

Свежие комментарии

- Николай к записи [STM Урок 13. HAL. USART. Передача данных](#)
- Николай к записи [STM Урок 13. HAL. USART. Передача данных](#)
- Narod Stream к записи [STM Урок 44. SDIO. FATFS](#)
- Михаил к записи [STM Урок 44. SDIO. FATFS](#)
- Анигилирую к записи [AVR Урок 8. Семисегментный индикатор. Статическая индикация](#)

Форум. Последние ответы

- alexander в [Программирование МК STM32](#)
2 дн., 4 час. назад
- Narod Stream в [Программирование МК STM32](#)
3 нед., 5 дн. назад
- Zandy в [Программирование МК STM32](#)
3 нед., 5 дн. назад
- Narod Stream в [Программирование МК STM32](#)
1 месяц, 1 неделя назад
- Narod Stream в [Программирование МК STM32](#)
1 месяц, 1 неделя назад

Февраль 2018

Пн	Вт	Ср	Чт	Пт	Сб	Вс
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28				
« Янв						

Архивы

- [Февраль 2018](#)
- [Январь 2018](#)
- [Декабрь 2017](#)
- [Ноябрь 2017](#)
- [Октябрь 2017](#)
- [Сентябрь 2017](#)
- [Август 2017](#)
- [Июль 2017](#)
- [Июнь 2017](#)
- [Май 2017](#)

[Главная](#) > [Программирование AVR](#) > AVR УРОК 39. Акселерометр LSM6DS3. Часть 4

AVR УРОК 39. Акселерометр LSM6DS3. Часть 4

Posted on [Январь 16, 2017](#) by Narod Stream
Опубликовано в [Программирование AVR](#)
— 5 комментариев ↓

Нужно программир-е контроллеров?

Комплексное обучение по продукции Siemens (SIMATIC S7). 5 уровней курсов!

[О компании](#) [Услуги](#) [Продукция](#) [Преимущества](#)

[festo.com](#) [Адрес и телефон](#)

Каталог БУ серверов! Более 300 шт.

Восстановленные серверы! Надежные и протестированные! Рабочие 100%. Гарантия 3 года!

[bystryi.ru](#) [Адрес и телефон](#)

Урок 39 Часть 4

Акселерометр LSM6DS3

Продолжаем работать с подключением акселерометра.

В [прошлой части занятия](#) мы закончили писать исходный код инициализации датчика.

Сегодня мы уже будем заниматься считыванием показаний нашего датчика по трем осям, которые время от времени нам нужно будет отправлять в шину USART. Для этого создадим функцию

```
void Accel_ReadAcc(void)
{
    _delay_ms(200);
}
//_____
void Accel_Ini(void)
```

Данная функция и будет заниматься обработкой показаний датчика и отправкой их для мониторинга в шину USART. Пока в ней только задержка.

искать здесь ...

Фильтровать

✕

DIVING

Анторус. Пут рейд. Все режимы. Маунт и тринкет.

7.3 новые услуги

Узнать больше

Заходите на канал

Narod Stream

- Март 2017
- Февраль 2017
- Январь 2017
- Декабрь 2016
- Ноябрь 2016

Напишем на неё протоип в заголовочный файл

```
void Accel_Ini(void);
void Accel_ReadAcc(void);
```

Затем вызовем её в бесконечном цикле в main()

```
while (1)
{
    Accel_ReadAcc();
}
```

В файле lsm6ds3.c напишем ещё одну функцию

```
void Accel_GetXYZ(int16_t* pData)
{
}
//_____
void Accel_ReadAcc(void)
```

Данная функция будет заниматься непосредственно выборкой данных из регистров и преобразованием их из пар в единые 16-битные целые числа. Причем данные значения могут быть и отрицательными. Кстати, здесь мы использовали тип значений необычный `int16_t`. Это то же самое что `short`. Кто программировал в Keil, тот данный тип знает. Мы постоянно там используем подобного рода типы. Считывание данных осей постоянно с частотой, которую мы настроили в инициализации, происходит в шесть регистров (для каждой оси — по два)

9.34 OUTX_L_XL (28h)

Linear acceleration sensor X-axis output register (r). The value is expressed as a 16-bit word in two's complement.

Table 99. OUTX_L_XL register

D7	D6	D5	D4	D3	D2	D1	D0
----	----	----	----	----	----	----	----

Table 100. OUTX_L_XL register description

[D7:0]	X-axis linear acceleration value (LSbyte)
--------	---

9.35 OUTX_H_XL (29h)

Linear acceleration sensor X-axis output register (r). The value is expressed as a 16-bit word in two's complement.

Table 101. OUTX_H_XL register

D15	D14	D13	D12	D11	D10	D9	D8
-----	-----	-----	-----	-----	-----	----	----

Table 102. OUTX_H_XL register description

[D15:8]	X-axis linear acceleration value (MSbyte)
---------	---

9.36 OUTY_L_XL (2Ah)

Linear acceleration sensor Y-axis output register (r). The value is expressed as a 16-bit word in two's complement.

Table 103. OUTY_L_XL register

D7	D6	D5	D4	D3	D2	D1	D0
----	----	----	----	----	----	----	----

Table 104. OUTY_L_XL register description

[D7:0]	Y-axis linear acceleration value (LSbyte)
--------	---

9.37 OUTY_H_XL (2Bh)

Linear acceleration sensor Y-axis output register (r). The value is expressed as a 16-bit word in two's complement.

Table 105. OUTY_H_G register

D15	D14	D13	D12	D11	D10	D9	D8
-----	-----	-----	-----	-----	-----	----	----

Table 106. OUTY_H_G register description

[D15:8]	Y-axis linear acceleration value (MSbyte)
---------	---

9.38 OUTZ_L_XL (2Ch)

Linear acceleration sensor Z-axis output register (r). The value is expressed as a 16-bit word in two's complement.

Table 107. OUTZ_L_XL register

D7	D6	D5	D4	D3	D2	D1	D0
----	----	----	----	----	----	----	----

Table 108. OUTZ_L_XL register description

[D7:0]	Z-axis linear acceleration value (LSbyte)
--------	---

9.39 OUTZ_H_XL (2Dh)

Linear acceleration sensor Z-axis output register (r). The value is expressed as a 16-bit word in two's complement.


Table 109. OUTZ_H_XL register

D15	D14	D13	D12	D11	D10	D9	D8
-----	-----	-----	-----	-----	-----	----	----

Table 110. OUTZ_H_XL register description


[D15:8]	Z-axis linear acceleration value (MSbyte)
---------	---

Так как регистры идут один за другим по адресам, то мы смело можем использовать чтение их всех в одном вызове, так как функция чтения из шины I2C, написанная нами ранее, это вполне



Конструирование и производство радиоаппаратуры

Купить



Рубрики

- 1-WIRE (3)
- ADC (6)
- DAC (4)
- GPIO (26)
- I2C (19)
- SPI (13)
- USART (8)
- Программирование AVR (131)
- Программирование PIC (8)
- Программирование STM32 (217)
- Тесты устройств и аксессуаров (1)

↗

Эт. день	140 271
	14 743
От. дней	38 361
	4 568
Вс. часов	4 468
	1 140
Сегодня	2 647
	712
Написано	78
	26

предусматривает. Считаем эти данные в предварительно объявленный буфер, а затем с помощью давно известных битовых сдвигов переложим в другой буфер уже в три нормальные 16-битные значения

```
void Accel_GetXYZ(int16_t* pData)
{
    uint8_t buffer[6];
    uint8_t i=0;

    I2Cx_ReadData(0xD4, LSM6DS3_ACC_GYRO_
    OUTX_L_XL, 6, buffer);
    for(i=0; i<3; i++)
    {
        pData[i] = ((int16_t)
        ((uint16_t)buffer[2*i+1]
        <<8)+buffer[2*i]);
    }
}
```

Теперь продолжим писать функцию Accel_ReadAcc. Здесь мы также подготовим переменные, вызовем только что написанную функцию и разложим всё уже в свои локальные переменные

```
void Accel_ReadAcc(void)
{
    int16_t buffer[3] = {0};
    int16_t xval, yval, zval;
    Accel_GetXYZ(buffer);
    xval=buffer[0];
    yval=buffer[1];
    zval=buffer[2];
    _delay_ms(200);
}
```

Теперь зайдём в файл **usart.c** и напишем там функцию для отправки данных в шину USART сразу по несколько байтов в самом низу файла, иначе у нас будет очень и очень много кода, если мы будем отправлять данные по одному байту

```
//-----
void USART_TX (uint8_t *str1,
uint8_t cnt)
{
    uint8_t i;
    for(i=0; i<cnt; i++)
        USART_Transmit(str1[i]);
}
//-----
```

Ну, и, конечно, напишем прототип в файле **usart.h**

```
void USART_Transmit( unsigned char
data );
void USART_TX (uint8_t *str1,
uint8_t cnt);
```

Вернёмся в нашу недописанную функцию. Сначала добавим глобальную переменную

```
unsigned char read_buf[10]={0};
char str1[30]={0};
```

Напишем в функции Accel_ReadAcc отправку данных в текстовом виде в шину USART

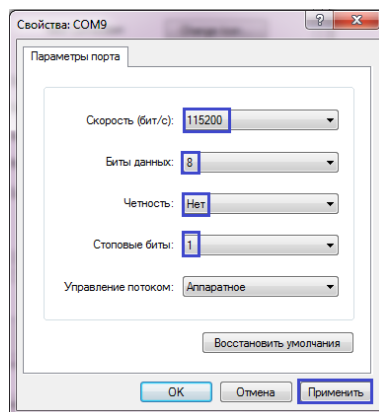
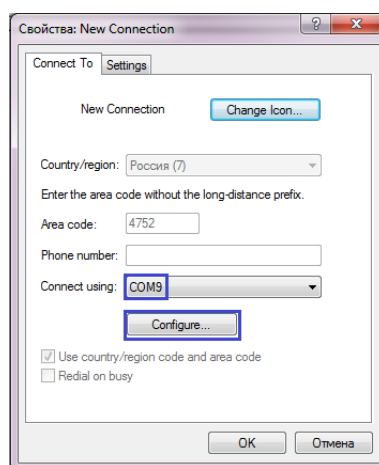
```
zval=buffer[2];
```

```

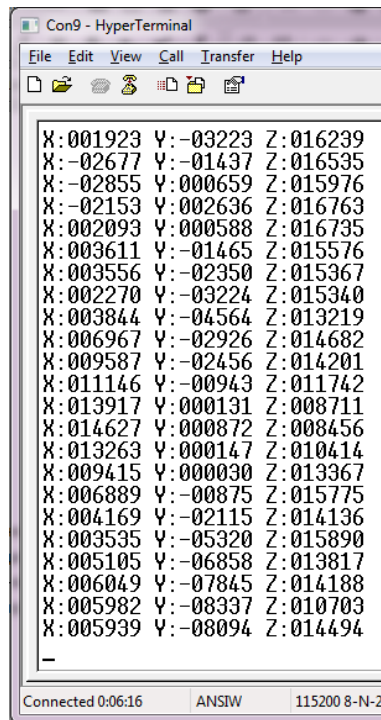
sprintf(str1,"X:%06d Y:%06d
Z:%06drn", xval, yval, zval);
USART_TX((uint8_t*)str1,strlen(str1)
);
_delay_ms(200);

```

Соберём код, прошьём контроллер и запустим терминальную программу. Правда программа будет несколько другая, так как предыдущая порой не справляется с внушительным потоком. Я решил использовать HyperTerminal. Была найдена версия для современных операционных систем (ссылку дам в конце урока). Настроим в данной программе порт



Нажмём в программе кнопку "call" и Программа нам приблизительно 5 раз в секунду начнёт показывать значения наших осей, правда это не в единицах ускорений свободного падения, но всё это спокойно можно посчитать, учитывая, что у нас максимум 2G, значит 2G будет у нас 32768, а -2G — -32768. Соответственно 1G будет в 2 раза меньше и т.д. Понаклоняем макетницу с платой ненадолго в разные стороны, и увидим, что показания постоянно меняются. Показания по оси Z приблизительно 1G означают, что данная ось у нас направлена вертикально и на неё постоянно действует ускорение земного притяжения 1G — она же гравитационная постоянная.



```

X:001923 Y:-03223 Z:016239
X:-02677 Y:-01437 Z:016535
X:-02855 Y:000659 Z:015976
X:-02153 Y:002636 Z:016763
X:002093 Y:000588 Z:016735
X:003611 Y:-01465 Z:015576
X:003556 Y:-02350 Z:015367
X:002270 Y:-03224 Z:015340
X:003844 Y:-04564 Z:013219
X:006967 Y:-02926 Z:014682
X:009587 Y:-02456 Z:014201
X:011146 Y:-00943 Z:011742
X:013917 Y:000131 Z:008711
X:014627 Y:000872 Z:008456
X:013263 Y:000147 Z:010414
X:009415 Y:000030 Z:013367
X:006889 Y:-00875 Z:015775
X:004169 Y:-02115 Z:014136
X:003535 Y:-05320 Z:015890
X:005105 Y:-06858 Z:013817
X:006049 Y:-07845 Z:014188
X:005982 Y:-08337 Z:010703
X:005939 Y:-08094 Z:014494

```

Если вы смотрите мои уроки также и по STM, то вы уже знаете, что я также люблю писать программки под ПК и поэтому мы сейчас немного переделаем код вывода в шину USART, и уже посмотрим данные с осей в виде живого графика (ссылку на программу я также дам ниже). Правда программа работает только на 64-битных ОС. Сначала добавим ещё один глобальный буфер

```

char str1[30]={0};
uint8_t buf2[14]={0};

```

Код вывода строки в шину USART пока закомментируем, и напишем код вывода бинарных данных осей. Байты 0x11 и 0x55 вначале отправляем для того, чтобы программа на ПК привязывалась по ним к началу каждой посылки и не путала оси, также убавим немного задержку

```

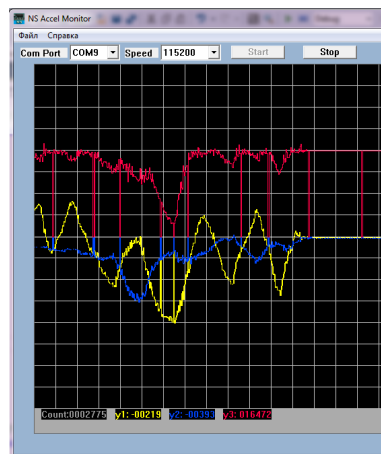
//sprintf(str1,"X:%06d Y:%06d
Z:%06drn", xval, yval, zval);
//USART_TX((uint8_t*)str1,strlen(str1))
;
buf2[0]=0x11;
buf2[1]=0x55;
buf2[2]=(uint8_t)(xval>>8);
buf2[3]=(uint8_t)xval;
buf2[4]=(uint8_t)(yval>>8);
buf2[5]=(uint8_t)yval;
buf2[6]=(uint8_t)(zval>>8);
buf2[7]=(uint8_t)zval;
USART_TX(buf2,8);
_delay_ms(30);

```

Соберём заново код и прошьём контроллер.

Запустим прогамму, настроим также её на наш порт и нажмем Start. Также покрутим немного плату и увидим изменения в графике показаний осей





Мы видим, что показания изредка падают на 0, но это возможно из-за плохих контактов, либо программа просто не успевает обрабатывать, что скорей всего. В текстовом виде у нас никаких сбросов не происходило.

[Предыдущая часть](#)

[Программирование МК AVR](#)

[Следующий урок](#)

Исходный код

Техническая документация:

[Документация на датчик](#)

[Документация на оценочную плату](#)

[Программа Hyper Terminal](#)

[Программа визуализации показаний](#)

Приобрести плату Atmega 328p Pro Mini можно [здесь](#).

Программатор (продавец надёжный) [USBASP USBISP 2.0](#)

Приобрести платы с датчиком LSM6DS3 можно у следующих продавцов:

Надёжный продавец [LSM6DS33 STEVAL-MKI160V1](#)

Здесь дешевле [LSM6DS33 STEVAL-MKI160V1](#)

Здесь другая плата, намного дешевле, но от другого разработчика [LSM6DS33](#)

**Смотреть
ВИДЕОУРОК** (нажмите на картинку)



Post Views: 330

AVR УРОК 39.

Акселерометр

5 комментариев на “AVR УРОК 39. Акселерометр LSM6DS3. Часть 4”



Алексей: STM Урок 51.

Февраль 15, 2017 в 10:42 дп

магнитометр

День добрый!

Очень хочется увидеть урок с OLED дисплеем 0.96" (шина I2C или TWI). Не планируете такой?

[Ответить](#)



admin:

Февраль 15, 2017 в 10:42 дп

К сожалению, у меня такого нет.

[Ответить](#)



Алексей:

Февраль 16, 2017 в 7:40 дп

Понял, ну если будет у Вас возможность и желание попробовать его — буду рад освещению процесса тут. Дисплей интересный — пробовал играть с ним через arduino ide с готовой библиотекой. А вот нормально запустить его просто через atmel studio не вышло. Максимум инициализировать и точки ставить получалось.

[Ответить](#)



admin:

Февраль 16, 2017 в 8:01 дп

Ну да, буду иметь в виду. То что работает с готовыми библиотеками ардуино, всегда нелегко запустить в Atmel Studio. Вот как раз сейчас я этим и занимаюсь. Пишу сценарий для работы с LAN-микросхемой enc28j60 под Atmel Studio, для которой под ардуино есть куча библиотек, приходится в них копать и выкапывать

оттуда некоторый
передовой опыт.

[Ответить](#)



Алексей:

Февраль 16, 2017 в 8:06 дп

LAN модуль — это тоже интересно!
Спасибо за уроки ... очень полезно))

[Ответить](#)

Добавить комментарий

Ваш e-mail не будет опубликован.




Обязательные поля помечены *

Комментарий

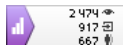
Имя *

E-mail *

Сайт

три +  =  

[Главная](#) | [Новости](#) | [Уроки по программированию МК](#)
| [Программирование микроконтроллеров AVR](#) | [Программирование микроконтроллеров STM32](#)
| [Программирование микроконтроллеров PIC](#) | [Тесты устройств и аксессуаров](#)
| [Устройства и интерфейсы](#) | [Ссылки](#) | [Форум](#) | [Помощь](#)



© 2018 Narod Stream

[Наверх](#)