,r0
clr temp
cp n_point,temp ;проверяем номер разряда зяпятой если
breq PC+6 ;если он равен 0 то выходим из подрограммы
cp counter,n_point ;если он не равен нулю сравниваем с
brne PC+4 ;номером разряда вкл в данный момент
mov temp,digit_out
andi temp,0b01111111 ;если равен зажигаем зяпятую иначе
mov digit_out,temp
ret ;выходим из подпрограммы

DcMatrix:
;массив — таблица истинности декодера
; hgfedcba hgfedcba
.db 0b11000000,0b11111001 ;0;1
.db 0b10100100,0b10110000 ;2;3
.db 0b10011001,0b10010010 ;4;5
```

```
.db 0b10000010,0b11111000 ;6;7
.db 0b10000000,0b10010000 ;8;9

Decoder_n_digit:
ldi ZL,Low(DnMatrix*2) ;инициализация массива
ldi ZH,High(DnMatrix*2)
clr Temp2 ;прибавление переменной
add ZL,counter ;к 0-му адресу массива
adc ZH,Temp2
lpm ;загрузка значения
mov n_digit,r0
ret

DnMatrix:
;массив — таблица истинности декодера разряда

.db 0b11111110,0b11111101 ;0;1
.db 0b11111011,0b11110111 ;2;3
```

★ DI HALT

21 Ноябрь 2009 в 11:43

Всю прогу макросом нельзя. Да и ни к чему это.

А небольшие кусочки — прочитай статью еще раз.

Flint

2 Декабрь 2009 в 23:18

Вопрос по .DSEG
Если взять структуру исходных файлов, как, например, в прогах ADCsoftFilter, UARTundSoftADC, в каком месте нужно её указывать, или это все равно — компилятор сам разберётся?
«var1: .BYTE 1» А как задать 2-х байтную переменную в ОЗУ, var1: .BYTE так?
И отсюда же следующий вопрос — как задать массив 2-х байтных чисел в ОЗУ. Может быть надо уазать var1: .BYTE 8 — это массив из 4-х двухбайтовых значений, правильно? Ну и соответственно, работать с ними, подразумевая, что каждый следующий элемент идёт через 2 байта.

★ DI HALT

2 Декабрь 2009 в 23:38

DSEG везде где не CSEG :) Я обычно перед CSEG ставлю. Сразу после инклюдников обычно.

вроде бы .WORD можно определить, не помню уже. ПОзырь в описаниях макроязыка.

Flint

2 Декабрь 2009 в 23:52

Собственно, я поставил в vectors.asm, но тоже получается перед .CSEG. Надо мне будет, наверное, завести отдельный фалик только с переменными ОЗУ и записать их все туда :) И подключить include'ом в нужном месте

Flint

2 Декабрь 2009 в 23:54

Кстати, в данном конкретном примере «Сразу после инклюдников» получится уже после .CSEG. Ну это так, к слову.

Flint

2 Декабрь 2009 в 23:25

А как задать 2-х байтную переменную в ОЗУ, var1: .BYTE 2 — опечатался
А запись var1: .BYTE 8 я имею ввиду рассматривать как «массив из 4-х двухбайтовых значений», хотя ассемблеру, возможно, мои мысли и неведомы и он подразумевает просто последовательность данных в ОЗУ

★ DI HALT

2 Декабрь 2009 в 23:45

А нет, вру. это не из той оперы. Тут только отожрать нужное количество байт. ВСе равно ты не можешь их инициализировать на уровне компилятора. Так что пофиг чем ты их задаешь, также тебе адреса придется вручную считать для каждого элемента массива.

Flint

2 Декабрь 2009 в 23:59

Инициализировать-то потом не проблема. Главное правильно мне их задать. А «адреса придется вручную считать» это в смысле подразумевается 1-и 2-х байтовый элемент — это адрес 0-го + 2 ? Ну это тоже не очень большая проблема. Ведь в программе же я оперирую символьным имененм, и следующий элемент массива находится по вышеприведённому способу исходя из размера элемента массива.

---

tarasadidas

9 Декабрь 2009 в 12:36

И опять вопрос по оператору (.BYTE) .
Если мы выделили массив с меткой Variable:
как установить указатель допустим Z на метку массива

ldi Zh,high(Variable)
ldi Zh,low(Variable)
а считывать массив
st Z+,r16

st Z+,r17
и т.д.
Или я ошибаюсь?

★ DI HALT

9 Декабрь 2009 в 12:39

Именно так, только

ldi ZH,high(Variable)
ldi ZL,low(Variable)

А чтение это не st (это сохранение), а ld

---

tarasadidas

9 Декабрь 2009 в 14:32

DI HALT Спасибо ошибки замелил сам но отредактировать не смог не поддерживается у вас чтоли.
Но самое главное суть понял а позже буду внимательней уж извини.

---

tarasadidas

11 Декабрь 2009 в 12:33

И еще вопрос по .include .Вставлять я так понял этот оператор мона где угодно хоть посреди кода а помещать в прикрепляемый файл как куски того же кода так и подпрограммы (даже обработчики прерывания).
И еще расширение у файла должно быть асм. или произвольное и помещать его нуна в папку текущего проекта?
Я так понимаю что любой отлаженный кусок кода (инициализация портов) или подпрограмму просто вырезаем и сохраняем как отдельный файл с расширением асм. в папке проекта
а в том месте пишем .include имя файла.asm или я не прав?

★ DI HALT

11 Декабрь 2009 в 14:46

Да. Расишерие может быть любым. Главное чтобы это был текстовый файл. Быть он может тоже где угодно, но длинные пути и пути с русскими буквами плохо отрабатываются.

tarasadidas

11 Декабрь 2009 в 17:48

Проверил расширение только asm катит.Иначе студия матерится.

Spok

2 Февраль 2010 в 13:35

Я могу ошибаться, но в статье косяк: надо не .EESEG, а .ESEG.

Flint

17 Февраль 2010 в 21:28

Не пойму, проект на основе Atmega48, при компиляции запнулось повидимому на макросе OUTI с такой ошибкой: macro.asm(6): error: Operand 1 out of range: 0xc4 И указывает на вторую строчку макроса, т.е. OUT @0,R16. И еще куча подобных ошибок, где этот макрос встречается. Мой проект http://slil.ru/28665386

★ DI HALT

18 Февраль 2010 в 5:00

Да, все верно. Открывай файл m48def.inc и смотри карту ио регистров. Те из них что помечены как memory mapped недоступны для команды OUT|IN которая используется в макросе OUTI. Загрузка/выгрузка в них идет как в ячейку памяти по командам STS/LDS

т.е. меняем OUT на STS и все работает.
Сделай еще один макрос OUTIm такого вида и юзай его для мемори маппед регистров.

Crystaly

12 Август 2013 в 12:36

Вот универсальные макросы — псевдо-команды для ввода и вывода. Команду nop можно выкинуть, она вставлена для того, чтобы независимо от типа контроллера команда всегда выполнялась одинаковое количество тактов (2 такта).

```
;-----------------------------------------------------------
;ВВОД ИЗ ПОРТА
;-----------------------------------------------------------
.MACRO INP ; reg , in_port
.if @1 > 255
.set in_port = BYTE2(@1)
.else
.set in_port = @1
.endif
.if in_port 255
.set out_port = BYTE3(@0)
.else
.set out_port = @0
.endif
.if out_port <= 0x3F
out out_port,@1
nop
.else
sts out_port,@1
```

```
.endif
.ENDMACRO
```

Crystaly

12 Август 2013 в 13:16

что-то выше не то вставилось, повторяю:
Вот универсальные макросы — псевдо-команды для ввода и вывода. Команду nop можно выкинуть, она вставлена для того, чтобы независимо от типа контроллера команда всегда выполнялась одинаковое количество тактов (2 такта).
Обратите внимание, как эффективно используется директива .set для переименования параметров макроса.

```
;-----------------------------------------------------
;ВВОД ИЗ ПОРТА
;-----------------------------------------------------
. ; @0 @1
. .MACRO INP ; reg , in_port
. .set reg = @0
. .set in_port = @1
. .if in_port <= 0x3F
. in reg,in_port
. nop
. .else
. lds reg,in_port
. .endif
. .ENDMACRO

;-----------------------------------------------------
;ВЫВОД В ПОРТ
;-----------------------------------------------------
```

```
. ; @0 @1
. .MACRO OUTP ; out_port , reg
. .set out_port = @0
. .set reg = @1
. .if out_port <= 0x3F
. out out_port,reg
. nop
. .else
. sts out_port,reg
. .endif
. .ENDMACRO
```

ZhekSooN

21 Март 2010 в 19:01

Опа, никто не упомянул о таких дьявольских штуках, как .if\.endif, .ifdef\.endif :)
С их помощью можно создать почти свой язык программирования :)

★ DI HALT

21 Март 2010 в 19:20

Точно. Совсем про них забыл. Надо добавить. Кстати, а AVRassebler1 их поддерживает? В доке про них нет ни слова.

ZhekSooN

22 Март 2010 в 20:52

Все поддерживает, искать надо лучше ;) Help — AVR Tool User guide — AVR assembler — User's guide — Directives. сам ведь не знал о их оффициальном существовании! O_o

★ DI HALT

22 Март 2010 в 22:48

Охрененно. Вот жеж надо было закопать O_o а я ведь там два раза ходил и не заметил.

---

auara

24 Апрель 2010 в 4:26

Общие макросы переношу из проекта в проект и подключаю через:
.include «MACROS.MAC»

(кстати, кто-то жаловался что только расширение «.asm»)

Вот примерчик, работающий на разных AVR8

```
;*****************************************************************************
.macro  mEORBi          ;XOR Bit in register I/O
.IF @0<64
        IN      rM,@0
        EOR     rM,@1
        OUT     @0,rM
.ELSE
        LDS     rM,@0
        EOR     rM,@1
        STS     @0,rM
.ENDIF
.endmacro
;;--------------------  mEORBi  regIO,reg
;;--------- Ex: ------  mEORBi  TIFR,R17
```

ZhekSooN

22 Март 2010 в 18:17

Насчет первой версии не знаю, но во второй всё охрененно пашет. Я рад: создал скрипты инициализации таймеров, USART, ADC и других нудных вещей — и не парится!
Это как Сишная вставка посреди ассемблерного поля :)

A_ndrej

22 Март 2010 в 23:11

Добрый день.
Подскажите пожалуЙста, можно ли в макросе, в качестве параметра»@» ставить ссылку на подпрограму?
Например:
,MACRO CALC_1
ldi @1, @2
cp r16, @2
BREQ @3

И тогда при вызове макроса написать:
CALC_1 r17,r18,METKA
где «METKA» будет являтся той ссылкой, на которую с макроса пойдет контроллер.

★ DI HALT

22 Март 2010 в 23:17

Думаю проблем не составит. Макропроцессору то без проблем что подсунуть.

A_ndrej

22 Март 2010 в 23:36

Большое спасибо.

dima_m

26 Март 2010 в 0:13

Сейчас пробовал макрос забабахать. Супер, все работает. Реально удобно. Уже начал использовать в деле. Только вот хочется узнать, откуда у DIHALTA слева в редакторе появился столбец с цифрами, то есть нумерация строк. У меня в AVR STUDIO нет такой фишки. Не подскажете кто, как такую себе сделать?

★ DI HALT

26 Март 2010 в 7:06

У меня тоже нет, а с чего ты взял, что у меня в ABP студио есть нумерация?

dima_m

26 Март 2010 в 11:23

Ну вот, когда ты вверху показываешь пример кода. У тебя в редакторе слева есть нумерация строк. Да и сам редактор привлекательный. Попривлекательнее чем в AVR. Просто думал ты какой нибудь плагин доставил к проге. Значит это не AVR STUDIO. А что же это за редактор?

★ DI HALT

26 Март 2010 в 12:02

Это WordPress syntax hilighter — плагин к движку сайта который сам код подкрашивает.

dima_m

26 Март 2010 в 13:39

OK! Понятно.

---

auara

24 Апрель 2010 в 4:52

Мы тоже хотим красиво в коментах писать,
а то сваливается все в кучу….
И можно-ли редактировать свое сообщение?

---

bdp

23 Апрель 2010 в 1:46

DI HALT в придложении
>В итоге, во флеше вначале будет лежать число 0A, затем побайтно будут хекскоды символов
>фразы “привет лунатикам”, а дальше 000A, 000B, 000F.
>Последнии числа, хоть сами и невелики, но занимают по два байта каждое, так как .dw.
несостыковка с примером идущим сверху в котором нет описи «dw».

Это при условии что я ничего не пропустил :)

★ DI HALT

23 Апрель 2010 в 1:48

ой точно куда то кусок примера отвалился :) Спасибо щас поправлю

---

auara

24 Апрель 2010 в 20:54

пока не видно

---

auara

24 Апрель 2010 в 4:45

Еще упущена «.SET» (про «.IF», «.IFDEF», «.IFNDEF» уже вспоминалось)

Вот кусок из программы:

```
|.set uD=$00
|.set uB=$00
|;————————
|.IF cInpMax >= $0C
| .IFNDEF cInp1_Port or cInp1_Pin
| .ERROR “Not defined cInp1_Port or cInp1_Pin”
| .ENDIF
| .set U=$FF-(1<<cInp1_Pin)
| .if cInp1_Port == PortD
| .db uD&U, U
| .db uB, $FF
| .else
| .db uD, $FF
| .db uB&U, U
| .endif
|.ENDIF
|
```

auara

24 Апрель 2010 в 5:27

ну Блин!
Уже и вертикальную черточку поставил, чтоб не сплюскивало, так автоматика сайта пробелы сократила. Компьютер и так прохавает, а я людям для наглядности форматировал. Для восприятия лучше…
— — — — — — — — — — — —
Может программка есть какая?
Написал в ней, отформатировал, сюда вставил и все красиво.
А если нету — то надо на сайте что-то менять.
Ну напрягает… тексты программ в комментариях ко всем статьям — просто каша.
я то разберусь, но начинающие могут упустить(не понять) много полезного.

auara

24 Апрель 2010 в 5:55

Глянул на местный форум — ситуация та же… код-лист == каша.
Даааа уж… Видимо не все так просто.
Удивительно! Обсуждения концепций OS и каменный век — рядом!

★ DI HALT

24 Апрель 2010 в 10:45

Я не хтмл программист. А готовый плагин который бы мог это делать я не нашел.

auara

25 Апрель 2010 в 3:40

Спасибо, вижу ты облагородил сообщение «апреля 24, 2010 at 4:26»
Еще и в «24 Апр 2010 4:45» желательно убрать передующие «|» и сдвиги добавить. Плиз…

★ DI HALT

25 Апрель 2010 в 9:30

лень

auara

25 Апрель 2010 в 3:41

Я думал ты сидишь на готовом движке сайта. А если все сам…
Ну может кто поможет/подскажет DI HALT-у доработать/переработать обработку сообщений на сайте, а то и сайт перемастырить.
Ему над уроками надо мозговать, а он тратит время на редактирование наших текстов.

★ DI HALT

25 Апрель 2010 в 9:27

На готовом и сижу. Потому и не могу взять и что то там подкрутить если этого нет в настройках.

auara

25 Апрель 2010 в 4:11

После установки AvrStudio4, запустить файлик помощи:
AVRTools\Help\AVRASM.chm
Там в закладке «Содержание» выбираем (раскрываем)
-> «AVR Assembler» -> «User’s Guide» -> «Directives»

И вот здесь, в разделе «Assembler directives», внимательно рассматриваем/изучаем директивы и примерчики (как раз по нашему учебному курсу).

---

dima_m

8 Июнь 2010 в 12:04

DI HALT привет. А что означает выражение: макрос — лучше оформить в виде процедуры или функции? У тебя на сайте это где-то объясняется? Я бы почитал.

★ DI HALT

8 Июнь 2010 в 12:25

Имеется в виду, что макрос в этом случае использовать не стоит, а лучше сразу написать процедуру и вставить ее в код.

---

skywalker

13 Июль 2010 в 19:33

Constant: .db 10 ; или 0xAh в шестнадцатеричном коде
Message: .db «Привет лунатикам»
Words: .dw 10, 11, 12

В итоге, во флеше вначале будет лежать число 0A, затем побайтно будут хекскоды символов фразы “привет лунатикам”, а дальше 000A, 000B, 000F.

А разве последняя строчка не равна 000A,000B,000C?

★ DI HALT

13 Июль 2010 в 19:36

ОПечатка, конечно же 000с

---

dima_m

24 Октябрь 2010 в 15:36

1. Получается каждая буква фразы .db «Привет лунатикам» займет один байт во флеш памяти по порядку?
2. И соответственно вместо каждой буквы будет вбит ее ASII символ?
3. Как бы это применить? Можно ли насоздавать таких сообщений которые нужно вывести на LCD дисплей, потом как нибудь вызывать эти метки?

★ DI HALT

24 Октябрь 2010 в 17:15

1. Да
2. Да
3. Так и применяют, вписывая текстовые строки во флеш. О том как вызываются текстовые метки из памяти программ написано далее в курсе. Где то про работу с памятью через команды LPM

---

dima_m

24 Октябрь 2010 в 19:07

Ты имееш в виду это?

```
LDI ZL,low(data*2) ; заносим младший байт адреса, в регистровую пару Z
LDI ZH,high(data*2) ; заносим старший байт адреса, в регистровую пару Z
; умножение на два тут из-за того, что адрес указан в
; в двубайтных словах, а нам надо в байтах.
; Поэтому и умножаем на два
; После загрузки адреса можно загружать число из памяти
```

LPM R16, Z ; в регистре R16 после этой команды будет число 12,
; взятое из памяти программ.

; где то в конце программы, но в сегменте .CSEG
data: .db 12,34,45,23

★ DI HALT

24 Октябрь 2010 в 19:15

Оно самое

dima_m

24 Октябрь 2010 в 20:16

Супер, осталось только замутить программу делающую это.

henvert

21 Ноябрь 2010 в 1:06

Подскажите пожалуйста где можно найти полное описание макроассемблера для AVR.
наткнулся на такой кусочек кода

; тактовая частота МГц
#define XCLK 9.6
.equ CN2MKS = INT((2*XCLK-7)/3) ; константа 2мкс
.equ CN5MKS = INT((5*XCLK-9)/3) ; константа 5мкс
.equ CN10MKS = INT((10*XCLK-9)/3) ; константа 10мкс
.equ CN50MKS = INT((50*XCLK-9)/3) ; константа 50мкс

А что такое INT() не понятно((

★ DI HALT

21 Ноябрь 2010 в 16:36

В хелпе к AVR Studio есть полное описание.

salpingot

7 Январь 2011 в 2:27

Видел я это полное описание не сильно густо там. Нам нравится когда Вы объясняете.

salpingot

7 Январь 2011 в 2:20

Я так и не разобрался с макросом, (хотя уже успел полистать и книжку revich-yuriy, да и в инете покапашился), но все равно пока нет полного списка операторов и их описания для ASM&AVR. Единственгное что понял точно, что макрос может иметь до 10 параметров, к которым в его теле обращаются через @0,@1,@2 и т.д. до @9, и что макрос это повторяющийся кусок кода который при трансляции просто тупо заменяет метки в тексте программы на «тело» макроса, т.е. экономии в размере самой программы нет. Применение макросредств целесообразно когда критично время выполнениия программы, а если нужна экономия памяти, тогда нужно использовать процедуру.

Взял вот отсюда: http://www.asm-faq.ru/direktivy-i-operatory-assemblera/69-makrooperatory.html

Макрооператоры
Макрооператор & (амперсанд) нужен для того, чтобы параметр, переданный в качестве операнда макроопределению или блоку повторений, заменялся значением до обработки строки ассемблером. Так, например, следующий макрос выполнит команду PUSH EAX, если его вызвать как PUSHREG A:

```
pushreg macro letter
push e&letter&x
endm
```

Иногда можно использовать только один амперсанд — в начале параметра, если не возникает неоднозначностей. Например, если передается номер, а требуется создать набор переменных с именами, оканчивающимися этим номером:

```
irp number,
msg&number db ?
endm
```

Макрооператор (угловые скобки) действует так, что весь текст, заключенный в эти скобки, рассматривается как текстовая строка, даже если он содержит пробелы или другие разделители. Как мы уже видели, этот макрооператор используется при передаче текстовых строк в качестве параметров для макросов. Другое частое применение угловых скобок — передача списка параметров вложенному макроопределению или блоку повторений.

Макрооператор ! (восклицательный знак) используется аналогично угловым скобкам, но действует только на один следующий символ, так что, если этот символ — запятая или угловая скобка, он все равно будет передан макросу как часть параметра.

Макрооператор % (процент) указывает, что находящийся за ним текст является выражением и должен быть вычислен. Обычно это требуется для того, чтобы передавать в качестве параметра в макрос не само выражение, а его результат.

Макрооператор ;; (две точки с запятой) — начало макрокомментария. В отличие от обычных комментариев текст макрокомментария не попадает в листинг и в текст программы при подстановке макроса. Это сэкономит память при ассемблировании программы с большим количеством макроопределений.

ВОПРОС: ЧТО ТАКОЕ ПАРАМЕТРЫ И КАК ЭТИМ ПОЛЬЗОВАТЬСЯ С ПРИМЕРАМИ ПОПРОЩЕ ТАК ЧТО БЫ МОЖНО БЫЛО БЫ УЛОВИТЬ СУТЬ, И СОСТАВТЕ ПОЖАЛУЙСТА СПИСОК МАКРООПЕРАТОРОВ И ОПИСАНИЕ К НИМ С ПРИМЕРАМИ.
с уважением Salpingot.

salpingot

7 Январь 2011 в 2:29

Я в курсе что это для x386 tasm masm и т.д.

★ DI HALT

7 Январь 2011 в 4:11

Прикол в том, что AVRASM не умеет и 5% того, что умел тасм. Так что тут почти все что он умеет.

Примеры попроще я привести не могу. Куда уже проще?

salpingot

8 Январь 2011 в 3:02

Уважаемый Ди Халт, а можно описать каждую директиву предпроцессора, (т.е. для чего она нужна, где не верно внесите исправления пожалуйста, некоторые без перевода не знал как перевести):

Preprocessor directives, КАК Я ПОНЯЛ ИХ ВСЕГО 14.
1. #define
2. #elif
3. #else, ИНАЧЕ, ТОГДА, лучше всего смотрится с комбинацией if (если, тогда/иначе)
4. #endif
5. #error, при трансляции вывод ОШИБКИ,
6. #if , условие ЕСЛИ,
7. #ifdef ЕСЛИ_СИМВОЛИЧЕСКОЕ ИМЯ ???
8. #ifndef ОТМЕНИТЬ ЕСЛИ_СИМВОЛИЧЕСКОЕ ИМЯ ???
9. #include, ВСТАВКА ФАЙЛА с другими макросами, подпрограммами (процедурами) и т.п.
10. #message в ходе трансляции вывод СОБЩЕНИЯ,

11. #pragma
12. #undef ОТМЕНИТЬ СИМВОЛИЧЕСКОЕ ИМЯ
13. #warning, ПРЕДУПРЕЖДЕНИЕ, типа смотри сюды и будь внимателен.
14. # (empty directive) ЗНАЧОК ПУСТОЙ ДИРЕКТИВЫ

Я правильно понимаю, что есть только ТРИ макрооператора: '@' и "." '#' при составлении макросов???
Идеальный вариант с примерам на каждый макрооператор (можно один большой).
ДА И ДЛЯ НОВИЧКОВ ТАКИХ КАК Я, МОЖЕТ ПРИГОДИТЬСЯ Справка по Ассемблеру для Atmel AVR
1. http://dfe3300.karelia.ru/koi/posob/avrlab/avrasm-rus.htm
2. http://mymcu.ru/Articles/Atmel11.htm
это понадобиться для того что бы было понятно дальше о чем пишет Ди Халт.

---

salpingot

9 Январь 2011 в 13:22

а вот пример и попроще как использовать макрос для включения ноги мк D7 на пин боард, т.е. в коде в том месте где нужно, можно просто написать имя макроса (у меня включение ноги D7 называется noga_mk_D7), как и говорил выше Ди Халт, с помошью макроязыка можно существенно оптимизировать программирование на асемблере.

```
1
2
3
4
5
6
7
8
9
10
```

```
begin:          nop;
                        nop;
.macro          noga_mk_D7;
                        ldi             r17,0b11111111;
                        out             DDRD,r17;
                        ldi             r16,0b10000000;
                        out             PORTD,r16;
.endm
                        nop;
                        nop;
```

```
11                          noga_mk_D7;
12                          nop;
13                          nop;
14          zamk_cikl:
15                          nop;
16                          nop;
17                          jmp             zamk_cikl;
```

★ DI HALT

9 Январь 2011 в 17:39

Пример крайне неудачный. Т.к. планируется включить вывод 7, а по факту меняем содержимое всего порта и воздействуем на выводы 0..6

Если уж делать то так:

.macro noga_mk_D7
SBI DDRD,7
SBI PORTD,7
.endm

AlexRom

9 Февраль 2011 в 18:55

Думаю, данный кусок кода можно просто через метку сделать, на мой взгляд так будет чуть проще.

★ DI HALT

9 Февраль 2011 в 19:09

В смысле через метку? В подпрограмму вынести? Не проще. ПРидется делать вызовы CALL и RET, что сожрет всю экономию.

AlexRom

10 Февраль 2011 в 16:49

Нет, не в подпрограмму. salpingot предложил как макросом включить ногу, как бы для существенной оптимизации. Спасибо, за данный пример, с него я и понял, что такое макрос. Но для простого включения ноги макрос… Вот, я и подумал, что может лучше через метку

rjmp noga_mk_D7;
noga_mk_D7:
SBI DDRD,7
SBI PORTD,7
Может я и ошибаюсь.
P.S. Огромное спасибо DI HALT, помощь огромная благодаря этому сайту!

★ DI HALT

10 Февраль 2011 в 16:54

Ну да, сделаешь ты джамп, а как вернешься обратно?

AlexRom

9 Февраль 2011 в 18:52

salpingot благодарю за ссылки ;)

maxgrind

24 Февраль 2011 в 13:56

а ячейки озу можно как нибудь задефайнить?

---

maxgrind

24 Февраль 2011 в 13:58

ммм… догадываюсь что нужно будет писать не дефайн а equ. типа: .equ my_var = 0x060

★ DI HALT

24 Февраль 2011 в 16:59

МОжно и так, но это неудобно. Проще задать метки в DSEG

★ DI HALT

24 Февраль 2011 в 16:57

Можно. Делаешь в .DSEG метки и указываешь величину данных. Вот так:

```
1         .DSEG
2  Mode:      .byte              1      ; Переменная режима отображения
3  ADC_Data:   .byte              1      ; Тут складируются данные АЦП
4  ADC_OLD:   .byte              1      ; Тут хранится предыдущее значение данных АЦП
5
6  ADC_DIG0:   .byte              1      ; В этих трех ячейках хранятся ASCII коды
7  ADC_DIG1:   .byte              1      ; данных из АЦП, подготовленные к выводу на экран
8  ADC_DIG2:   .byte              1      ;
```

★ DI HALT

24 Февраль 2011 в 16:58

При этом число после byte показывает размер переменных в байтах.

maxgrind

24 Февраль 2011 в 20:49

спасибо

salpingot

26 Февраль 2011 в 2:56

```
; В свое время не смог понять смысл параметра в макросе,
; и тут меня осенило решил поделиться с теми до кого не дошло
; что такое параметр. Параметр — это переменная (X,Y и т.п.) в уравнении
; макроса, они будут заменены на значения РОН, (0-255)
; или выражения (.equ или .def) или и т.п.
; нумерация начинается с нуля разделяются запятой 0@,@1,@2,@3…@9,
; итого 10 параметров в теле одного макроса. Т.е. с помощью
; макросов можно писать своеобразные уравнения/формулы для
; значений в программе.
.MACRO stek;
ldi @1,low(@0);
ldi @2,high(@0);
out SPL,@1;
out SPH,@2;
```

.ENDM;
; применение макроса с параметрами для инициализации стека одной строчкой,
stek (RAMEND),r20,r21; где:
; ^ ^ ^ ^ (попытался замутить стрелочки
; имя макроса | | | не знаю на сколько получилось)
; где параметр @0 , @1, @2, и т.д. можно до @9 разделитель запятая.
; и еще одна фишка если скампилировать код (ctrl + F7), и навести после
; в моем примере на (RAMEND) то можно будет увидить чему равнялось
; значение .equ или .def можете проверить в файле m16def.inc,
; значение а там .equ RAMEND =$45F, удобно получается очень, не держать
; все переменные в голове.
; ДА И ДЛЯ НОВИЧКОВ ТАКИХ КАК Я, повторю ссылки еще раз,
; МОЖЕТ ПРИГОДИТЬСЯ Справка по Ассемблеру для Atmel AVR
; 1. http://dfe3300.karelia.ru/koi/posob/avrlab/avrasm-rus.htm
; 2. http://mymcu.ru/Articles/Atmel11.htm
; как смог объяснил, если что пишите в личку, и обязательно на форум, что бы
; другие товарищи смогли нагуглить и ознакомиться.

---

salpingot

26 Февраль 2011 в 2:59

стрелочки не получились напишу по другому stek=имя макроса (RAMEND)=@0,r20=@1,r21=@2;

★ DI HALT

26 Февраль 2011 в 8:30

Проблема твоего макроса в том, что он не учитывает тот факт, что у малых авр (тини) указатель стека может быть и однобайтным.
Зависит от размера ОЗУ больше 256 байт или меньше

---

salpingot

26 Февраль 2011 в 10:32

есть такая тема, просто я сделал специально такой пример, (вроде в пинбоорде стоит мега 16), цель которую я преследовал этим примером показать что это за знак «@» — т.е знак параметра, сам в одно время не мог понять что это просто обозначение переменной в уравнении, как в обычной математике.

---

salpingot

1 Март 2011 в 4:35

а вот это другой пример более универсальный:

```
.macro stek;
ldi r16,low(RAMEND);
out SPL,r16;
.if (RAMEND)>=0x0100;
ldi r16,high(RAMEND);
out SPH,r16;
.endif;
.ENDM;
```

★ DI HALT

1 Март 2011 в 10:05

Во. Шаришь

Crystaly

10 Август 2013 в 3:55

А еще, для удобочитаемости можно делать так:

```
;---------------------------------------------------
;ВЫВОД В ПОРТ
;---------------------------------------------------
; @0 @1
.MACRO OUTP ; port , reg
.set port = @0
.if port <= 0x3F
out port,@1
.else
sts port,@1
.endif
.ENDMACRO
```

Здесь в начале макроса мы в комментарии указываем имена параметров,
а внутри макроса параметры можно поименовать директивами .set и работать уже с именами, а не с @..

Crystaly

10 Август 2013 в 3:57

сожрались дополнительные табуляции, в общем @0 должен находиться над port, @1 должен находиться над reg

---

salpingot

28 Февраль 2011 в 23:13

А вот и дерективы препроцессора ссылочка для новичков http://azbukavb.narod.ru/cdoc/pdirectives.html

---

sam505

16 Апрель 2011 в 1:27

Подскажите, люди добрые, куда нырнуть на тему «ассемблер для AVR для занятых».
Погуглить то можно, но нет времени на проглатывание лишней инфы.

---

salpingot

17 Апрель 2011 в 18:08

Что значит “ассемблер для AVR для занятых”? типа времени мало а хочу много? есть краткие курсы см. раздел книги на этом сайте справа есть указатель http://easyelectronics.ru/category/knigi скачай в телефон и читай. а если совсем коротко то можно обойтись и двумя командами, sbi ddrX, ? (где ? номер ножки) sbi portX,? (где ? номер ножки) вкл, cbi portX,? (где ? номер ножки) диодик выкл. (впервое время всегда хочется экшена, хотя бы проверить работает ли хоть что нибудь) Все равно нужно время, может уточнишь что значит занятых???

---

sam505

17 Апрель 2011 в 22:44

Простите, что не уточнил. Время выделено и почитать можно и с эрана и с твердой копии.
От уважаемого сообщества мне нужен необходимый и достаточный перечень литературы по теме. Так чтобы лаконично, но твердо схватить ассемблер за хвост. Можно взять не Ту книжку и… :)
З.Ы. Когда разменяешь 5й десяток, то могзи скрипеть начинают от накопленной инфы… :).

---

salpingot

18 Апрель 2011 в 1:22

1. http://narod.ru/disk/7773184001/Микроконтроллеры%20AVR%20семейства%20Mega.djvu.html
2. http://narod.ru/disk/10432792001/Последняя%20на%20сегодня%20версия.doc.html
3. http://narod.ru/disk/10432877001/Расчеты%20сопративления.xls.html

Ну вот если надо кратко. Остальное есть в инете (включая этот ресурс, он тоже достаточно краткий).

★ DI HALT

18 Апрель 2011 в 5:18

Да тут и книжка не нужна. Достаточно систему команд перед глазами иметь.

d-lun

12 Май 2011 в 14:55

Странно, у меня компилятор на .undef ругается.

★ DI HALT

12 Май 2011 в 19:49

Вот чо гласит мануал:

This directive is only available with AVRASM2.

‘The UNDEF directive is used to undefine a symbol previously defined with the DEF directive. This provides a way to obtain a simple scoping of register definitions, to avoid warnings about register reuse.

Syntax:
.UNDEF symbol

Example:
.DEF var1 = R16
ldi var1, 0x20
… ; do something more with var1
.UNDEF var1

.DEF var2 = R16 ; R16 can now be reused without warning.

T.e. надо в качестве компилятора указать avrasm2

---

gks5

4 Июнь 2011 в 21:28

Опечатка:
Указывать размеры перменных нужно для того

---

tyuseman

10 Июнь 2011 в 15:49

Читал статью «Программы и прерывания». Дошёл до раздела «Подпрограммы vs Макросы». Решил попробовать написать макрос, аналогичный подпрограмме Wait (цикл DELAY раз). Получилось:

```
.MACRO MWAIT
M1: LDI R17, @0
DEC R17
NOP
BRNE M1
.ENDM
```

Зацикливается на нём. Пробовал разобраться, но при отладке программа не заходит в макрос, просто висит на этой строке. Видимо, это связано с меткой. В макросах в принципе нельзя использовать метку? Возможно, ответ на этот вопрос откроет мне глаза на то, что такое метка.

P.S. Почему-то под той статьёй комментарий не смог оставить. Пишет «необходимо войти».

★ DI HALT

10 Июнь 2011 в 16:02

И не будет заходить. Оно его будет выполнять как одну команду. Но т.к. студия подтупливает, а там у тебя задержка, то оно на нем и втупляет надолго.

В макросах нельзя использовать метки, т.к. если макрос используется дважды. то метка повторится, а это моветон. Но можно использовать переходы со смещением типо BRNE PC-1 или PC+1 где PC текущая строчка, а +X или -X смещение в словах относительно нее. Но учти что не все команды занимают размер одно слово. Бывают и в два слова. Тут лучше под отладчиком прогонять такой кусочек прежде чем его в макрос пихать.

---

tyuseman

10 Июнь 2011 в 16:06

Спасибо! Понял — всё руками, попробую. Только сразу скажу — не то, что надолго втупляет — а как будто бы навсегда :). То есть ставлю брейк на следующую после макроса команду, F5, ну и всё — висим.

---

tyuseman

10 Июнь 2011 в 16:09

Ой, я дурак! Метку на LDI, а не на DEC поставил. Но всё равно было полезно — узнал, что всё же нельзя метки ставить в макрос на случай повторения.

★ DI HALT

10 Июнь 2011 в 16:37

Кстати, отлаживать с развернутыми макросами можно по вкладке дизассемблер. Там нет символических имен, зато все макросы развернуты

meps

14 Август 2011 в 19:23

А что — совсем нет локальных меток в макросах? По-моему в masm под x86 локальные метки есть. Я хочу использовать вложенные макросы и не хотелось бы каждый раз вручную вычислять длину перехода.

★ DI HALT

15 Август 2011 в 19:18

Увы сравнивать masm и avrasm это что сравнивать последнюю модель мерседеса и жигули копейку.

Crystaly

10 Август 2013 в 4:16

Локальные метки можно сделать, хитрыми способами,
СПОСОБ 1.
Если переходы в макросах только вверх, как в примере с задержкой, то метку надо обозвать через директиву .set и программный счетчик PC:

```
.MACRO MWAIT
LDI R17, @0
.set M1 = PC
DEC R17
NOP
BRNE M1
.ENDM
```

СПОСОБ 2.
Поименование меток дополнительным параметром! Используем хитрый оператор «склеивания» ##

```
.MACRO MWAIT
LDI R17, @0
CPI R17,0
BREQ M2##@1
M1##@1:
DEC R17
NOP
BRNE M1##@1
M2##@1:
.ENDM
```

Каждый раз при вызове макроса надо добавлять параметр — номер вызова, например:

```
MWAIT 10,1
..
MWAIT 10,2
..
MWAIT 10,3
```

и т.д.

теперь в первом макросе метки будут M11 и M21, во втором M12 и M22 и т.д. и не будут повторяться!

---

lixlex

24 Декабрь 2011 в 20:56

а есть директивы как в ассемблере под mcs51 типа .org чтобы распологать команды в определенных местах памяти?

★ DI HALT

24 Декабрь 2011 в 21:00

конечно есть. Угадай как она зовется? ;)

---

lixlex

24 Декабрь 2011 в 21:12

ой невнимательно просмотрел!!

---

oleg22ov

29 Декабрь 2011 в 11:04

здравствуйте. Di halt скажи че не так я написал, студия ругается.
ругается на макрос pop_stec

```
.macro push_stec
in r00,sreg
push r00
.endmacro

.macro pop_stec
pop r00
out r00,sreg
.endmacro

;=========================================êîíåö ìàêðîñà=======================

;————————————————————îáðàáîò÷èêè ïðåðûâàíèé————————-

rx_ok: push_stec
in r16,udr
```

pop_stec
reti

★ DI HALT

29 Декабрь 2011 в 19:26

wtf r00 ??? Нет такого регистра, есть r0

Crystaly

10 Август 2013 в 4:19

в команде OUT надо сначала порт, потом регистр, то есть:
out sreg,r00

---

tiamat-666

15 Январь 2012 в 4:36

Здраствуйте. Вот прачьол статью все коментарии, и так не могу понять что такое параметр макроса «@0, @1,…@9». Если можна поясните болие детально, ато зациклиса ни них!!! :-(
Интересуют такие свойства как:
1. Что такое етот параметр «@» детальная инфа ?
2. Как присвоить ему дание? тоесть,обявление макроса(возьмем пожалуй товой пример DI HALT):
.MACRO SUBI16
subi @1,low(@0) ———вичитать с параметра 1 параметр 0 (но где и каким образом мне задать ети значения, дание или константи, чтоб копилятор занал что вичитать)
sbci @2,high(@0)
.ENDM

Зарание большое спасибо

Crystaly

10 Август 2013 в 4:32

Параметры макроса — это значения, которые перечисляются при вызове макроса после его имени, через запятую. Например, в вашем примере макрос SUBI16 может вызываться так:
SUBI16 r16, r17, 1000
в этом макросе 3 параметра: @0, @1 и @2 (нумерация с нуля)
тогда параметр @0 будет равен r16, параметр @1 будет равен r17, а параметр @2 будет равен 1000

К сожалению, реализация макросов в AVRStudio абсолютно убогая в 21 веке! Еще в 2000 году я работал с убогими (по сравнению с AVR) микроконтроллерами Z86 но какой замечательный ассемблер у них был в Zilog Developer Studio! Там наш макрос выглядел бы примерно так:

```
.MACRO SUBI16 register1, register2, value
subi register1,low(value)
sbci register2,high(value)
.ENDM
```

Don

13 Апрель 2012 в 2:33

Можете привести пример многомерного массива?В качестве практической части дипломной работы хочу соорудить матрицу 8 x 8 + для нее написать змейку,а без массива тут не обойтись.Подкиньте хотя-бы пищу для размышлений,иначе придется на C писать))

★ DI HALT

13 Апрель 2012 в 2:45

В ассемблере нет понятия массива. Она только у тебя в голове. Т.е. у тебя будет X строки и будет Y столбцы
А еще будет начало массива, адрес нулевой ячейки N

А адрес произвольной ячейки будет как Z = N+(X*8+Y) умножение на 8 лучше сделать сдвигом.

Ну и просто резервируешь 64 байта в ОЗУ под это дело

Crystaly

10 Август 2013 в 3:19

Ну зачем же человека расстраивать… можно и массивы и структуры придумать.

```
;Примеры работы с массивом, индексы-константы,
;нумерация индексов начинается с 0
;проверки на допустимые индексы нет
;(но можно сделать, особенно в первом варианте)

;Задать массив
;Размеры массива
.equ X_SIZE = 8
.equ Y_SIZE = 8

ARRAY: .byte X_SIZE*Y_SIZE

ВАРИАНТ 1
---------

;Читать ячейку массива
.MACRO RD_ARRAY
lds @0, ARRAY+@1+Y_SIZE*@2
.ENDM
```

```
;Записать ячейку массива
.MACRO WR_ARRAY
sts ARRAY+@0+Y_SIZE*@1, @2
.ENDM

;Использование

;Читать в регистр r16 ячейку (3;5)
RD_ARRAY r16,3,5

;Записать в ячейку массива (4;6) регистр r17
WR_ARRAY 4,6,r17
```

ВАРИАНТ 2

```
;Вычислить адрес ячейки массива
#define ARR(x,y) x+Y_SIZE*y

;Использование

;Читать в регистр r16 ячейку (3;5)
lds r16,ARR(3,5)

;Записать в ячейку массива (4;6) регистр r17
sts ARR(4,6),r17
```

Если хочется использовать индексы-переменные, то и так придумать можно))

DAVE_ELEKTRIK

27 Апрель 2012 в 0:37

Доброго времени.
Вот столкнулся с очень интересным выражением:
Typa_metka:
.db (DevNameStringDescriptorEnd-DevNameStringDescriptor)*4-2,3
или
.dw 9+9+7
вот еще
.db MaxUSBCurrent/2,0×09
(случайно попался исходник)
Так вот, возвращаюсь к выше описанной статье для начинающих. Сложные проги никогда не писал, я любитель (хобби). Я считал что после директивы .db пишется только константа через запятую (какое-то число) и все.Но тут после директивы какие то вычисления, я сначала не поверил что так можно! А куда результат то пишется и как это происходит? Я обычно пытаюсь составить кусок программы и пошагово пройтись в дизассемблере что бы разобраться как это работает, а тут из за не понимания я даже не знаю как это сделать!
Вот блин засада! Хелп!

★ DI HALT

27 Апрель 2012 в 1:02

А тут все элементарно. Перед компиляцией компилер проходится по этим вычислениям, вычисялет их и в код уже идет констатнта. Там же все считается из заданных констант. Просто так наглядней, видно что из чего, удобно править. Можно и на калькуляторе самому посчитать и вставить итоговый результат. Но зачем? Компилер справится с этим быстрей и проще.

DAVE_ELEKTRIK

27 Апрель 2012 в 8:16

Вобщем то понятно, только с константами спрятанными под символическими именами. Хорошо а как быть тогда с таким вот выражением:

LangIDStringDescriptor:
.db (LangIDStringDescriptorEnd-LangIDStringDescriptor)*2,3 ;length, type: string descriptor
.dw 0x0409 ;English
LangIDStringDescriptorEnd:
;——

Из LangIDStringDescriptorEnd вычесть не подсчитанный результат? Что то или я не недопонимаю, или тот исходник, шаблон недописанный. Всю программу можно посмотреть если нажать на выделенный текст в моем предыдущем посте. Все выделено красным, извиняюсь, неправильно использовал HTML

★ DI HALT

27 Апрель 2012 в 12:09

А что тут сложного то?

LangIDStringDescriptor и LangIDStringDescriptorEnd это метки.

Метки с точки зрения компилятора это всего лишь адреса. Т.е. ты берешь адрес LangIDStringDescriptorEnd вычитаешь из него LangIDStringDescriptor получаешь размер секции LangIDStringDescriptor умножаешь на 2, чтобы получить размер в байтах, а не в словах. Это и пишется в db следующий байт там 3. А дальше коммент.

Это можно и цифрами записать, но намного намного сложней, т.к. ты же не знаешь где у тебя будет секция и по какому адресу — код чуть увеличился, секция поплыла, а так компилятор сам подставит итоговое значение.

DAVE_ELEKTRIK

29 Апрель 2012 в 14:13

Спасибо, немного разобрался. Тяжело так сразу понять, но… нечего, тяжело в ученье легко в бою :).

XupypG

19 Май 2012 в 1:22

Не первая Ваша статья, которая мне очень нравиться и пригодилась. Спасибо!

TKV

22 Май 2012 в 16:58

Добрый день! Стоит студия 4.18.684. Имею такой макрос(вырезал донельзя пока искал ошибку):
.MACRO NAME_X
push r16
ldi r16,@0
pop r16
.ENDM
На строке с командой ldi выдаёт ошибку: syntax error, unexpected '\n'
Меняю @0 на @1 всё работает, причём правильно(случайность?). Где косяк?

★ DI HALT

23 Май 2012 в 22:34

Хм, а может @0 просто нелегальный параметр? Надо справку почитать, уточнить от какого номера все эти @ идут.

BARMALEY802

31 Май 2012 в 1:48

Подскажите, вычитал сверу в постах, что: «В макросах нельзя использовать метки, т.к. если макрос используется дважды. то метка повторится, а это моветон.» На сайте РАДИОДЕД видел статью про написание макроса, в качестве примера приводилась часть какой то программы: ;==
.equ PAUSE =60000 ;константе (15 с.) присвоили имя – « PAUSE »

```
.equ LED_ON =800 ;константе (0,2с) присвоили имя « LED_ON »
.equ LED_OFF =4500 ;константе (1,125с.) присвоили имя – «LED_ OFF »
.def counter =r18 ;Регистру повторения количества раз *delay* имя — « counter »
;============================================================
;============================================================
; macros delay
;============================================================
.macro delay ; присваиваем имя макросу — « delay »
Ldi counter,@1 ; здесь @1- это коэффициент повторения макроса K раз(1-255)
K:
ldi XL,Low (@0) ;Загрузить младший байт константы времени
ldi XH,high(@0) ;Загрузить старший байт константы времени
delay_:
sbiw XL,1; вычитаем единицу из двухбайтного слова загруженного в регистровую пару X
brne delay_
dec counter
brne K
.endm ; конец макроса
;============================================================
```

В этом макросе использованы аж две метки » delay_» и «K». В чем прикол? Я правдо запустить его не смог, стидия ругается на него.

★ DI HALT

31 Май 2012 в 6:45

А прикол наверное в том, что макрос был написан и использован всего один раз. Т.е. пример был исключительно учебный, не пригодный к реальному использованию.

Crystaly

10 Август 2013 в 4:38

про локальные метки в макросах я написал здесь:
http://easyelectronics.ru/avr-uchebnyj-kurs-makroassembler.html#comment-36719
ваш пример нужно переписать так (здесь переходы только вверх):

```
.macro delay
ldi counter,@1
.set K = PC
ldi XL,Low (@0) ;Загрузить младший байт константы времени
ldi XH,high(@0) ;Загрузить старший байт константы времени
.set delay_ = PC
sbiw XL,1; вычитаем единицу из двухбайтного слова загруженного в регистровую пару X
brne delay_
dec counter
brne K
.endm
```

и должно работать при многократных вызовах

Chelovek

3 Сентябрь 2012 в 17:48

А как сохранять макросы? Или их каждый раз надо заново вписывать?

Shureg

27 Октябрь 2012 в 18:55

Где должны лежать include-файлы? В папке с проектом?

★ DI HALT

27 Октябрь 2012 в 22:18

Не принципиально, лишь бы пути были прописаны. ПО дефолту ищет в папке с проектом, но можно же и прямой путь до файла указать, а не только имя.

Andrey41

5 Декабрь 2012 в 1:07

«В макросах нельзя использовать метки, т.к. если макрос используется дважды. то метка повторится, а это моветон.»
Использовать метки в макросах можно. Метки в макросах всегда локальные, их действие ограничено директивами .macro и .andm

---

★ DI HALT

5 Декабрь 2012 в 1:54

Обана, точно. Странно, всегда был уверен в обратном. И ведь неспроста, где то я на эти грабли наступал уже. Может это только в avrasm2 или в новых версиях компилера. Потому, что помню что трахался с локальными метками и пришлось высчитывать адреса, чтобы использовать переходы в макросах.

Crystaly

10 Август 2013 в 4:44

Интересно когда это появилось? В какой версии студии? А то я как тот партизан из анекдота: «…А я до сих-пор поезда под откос пускаю».

VeniaminCaver

7 Ноябрь 2013 в 2:25

Сейчас скажу как я понял, а Вы мне ответьте верно ли понял, и если не верно — то поправьте:
Когда мы указываем .CSEG то всё что мы напишем помещается во флеш — с которого можно только читать и это будет сам код

программы. В самом коде программы используются директивы — адресная метка — это директива, .ORG — это директива.
Во флеше могут содержаться как всякие данные так и код, который нужно выполнять последовательно — то есть мы например когда задаём кучу констант, указываем адресные метки — то их необязательно выполнять последовательно, следовательно мы эти данные (допустим на 200 строк) просто перепрыгиваем и говорим .ORG адрес_с_которого_выполняем_последовательно -и программа понеслась, но в этом коде, который пошёл мы можем использовать все те данные, которые мы перепрыгнули.

VeniaminCaver

7 Ноябрь 2013 в 2:49

Когда пишем .DSEG то всё что мы напишем попадёт в оперативку. В оперативке есть оператор(что-то вроде директив во флеше, хотя и не совсем то) BYTE
Когда мы говорим
var1: .BYTE 1
table: .BYTE 10
То это значит что у нас в оперативке зарезервируется 11 байт памяти, 1 байт мы оставляем для переменной var1, следом за ним 10 байт мы оставляем для переменной table. Эти 11 байт у нас резервируются в самом начале оперативной памяти, если напишем ещё time: .BYTE 89 то под переменную time будет выделено ещё 89 байт памяти, итого у нас в оперативке будет занято 100 байт с самого начала, и останется ещё 924 байта. При всём при этом — ячейки будут только зарезервированы, и переменная var1 может записаться в ячейку 1 через 2 часа работы контроллера, когда например какие нибудь данные поступят с порта от подключённого к одной из ног датчика.

VeniaminCaver

7 Ноябрь 2013 в 2:59

EEPROM — этакий мини жёсткий диск контроллера — медленный и часто писать не стоит, но можно сохранять данные, которые потом могу понадобиться после ряда «включений-выключений» устройства — например можно сохранить сюда индивидуальные настройки пользователя устройством, или скажем сохранять какие то данные отловленные за день — например замеряем температуру в течении дня, а вечером вычисляем среднюю и записываем в EEPROM в константу, а потом можно будет за пару недель посмотреть отчёт о том в какой день какая температура была, при том, что на ночь эта приблуда отключается.

VeniaminCaver

7 Ноябрь 2013 в 3:09

Написали:
MACRO SUBI16 ; Start macro definition
subi @1,low(@0) ; Subtract low byte
sbci @2,high(@0) ; Subtract high byte
.ENDM ; End macro definition

Затем в любом месте программы написали
SUBI16 0x1234,r16,r17
и прога налету берёт тот самый макрос, что у нас в начале файла(или вообще вынесен в другой файл) и на места @0 @1 @2 подставляет соответственно 0x1234,r16,r17.

Вот тут вопросик только: макрос вызывать можно только в .DSEG?

Мне важно понять — вообще я верно вкуриваю или нет — потому что я сколько раз пытался освоить что то из программирования — не получалось — зато сейчас, когда устройство железа понятно, механизмы работы самого устройства понятны осталось понять механизмы обработки и записи данных в память и взаимодействие с самой программой, чтобы идти дальше. Спасибо большое!

★ DI HALT

7 Ноябрь 2013 в 8:38

Именно так. Компилятор собирает из макроса кусочек кода и вставляет в общую кучу. Это, кстати, видно в lst файле. Там все макросы развернуты, а на полях написано что за макрос тут вставлен.

Можно его вызывать только там, где есть код. В .CSEG. Собственно макрос это и есть такой же исполняемый код, но не в виде окончательного решения, а в виде некой заготовки с условиями, по которым ее допишет при первом проходе компилятор,

исходя из входных условий самого макроса, а на втором проходе жестко впишет в код и превратит в прошивку.

★ DI HALT

7 Ноябрь 2013 в 8:34

Так точно. Там, собственно и хранят всякие константы, настройки и прочую фигню которая должна пережить выключение.

VeniaminCaver

7 Ноябрь 2013 в 3:22

Извиняюсь за простыню из камментов. Насколько я понял директива это команда, которая не переводится в команду процессора, а оператор — переводится, но при программировании для нас это разницы не имеет.

★ DI HALT

7 Ноябрь 2013 в 8:46

Директива это то, что указывает компилятору где и как делать. Т.е. это команда, но не для процессора, а для компилятора который будет собирать программу в прошивку. Директивами указывают с какого адреса записывать команды, какой код должен быть развернут макросами по условию. Какой код должен быть включен и скомпилирован, а какой пропущен.

И для программиста разница конечно же есть. Т.е. если ты попытаешься сделать логику работы прошивки директивами компилятора, например макроусловиями .if то получишь полную фигню, у тебя компилятор эти if прогонит, запишет по ним жесткий код одной и ты его прошьешь и выполняться он будет по одному условию. Т.е. будет линейным как бревно.

А вот если тебе надо сделать и поддерживать две почти одинаковые программы, но с чуть чуть разной логикой, для Васи и для Пети, то директивами компилятора можно разграничить пути компиляции. Через if Вася — компилируем так, а if Петя компилируем эдак. Т.е. у нас в коде существуют сразу две разные версии одновременно и не мешают друг другу, т.к. каждая ветвь компилируется исходя из своих условий и на выходе получается один какой то вариант.

★ DI HALT

7 Ноябрь 2013 в 8:33

Именно так. Просто резервируется. Причем даже не столько резервируется (т.к. под резервированием обычно понимается некое монопольное использование переменной, ее какая то уникальность), сколько присваиваются именованные константы для екущего адреса. Т.е. var1 будет на адрес X, а table адрес x+1, т.к. в предыдущем var мы заявили 1 байт. При этом сами ячейки никак не инициализируются, никак не готовятся. С ними вообще ничего не происходит и никуда ничего не записывается. Но компилятор после всего этого знает, что говоря в программе var1 мы имеем ввиду конкретный адрес.

Мы можем, например указать десять раз по byte 1 и смело считать это десятибайтной переменной, но при этом будем иметь адрес каждой из них непосредственно. Т.е. var: это тоже такая же метка как и в коде, но в ОЗУ. Просто чтобы задать адрес. А BYTE это просто затычка, чтобы указать сколько байт отступить до следующей метки.

Можно вообще ничего не резервировать и не указывать, А себе на листочке написать что var1 = 0100, table = 0101 и когда нам надо будет где-то использовать var1 подглядывать в этот листочек и сразу вписывать нужный адрес сразу числом. Но это же неудобно! Адреса должен помнить и вычислять компилятор.

★ DI HALT

7 Ноябрь 2013 в 8:25

Да. Более того, во флеше, с точки зрения контроллера, данные не отличаются от кода. Для него это просто байты которые содержат команды. Т.е. он запросто может выполнить данные. Только мы знаем что это такое мы тут записали и что вот эти байты это не команды, а какие то данные. Потому их надо перепрыгивать.

VeniaminCaver

7 Ноябрь 2013 в 16:18

Спасибо! Всё предельно ясно! Пошагал я дальше — ждите ещё одного лоскута комментариев к следующему посту.

serega55

8 Ноябрь 2013 в 10:47

DI подскажи, что то не прокатывает такой макрос (в него я пытаюсь передать начало таблицы в программной памяти и прочитать таблицу). Studio ругается
error: Display: Unknown instruction or macro
==========
Display Menu_0
.MACRO Display
ldi Temp3,0
ReadArray:
ldi ZH,High(@0*2)
ldi ZL,Low(@0*2)
….. и т.д.
.ENDM
Menu_0:
.db «Menu0: Rx->cmd_1~»; ~ признак окончания строки

---

serega55

8 Ноябрь 2013 в 10:49

забыл….ругается именно на параметр Menu_0

★ DI HALT

8 Ноябрь 2013 в 10:55

Хм, может его парит вложенный макрос? Или подчеркивание в параметре?

serega55

8 Ноябрь 2013 в 10:54

все разобрался))) …. макрос в конец кода сунул)

serega55

11 Ноябрь 2013 в 17:16

Может кто помочь ? Простой проект: опрос клавы и вывод на экран меню в соответствии с нажатой кнопкой (т.е. не перелистывание вверх, вниз). Также передача по SPI по нажатию одной из кнопок. Все написано — не могу понять почему прога не висит в стартовом меню «Zastavka» а перескакивает как будто нажата первая (программно нулевая) кнопка и выводит её меню «Menu0». Причем при программной эмуляции в студии все работает нормально а на железе не хочет. Код весь взят из уроков DI, только опрос клавы с Радиокота скомуниздил. ПАМАГИТЕ ! весь мозг себе уже расколупал. Кому надо архив кину на мыло. …. DI мож глянешь, ну хотя бы бегло, на предмет глобальных багов)))

★ DI HALT

11 Ноябрь 2013 в 17:23

А ты сделай все мееедленно. Поставь в опрос задержку. А то может у тебя как у меня вот тут:

http://easyelectronics.ru/kovarnye-vch-cepi.html

serega55

11 Ноябрь 2013 в 17:48

Дело в том что клава не подключена еще а опрашиваемые входы подтянуты пулапами. Т.е. на входах по идее единица как будто не нажато ничего….поэтому маловероятен такой вариант…посмотрю конечно

serega55

11 Ноябрь 2013 в 17:56

т.е. юзаю все на твоей pinboardII нет никакой печатки, нет клавы, а только ЛСД подключен .

★ DI HALT

11 Ноябрь 2013 в 17:59

НУ тогда залоггируй все. Сделай затычку отладочную. Макрос или подпрограммку (макрос лучше) Натыкай их везде и по каждому шагу кидай в UART код и не сходи с этого места пока байт не уйдет. В результате у тебя программа каждый свой шаг выложит в уарт и ты поймешь по какому пути она у тебя крутится.

★ DI HALT

11 Ноябрь 2013 в 18:00

Еще проверь, а не нажата ли кнопка в реале? Может мусор какой попал или сопля припоя. Иной раз бывает.

serega55

11 Ноябрь 2013 в 18:13

буду проповать с затычками, спасибо за совет…..P.S. я даже на пинбоарде не подключаю клаву, эта ситуация эквивалентна отсутствию нажатых кнопок…..счас я этот код пытать буду….он мне через УАРТ все скажет))))

★ DI HALT

11 Ноябрь 2013 в 18:15

Скажу тебе по секрету… кнопка начинается от вывода микросхемы :)

serega55

11 Ноябрь 2013 в 18:45

Это я даже не спорю)))…тогда получается остается наводка одной ноги на другую…..у меня выставление строк и опрос столбцов енто один порт ….. попробую разнести

★ DI HALT

11 Ноябрь 2013 в 18:50

Я к тому, что проверь уровни на ногах мультиметром. А то может он нажат. Мало ли по какой причине. Я один раз пол дня искал почему у меня МК сбрасывается, а все дело было в крошечной ворсинке от провода которая упала под панельку и замкнула ресет на уарт.

serega55

11 Ноябрь 2013 в 19:43

DI оказывается и точно блин наводки….вобщем повесил на другой порт….хотя работает неправильно, но ближе к правде….держится несколько секунд в главном меню, а потом перескакивает на меню0…буду частоту сканирования уменьшать…

serega55

11 Ноябрь 2013 в 19:46

Но у меня еще один вопросик нарисовался….вобщем решил порты проверить на 16 меге — D и C.
Выставил их на вход с подтяжкой и начал напруги на ногах глядеть. Так вот на одно пине порта C (PC4) напруга 1,8 В, на остальных 3,8 (где то даже 4,1). Получается нога PC4 неисправна ?

★ DI HALT

11 Ноябрь 2013 в 19:48

Нет, она занята JTAG интерфейсом. Жтаг можно выключить в регистре MCUCSR битом JTD

serega55

11 Ноябрь 2013 в 19:54

Спасибо огромное … будешь у нас тут под Владимиром проезжать, есть тут городок на букву К — — — — -, заезжай на пиво))))…. извиняюсь за флуд)))

serega55

13 Ноябрь 2013 в 15:36

DI ;) что-то я рано обрадовался, помехи это да был грешок, но в процедуре опроса клавы херь какая-то … точно установил путь куда идет прога в железе и не могу понять почему туда…вот код, глянь пожалуйста если можешь, тут чисто для школьников))) ….почему прыгаю на SetKey ? Программная эмуляция говорит что все замечательно….((((….в коменте я расписал какие меры предпринял чтобы обеспечить 1 на PC0…PC2 … т.е. типа нажатых кнопок нет !!!!!!!
P.S. в инициализации портов можно не сомневаться, 100 раз проверил…DDRD — выход,
DDRC — вход с подтяжкой http://easyelectronics.ru/repository.php?act=view&id=116

serega55

13 Ноябрь 2013 в 15:46

Может кто-нибудь попробует по этому алгоритму клаву опросить ?

serega55

13 Ноябрь 2013 в 19:35

Вот тут дискуссию развернул http://forum.easyelectronics.ru/viewtopic.php?p=299349#p299349

★ DI HALT

13 Ноябрь 2013 в 19:41

Жтаг выключил? Ты пользуешься портом С, но его линии заняты жтагом.

В порту Д две линии заняты уартом бутлоадера это RX и TX PD0 и PD1 соответственно.

---

serega55

13 Ноябрь 2013 в 19:50

эээээ..не не выключил..просто пины переназначил ..но опять же половина с С половина с D…..а RX и TX просто перемычки снимал….пошел пробовать

★ DI HALT

13 Ноябрь 2013 в 19:51

Перемычки мало снять, надо UART выключить. Биты RXE и TXE снять. Иначе он так просто свои пины не отдаст, будет держать на них уровни свои. RX будет телепаться без подтяжки, а TX держать линию.

---

serega55

13 Ноябрь 2013 в 20:18

АААААААААААААААААААА ….. все РАБОТАЕТ…..теперь можно и америку ПАБЕДИТЬ …)))

Тру ля ля ….. ха ха ха ….четыре дня с бубном … даааа матчасть учить надо…)))))))

0073402

3 Декабрь 2013 в 4:17

Помогите разобраться в чём причина!

В коде программы управления датчиком температуры DS18B20, активно пользуюсь директивой .include

Код программы разросся, пришлось перейти на Atmega16. Сам код включает в себя 20 штук .include, каждое включение отвечает за отдельный сегмент кода. Например: .include»C:\AVR16\DIREKTIVm16\LCD_winstar_init \LCD_winstar_init.asm» выполняет инициализацию LCD дисплея. А в директиве .include»C: \AVR16\DIREKTIVm16\TIMEdelay\TIMEdelay.asm» записаны все временные задержки проекта.
По ходу роста кода увеличивалось количество директивы .include , до момента пока AVRstudio4 не выдал ошибку, Relative branch out of reach. Сообщая о невозможности перехода. Таких сообщений было 61 штука. При перестановки директивы на другую строчку, например она стояла по счёту на 10 месте, когда я её перенёс на 5-ое место ошибок стало 6шт. Следущий перебор даёт разное количество идентичных ошибок.
Перезагрузка компьютера, работа на другом компьютере, удаление и заново запись папки с проектом результатов не принесли. Установка проекта в AVRstudio6 дает идентичную ошибку.

★ DI HALT

3 Декабрь 2013 в 4:28

Широко шагаешь, штаны порвал.

Дело в том, что команды BR*** RJMP и прочие относительные переходы имеют ограниченную длину перехода. Т.е. BR*** может ходить только на 127 команд вверх/вниз от себя. Для более дальнобойных переходов используется уже JMP т.к. он двубайтный и там полный адрес вмещается, который покарывает всю память. RJMP прыгает дальше чем BR**, но тоже не во всей области памяти, но его обычно все же хватает всегда.

Соответственно ваши переносы директив приводя к сокращению шагов и уменьшению количества ошибок.

---

valentinych

31 Январь 2014 в 1:06

Добрый день. Изучаю ваш учебный курс по AVR
Поставил 4ю версию студии и ВинАвр.
Попробовал первые строчки как по учебнику…

…Теперь если ты забьешь в проект простейший код, например,
1 .include «m8def.inc»
2 NOP
…
Жму Кнопку «скомпилить и выполнить» Вылетает окно с сообщением
AVRASM: AVR macro assembler 2.1.13 (build 208 Sep 11 2007 14:22:34)
Copyright (C) 1995-2007 ATMEL Corporation

C:\WORK\first1\first1.asm(1): error: syntax error, unexpected INTEGER
Assembly failed, 1 errors, 0 warnings

В чем ошибка не пойму

★ DI HALT

2 Февраль 2014 в 16:51

С кавычками не накосячил?

---

valentinych

2 Февраль 2014 в 22:28

Не должно быть? Делал все как у тебя))) написано. Вопрос у меня конечно ламерский, от незнания синтаксиса макроассемблера.

★ DI HALT

2 Февраль 2014 в 22:35

Копипастил поди, а копипаст зло. У меня блог кавычки заменяет на неправильные :)

valentinych

3 Февраль 2014 в 21:25

Сюда копипастил, а так кавычки верхние. Не идет , не понимаю. Может из за виртуальной машины? Основная система W8, x64, виртуалка XP. Может пути надо прописывать в настройках???

★ DI HALT

3 Февраль 2014 в 21:59

Ругается только на инклюдник? А если полный путь прописать? Компилер не любит длинных имен и русских символов.

valentinych

3 Февраль 2014 в 23:05

Разобрался, первая строка по подключению библиотеки нужного МК лишняя, в начале проекта уже выбирается тип камня, такая же штукес в 6 версии, только что проверил и там и там. Одна строка с NOP работает на ура.
Доктор спасибо за уделенное время. Это реально моя первая удачная Хеловорлд в АВРках)))

★ DI HALT

3 Февраль 2014 в 23:10

Рано радуешься. В начале проекта выбираешь тип камня в котором будет работать эмулятор, только и всего.
А в инклюднике подключаешь конкретные имена регистров. На NOP ты это не проверишь, т.к. NOP это инструкция процессора, а не то, что описано в инклюдах на проц. Там привязки Имя регистра периферии -> адрес в памяти.
Попробуй обратиться напрямую к периферии, например к регистрам таймеров. Без инклюдов тебя компилер пошлет, мол не знаю таких буков.

valentinych

3 Февраль 2014 в 23:52

будем посмотреть)

valentinych

3 Февраль 2014 в 23:07

И еще вопрос. Есть в природе нормальное описалово 6-ки? Атмеловский сайт листал — ничего путного даже на англицком.

★ DI HALT

3 Февраль 2014 в 23:11

А зачем она нужна? Она кривая, глючная и не поддерживает старые системы отладки.

valentinych

3 Февраль 2014 в 23:51

Те ты советуешь работать в 4ке?

★ DI HALT

4 Февраль 2014 в 0:14

На начальном этапе да. Пока нужен симулятор, отладчик и все то, что выпилили в 6й версии.

Ygrryk

5 Февраль 2014 в 0:42

Доброго дня DI HALT.
Задача необходимо большой кусок программы использовать сначала с одними а потом с другим :
1) PORTD,PIND,DDRD с номерами 4 и 5

2)PORTB,PINB,DDRB с номерами 0 и 1
Время существенно.MA и SLO реализуется SBI и CBI , SBIC\RJMP SBIS\RJMP
Реализовано все на Tiny2313.если дважды переписывать и использовать .set и макросов 2K Bytes переполняются.
Подскажите как можно оптимизировать код.И возможно ли использовать регистр для адресации порта и ноги.
Прошу прощение за глупый вопрос.Но не могу найти не где необходимую информацию.
Спасибо.

★ DI HALT

5 Февраль 2014 в 1:38

Я не пробовал, но адресовать порт наверное косвенно можно. Они же все в память завернуты своими адресами. Как минимум через lds, sts. Но естественно побайтно. О бытовых операциях придется забыть. Так как краткие адреса вписаны непосредственно в опкод команды и так просто в них не везешь. Можно попробовать извернуться через условное компилирование, всякие .if операнды макроязыка. Чтобы в конкретное место писать конкретную команду.

---

and_master

2 Июль 2014 в 12:10

Всем привет. Несколько вопросов.
1. Я так понимаю, что метка представляет из себя начало адреса команды или данных. Значение метки 16-ти разрядное. Почему в некоторых командах нельзя метку заменить регистровой парой. Например jmp Z.
2. Почему нельзя писать ld r18, 0 . Непосредственный адрес и адрес , записанный в регистровой паре это не одно и тоже?

★ DI HALT

2 Июль 2014 в 13:46

1. Метка с точки зрения компилятора и есть адрес. Адрес текущей инструкции или области памяти. А нельзя заменить потому, что ассемблер это не язык программирования. А просто команды проца с человеческим лицом, команд несколько сотен. MOV R16,R0 это

одна команда, а MOV R17,R0 уже совсем другая команда. И если что то нельзя сделать, то только потому, что такой команды просто нет.

Может быть команда LDI R16,*** , а вот команды LDI R15,***, внезапно, уже не существует. Хотя, казалось бы, должна. Так и команды JMP Z не существует. Только и всего. (есть правда ijmp которая как раз для этого, но она есть не во всех МК AVR)
2. Опять же потому, что нет такой команды LD R18,0 LD таскает данные только между регистрами. Хочешь записать в регистр число используй группу команд LDI. Непосредственный адрес это просто некое число. И только. И с точки зрения компилятора и процессора это просто число. Он даже не знает, что это какой то адрес. Число и все. И только ты знаешь, что это не просто так число, а некий адрес. Его можно внести как непосредственные данные (команда LDI), куда либо, можешь записать в регистровую пару.

---

and_master

2 Июль 2014 в 21:43

Большое спасибо за ответы! Но мне кажется , что команды ld r16,r0 не существует, поправьте , если ошибаюсь. И еще. Если в сегменте кода написать .db 1, то под данные будет отведено 2 байта, где неуказанный второй байт будет нулевым и остальной код продолжится через два байта, а не один. Правильно?

★ DI HALT

2 Июль 2014 в 21:55

Да, верно. Я про MOV. LD это на Z80 :)

Да, все верно. Байт лишним будет.

---

and_master

3 Июль 2014 в 18:46

Привет Di! Еще пару вопросов. Процессор AT Mega16 8-рязрядный.
1. Каким образом он исполняет команды, которые по сути все 16-разрядные. И куда команды загружаются для исполнения из декодера

команд. Непосредственно в АЛУ? Сразу все 16 разрядов? Или понятие разрядности относится только к размеру регистров? Хотя регистр PC также является 16-разрядным. Это какой-то мутный момент для меня).
2. И еще. Адресация данных во флеш также 16-разрядная (по команде lpm), если принять старшие 15 разрядов за номер слова, а младший бит за флаг выбора старшего или младшего байта в слове. Правильная концепция?

★ DI HALT

3 Июль 2014 в 22:06

Команды могут быть любой разрядности. Главное что за раз он обрабатывает 8 бит данных.
Команды загружаются в управляющее устройство которое их уже тащит на конвеер. Это все аппаратно и пользователю неподвластно никак. А уже УУ дергает и АЛУ и шину данных и все остальное. PC это счетчик, но считывать и обрабатывать его можно только побайтно. А так хоть сколько разрядов у него может быть.
Адресация тоже может быть любой разрядности. главное что мы жрать ее можем только по кускам в 1 байт.
Если брать адресацию в памяти для LPM, то она побайтная, т.е. ты просто указываешь адрес непосредственного байта. Другое дело ,что компилятор адрес метки всегда в словах берет т.к. ему главное тут шаги PC, а LPM не указ. Так что меткой ты байты только через одного адресовать можешь. Но ничего не мешает тебе сделать +1 и все.

---

and_master

4 Июль 2014 в 14:47

За раз обрабатывать это как? Вот команда относительного перехода Cx xx. УУ берет первый байт и каков результат обработки первого байта этой команды? По -поводу адресаци во флеше, Для команды lpm по-байтно, а вот для команд перехода в словах. Поскольку сам адрес в словах записывается в код команды, которая выполняется на аппаратном уровне, то почему говорится, что адрес в словах для удобства компилятора. Правильно понимаю?

★ DI HALT

4 Июль 2014 в 15:02

УУ, насколько помню, читает сразу по два байта за такт. И обрабатывает команду целиком. Разве что бывает команда из 4 байта — длинные переходы, например. Тогда перый байт говорит куда, а на втором такте забирается адрес и кладется в PC.

Да, это чисто компиляторские заморочки. Дело в том, что переходы считаются по PC, а адресация PC она с 0 и далее+1+1+1, где этот +1 означает выборку следующего слова, а не байта. Т.е. PC меряет код словами. И соответственно адресация переходов эта в словах. Т.е. смысла делать байтную адресацию для меток нет.

А вот LPM грузит данные и грузит по байту, поэтому может память адресовать не словами, а байтами.

---

and_master

4 Июль 2014 в 16:41

Спасибо за помощЬ, di! Правда интересует еще один вопрос). Если взять контроллер, в котором объем озу меньше 256 байт, то есть можно адресовать 8-разрядами, то можно ли в командах ld,st использовать любые регистры вместо регистровой пары?

★ DI HALT

4 Июль 2014 в 16:56

А ты в ДШ посмотри какие регистры входят в группу этой команды. Вроде бы только индексные. Конечно если адресация позволяет, то можно старший байт не менять (он всегда 0 будет), а младший будет адресом.

---

and_master

7 Июль 2014 в 12:40

Привет, Di! Поставил студию версии 4.19 вместо глючной 6.2, но и в ней косяки какие-то. Не очищается после rebuild память программ. Как с этим бороться? И вообще какую версию сам используешь?

★ DI HALT

1 Август 2014 в 1:31

И не должна. Кто тебе сказал, что она очищается? Это надо делать вручную циклом, иначе глюков не оберешься.

AFG 92

1 Август 2014 в 0:56

Всем здравствуйте.
Написал пробную программку в AVR Studio 4 вот отсюда(в самом низу):http://radiokot.ru/start/mcu_fpga/avr/07/. При попытке компиляции студия выдает:
D:\Projects\led2313\led2313.asm(9): error: Undefined symbol: DDRB
D:\Projects\led2313\led2313.asm(12): error: Undefined symbol: PortB
D:\Projects\led2313\led2313.asm(29): error: Undefined symbol: PortB
D:\Projects\led2313\led2313.asm(46): error: Undefined symbol: PortB
D:\Projects\led2313\led2313.asm(62): error: Undefined symbol: PortB
D:\Projects\led2313\led2313.asm(78): error: Undefined symbol: PortB
D:\Projects\led2313\led2313.asm(94): No EEPROM data, deleting D:\Projects\led2313\led2313.eep
Подскажите пожалуйста, в чем проблема?

★ DI HALT

1 Август 2014 в 1:31

Не подключен файл с символическими именами. Вначале должно быть .include «твой контроллерdef.inc» или как то так. Точный синтаксис уже не помню, давно на асме не писал.

Full_30h

26 Август 2014 в 20:15

Попытался студии 6 свою хуергу впарить, чего то не понравилось ей
.def LED_port = PORTB ; B5

.def LED_pin = 5

★ DI HALT

26 Август 2014 в 22:22

А макроопределения подцепил? А то может она вообще не знает что такое PORTB

Full_30h

27 Август 2014 в 0:02

Не, там оно по умолчанию всё подцеплено. Как выяснилось через .def только регистры обзывать можно, а все эти порты-шморты сами не местные и через m8Adef.inc к циферкам привязаны. Поэтому их через .equ

Bredov-IV

18 Декабрь 2014 в 3:21

Господа! Есть вопрос про статичные данные — data byte, data word и др.
Ну отметили мы массивчик .db, а мацать-то его чем? Команды чтения из флэша со смещением? А адрес начала где брать? Али надо этот оператор только после директивы .орг писать?
Хочу просто забить массив-битмап для разового вывода на дотматрикс.

★ DI HALT

18 Декабрь 2014 в 4:28

Ставишь метку, после нее пихаешь свою статику. Метка*2 = абсолютный адрес для LPM группы команд. Разумеется все делается в зоне ORG хочешь в начале, хочешь в конце. Только перешагивать не забывай, а то словишь.

kaki-malaki

13 Август 2015 в 21:36

Здравствуйте!
Товарищи, что-то упустил в статьях, не могу вточить.
Почему к примеру стек инициализируется в два подхода, захват старшего байта, потом младшего(most significant bite, lsb?)?
Это как-то свзяано с тем, что на самом деле адрес в мк в бинарном коде, а мы используем шестнадцатиричную систиму для краткости записи?
Заранее извиняюсь, если этот момент уже где-то оговаривался или вопрос сам по себе является глупым. Вот уже пару недель не дает мне покоя.
Заранее спасибо!

★ DI HALT

13 Август 2015 в 22:25

Ну так адрес у нас 16ти разрядный, а МК 8ми разрядный. Т.е. он может обрабатывать через свои регистры только по половине адреса. Вот мы его в два хопа и грузим.

---

kaki-malaki

14 Август 2015 в 10:29

спасибо!