Измеритель температуры и влажности на базе датчика Silicon Labs Si7005 и дисплейного модуля 4D Systems uLCD-43PT

В статье приведен пример простого приложения для измерения температуры и влажности и рассказано о новом цифровом датчике температуры и влажности компании Silicon Labs Si7005. Производство датчиков Si7005 было начато в 2012 году, но до сих пор они не получили такой известности, как их аналоги от Sensirion и Honeywell. Хотя сейчас эти датчики Silicon Labs имеют самую низкую цену на нашем рынке.
В первой части статьи рассмотрены основные характеристики Si7005. Во второй — показан пример подключения датчика к графическому модулю uLCD-43PT. Этот модуль служит для обработки информации, полученной от датчика, и вывода ее на экран в графическом виде.

Сергей ДОЛГУШИН
dsa@efo.ru

Основные характеристики датчика Si7005

Датчик Si7005 предназначен для измерения температуры и относительной влажности. Он выполнен по КМОП-технологии, а в качестве чувствительного элемента для измерения относительной влажности используется полимерный диэлектрик с низкой диэлектрической постоянной. Оба чувствительных элемента датчика калибруются при изготовлении, поправочные значения хранятся в энергонезависимой памяти датчика (рис. 1).

Датчик Si7005 обеспечивает измерение относительной влажности с типовой погрешностью ±3%RH (рис. 2), точность измерения температуры составляет ±0,5 °С (рис. 3).

Из ближайших аналогичных по параметрам датчиков с интерфейсом IIC компании Sensirion можно назвать датчик SHT11. Он также имеет погрешность измерения относительной влажности ±3%RH в диапазоне от 20 до 80%RH. Погрешность измерения температуры минимальная: ±0,5 °С в точке +25 °С, с линейным ростом до ±2,5 °С в стороны увеличения и уменьшения температуры. При этом стоимость датчика Sensirion SHT11 выше Si7005 не менее чем на 50%. (Для сравнения использованы розничные цены.)

Ближайшим аналогом Si7005 можно также назвать датчик HIH7120 от Honeywell. Этот датчик обладает погрешностью измерения относительной влажности в пределах ±3%RH в диапазоне от 20 до 80%RH. Погрешность измерения температуры составляет ±1 °С в пределах от +5 до +50 °С. Его розничная стоимость на 20–30% выше по сравнению с датчиками Silicon Labs.

Приведенные выше характеристики позволяют сделать вывод, что датчики Si7005

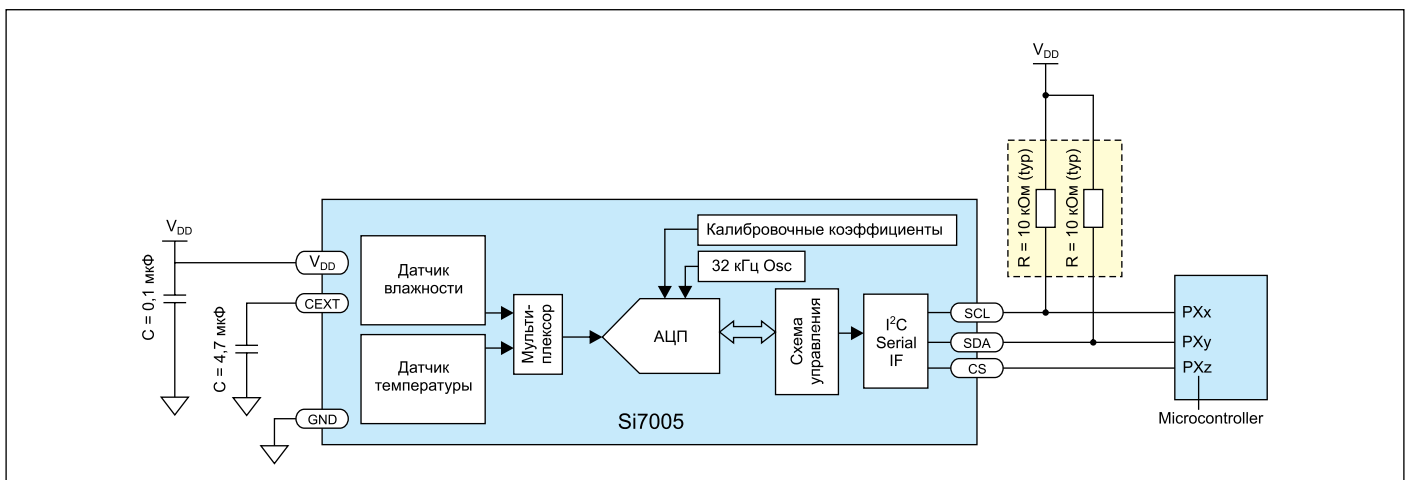

Рис. 1. Функциональная схема Si7005

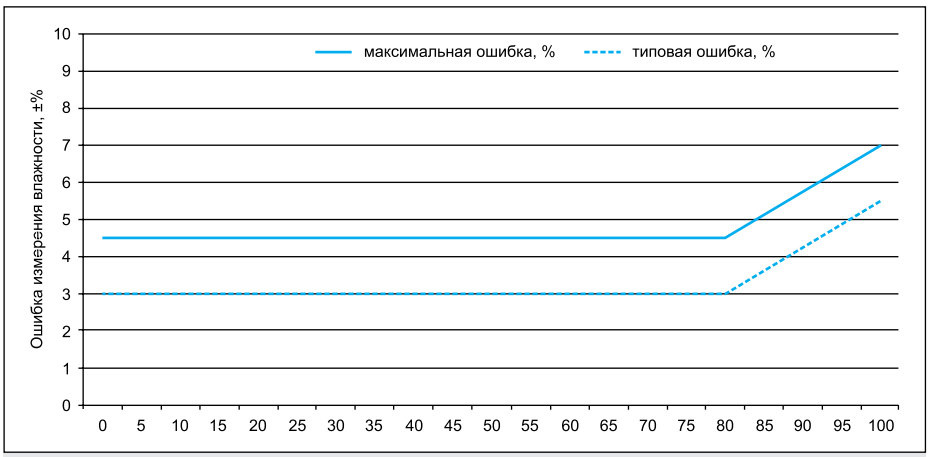

Рис. 2. Изменение ошибки измерения RH в рабочем диапазоне (T = +30 °C)

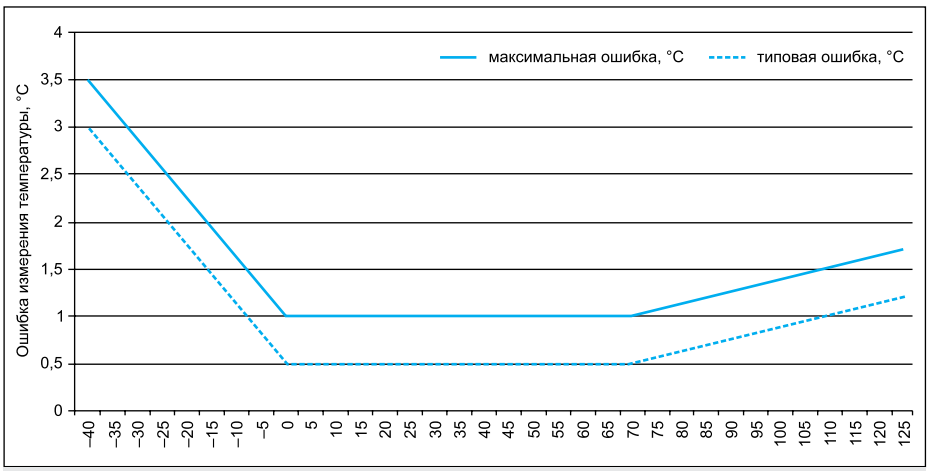

Рис. 3. Изменение ошибки измерения температуры в рабочем диапазоне

заслуживают внимания. Их заявленные характеристики сопоставимы с таковыми у аналогичных конкурентных решений. При этом использование датчиков Silicon Labs позволит снизить себестоимость изделия.

Из дополнительных возможностей датчика Si7005 можно отметить встроенный нагреватель. Его наличие позволяет компенсировать эффект памяти без применения дополнительных внешних элементов. Такой эффект возникает у большинства датчиков относительной влажности при длительном нахождении их в условиях повышенной влажности. Чувствительный элемент накапливает влагу, и показания датчика не соответствуют действительности. Для того чтобы вывести датчик в рабочий режим, его необходимо предварительно просушить. Управление нагревателем осуществляется по интерфейсу IIC.

Кроме нагревателя, для защиты от конденсата, а также от пыли служит опциональный защитный фильтр (рис. 4). Фильтр устанавливается в процессе производства. Он препятствует попаданию влаги и частиц пыли на чувствительный элемент датчика.

Датчик Si7005 выпускается в корпусе QFN-24 с размерами 4×4 мм. Рабочее напряжение питания датчика лежит в пределах от 2,1 до 3,6 В. Максимальное потребление при измерении температуры может достигать 565 мкА, типовое значение — 320 мкА. В режиме измерения влажности максимальное потребление составит 560 мкА, типовое — 240 мкА. В режиме просушки чувствительного элемента максимальное потребление будет равно 31 мА, а типовое — 24 мА.

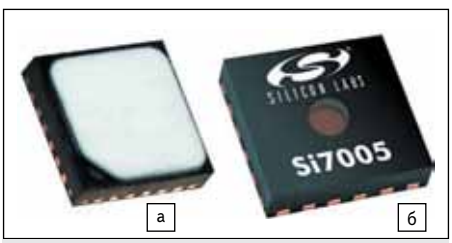

Рис. 4. Датчик Si7005:
а) с установленным фильтром; б) без фильтра

Ознакомившись с основными характеристиками датчика относительной влажности и температуры Si7005 компании Silicon Labs, рассмотрим далее работу с этим датчиком на примере реализации простого устройства для измерения и отображения температуры и относительной влажности.

Работа с датчиком Si7005

Для управления датчиком Si7005 и приема измеренных данных температуры и влажности служит последовательный интерфейс IIC. Интерфейс Si7005 работает в режиме ведомого устройства с адресом 0x40. Передача данных по нему может осуществляться на частоте до 400 кГц.

Для управления датчиком Si7005 и отображения измеренных значений используем графический модуль uLCD-43PT на базе контроллера Picaso компании 4D Systems. Интерфейс IIC контроллера Picaso поддерживает режим ведущего устройства с тремя тактовыми частотами: 1 МГц, 400 и 100 кГц. А графические функции позволят быстро реализовать отображение информации на экране TFT-дисплея модуля uLCD-43PT. Подробнее с функциональными возможностями графических контроллеров и модулей компании 4D Systems можно познакомиться в [2].

Итак, для реализации приложения будут использованы отладочный модуль Si7005EVB-UDP (рис. 5) и графический модуль uLCD-43PT (рис. 6). В таблице 1 указаны номера выводов разъемов обоих модулей,

Рис. 5. Отладочный модуль Si7005EVB-UDP

Рис. 6. Графический модуль uLCD-43PT

Таблица 1. Подключение модулей

	SCL	SDA	3,3 В	GND
Разъем J1 модуля uLCD-43PT	4	6	20	11
Разъем J1 модуля Si7005EVB-UDP	15	14	17	18

Рис. 7. Чтение регистров

которые должны быть соединены. В описываемом примере не используется управление линией CS датчика Si7005. Этот вывод подключен к «земле», и датчик всегда находится в активном состоянии.

Теперь, когда модули подключены друг к другу, можно приступать к созданию нашего приложения. Базовые принципы работы с графическими модулями, средой разработки 4D Workshop и программированием рассмотрены в [2, 3], в этой статье на них останавливаться не будем.

Первым шагом будет реализация функции чтения регистров датчика Si7005. Эта функция будет полезна для проверки подключения и работоспособности Si7005, а также может быть использована для проверки готовности результатов измерений температуры и влажности в процессе работы приложения. Общая последовательность команд для чтения регистров и примеры чтения идентификатора и признака готовности результатов измерений приведены на рис. 7. Белым цветом указаны операции, исполняемые ведущим, серым — ответ ведомого устройства.

Операция чтения регистров начинается с того, что ведущее устройство выставляет признак начала обмена (S). Далее передается байт, содержащий адрес ведомого устройства (Slave) и признак команды записи (W = 0). После получения подтверждения от ведомого устройства (A) мастер передает ведомому адрес регистра, информацию с которого необходимо прочитать. После получения подтверждения (A) мастер формирует признак рестарта (Sr), за которым следует команда начала чтения запрошенного регистра. Эта команда состоит из одного байта, в который входит адрес ведомого и признак чтения (R = 1). Ведомое устройство подтверждает прием команды (A) и передает состояние запрошенного регистра. Мастер подтверждает прием байта статусом **not acknowledge** и завершает обмен статусом завершения обмена (P).

Функция чтения регистров для контроллера Picaso будет выглядеть следующим образом:

```
func readbyte(var address)

   r:=0;
   while (r!=1) // Пример проверки наличия подтверждения от ведомого
   I2C_Start(); // Выставляем признак начала обмена
   r := I2C_Write(0x80); // Передаем адрес устройства 0x40 (старшие биты) и признак записи 0x00 (младший бит). Функция записи возвращает статус ведомого — ACK, Not ACK. Статус после последнего такта может быть получен функцией I2C_AckStatus()
   wend
   r := I2C_Write(address);
   I2C_Restart(); // Устанавливаем признак рестарта
   r := I2C_Write(0x81); // Передаем адрес устройства 0x40 (старшие биты) и признак чтения 0x01 (младший бит)
   r := I2C_Read(); // Чтение регистра
   I2C_Nack(); // После приема данных устанавливаем признак Not Ack
   I2C_Stop();// Окончание цикла чтения
   return r;

endfunc
```

Рис. 8. Запись в регистры

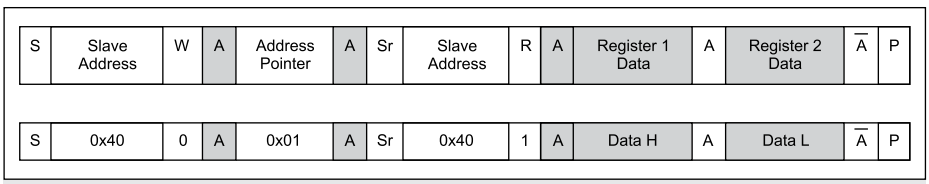

Рис. 9. Чтение результатов измерений

В основной программе эта функция может быть вызвана следующим образом:

```
func startconversion(var reg)
   r:=0;

   I2C_Start(); // Установка признака начала обмена
   r := I2C_Write(0x80); // Передаем адрес устройства 0x40 (старшие биты) и признак записи 0x00 (младший бит)
   r := I2C_Write(0x03); // Передаем адрес конечного регистра
   r := I2C_Write(reg); // Запись команды в регистр
   I2C_Stop(); // Установка признака конца обмена

endfunc
```

Откомпилируем и загрузим программу в модуль, в результате ее выполнения на экран будет выведено значение идентификатора датчика Si7005, равное b01010000.

Следующим этапом будет реализация последовательности записи в управляющий регистр Si7005. Управляющий регистр у Si7005 один, он имеет адрес 0x03, и через него устанавливаются все режимы работы датчика (табл. 2, 3).

Таблица 2. Функции регистра управления

Бит	7	6	5	4	3	2	1	0
Функция			Режим измерений				Подогрев	Начало измерений

Таблица 3. Значения функций регистра управления

Бит	Функция	Режим
7:6	Зарезервировано	Значение не определено
5	Режим измерений	0—35 мс 1—18 мс
4	Температура/влажность	0 — измерение влажности 1 — измерение температуры
3:2	Зарезервировано	Значение не определено
1	Подогрев	0 — выключен 1 — включен
0	Начало измерений	0 — нет измерения 1 — запуск измерения

Последовательность команд для записи в регистры датчика и примеры команд запуска измерений температуры и влажности приведены на рис. 8. Функция для контроллера Picaso будет выглядеть следующим образом:

```
func startconversion (var reg)
r:=0;
I2C_Start ();//Установка признака начала обмена
r:= I2C_Write (0x80);//Передаем адрес устройства 0x40 (старшие биты) и признак записи 0x00 (младший бит)
r:= I2C_Write (0x03);//Передаем адрес конечного регистра
r:= I2C_Write (reg);//Запись команды в регистр
I2C_Stop ();//Установка признака конца обмена
endfunc
```

Последней функцией для организации обмена по I^2C будет чтение результатов измерений температуры и влажности.

Таблица 4. Форматы данных

	Значение	
	Стандартный режим	Быстрый режим
Время измерения, мс	35	18
Разрешение по температуре, бит	14	13
Разрешение по относительной влажности, бит	12	11

Последовательность обмена между ведущим и ведомым устройством приведена на рис. 9. Данные измерений передаются в форматах, указанных в таблице 4.

Функция чтения результатов измерений будет выглядеть так:

```
func readword(var address)
   r:=0;
   c:=0;

   I2C_Start(); // Установка признака начала обмена
   r := I2C_Write(0x80); // Передаем адрес устройства 0x40 (старшие биты) и признак записи 0x00
   r := I2C_Write(address); // Передаем адрес регистра с результатами измерений
   I2C_Restart();
   r := I2C_Write(0x81); // Передаем адрес устройства 0x40 (старшие биты) и признак чтения 0x01 (младший бит)
   r := I2C_Read(); // Чтение старшего байта
   I2C_Ack(); // Подтверждение приема
   c := I2C_Read(); // Чтение младшего байта
   r:=r<<8;
   c:=(r|c);
   I2C_Nack(); // Устанавливаем статус Not Ack
   I2C_Stop(); // Завершаем обмен
   return c; // Возвращаем результат измерений

endfunc
```

Итак, управление датчиком Si7005 и чтение результатов измерений реализованы. Полученные результаты необходимо вывести в графической форме на дисплей. Для этого используем следующие графические элементы, доступные в среде разработки 4D Workshop: Thermometer, Led Digits и Static Text. Первый элемент будет отображать информацию в графическом виде, второй — в цифровом, последний — информационные надписи (рис. 10). Настройки каждого элемента можно посмотреть, скачав архив проекта с сайта [4]. Размещаем выбранные элементы на форме, отображающей выбранный в проекте дисплей, задаем параметры и генерируем исполняемый код для вызова графических объектов в приложении [3].

Рис. 10. Внешний вид приложения

Перед выводом измеренных значений температуры и влажности на экран эти данные должны быть преобразованы в привычный формат представления температуры — °C и относительной влажности — %RH.

Расчет значения температуры и вывод его на экран будут выглядеть следующим образом:

```
startconversion(HUM); // Команда начала измерения температуры
pause(50);
while (id!=0)
id:=readbyte(0x00); // Чтение регистра готовности
wend

c:=readword(0x01); // Чтение значения измеренной температуры

Temperature := ((c>>2)*10)/32 - 500; // Измеренное значение температуры представлено 14-разрядным значением. Датчик передает в контроллер 16-разрядное слово, 2 младших бита которого равны 0 и являются незначащими. Поэтому вначале выполняем операцию сдвига на 2 разряда, чтобы отбросить 2 незначащих бита. Исходная формула преобразования температуры в °C: Temperature := (c>>2)/32–50. Контроллер Picaso работает только с целыми числами, разрядность операндов и результатов не должна превышать 16. Для вывода значения температуры с десятыми долями необходимо умножить полученное значение на 10. Цифровой индикатор Led Digits выведет на экран значение в требуемом формате, с десятыми долями
img_SetWord(hndl, iThermometer1, IMAGE_INDEX, Temperature/10); // Вывод полученного значения в графическом виде. Для графического представления достаточно вычисленного целого значения, поэтому делим значение температуры на 10
img_Show(hndl,iThermometer1);
ledDigitsDisplay(Temperature, iLeddigits1+1, 108, 4, 2, 34, 0); // Вывод значения температуры в цифровом виде
```

Расчет значения относительной влажности и вывод его на экран будут выглядеть так:

```
startconversion(TEM); // Запуск измерения влажности
pause(40);
while (id!=0)
id:=readbyte(0x00); // Чтение регистра готовности
wend
c:=readword(0x01); // Чтение значения измеренной влажности

Humidity:=((c>>4))/16-24; // Аналогичное преобразование значения влажности

Linear := (Humidity *256 - Humidity *A1 - (Humidity * Humidity *A2) - A0) / 256; // Линеаризация измеренных значений влажности. Коэффициент 256 введен для работы в разрядной сетке контроллера Picaso, коэффициенты коррекции A0—A2 указаны производителем в документации на датчик

img_SetWord(hndl, iThermometer2, IMAGE_INDEX, Linear); // Вывод полученного значения в графическом виде
img_Show(hndl,iThermometer2);

ledDigitsDisplay(Linear, iLeddigits2+1, 368, 3, 1, 36, 0); // Вывод значения температуры в цифровом виде
```

Итак, приложение готово к работе, остается только откомпилировать и загрузить его в графический модуль uLCD-43PT или любой другой на базе контроллера Picaso. Это простое приложение демонстрирует работу датчика влажности и температуры Si7005, на текущий момент — одного из самых недорогих и простых в работе.

Литература

1. Si7005 Digital I²C humidity and temperature sensor. Datasheet.
2. Долгушин С. Графические контроллеры и дисплейные модули компании 4D Systems // Компоненты и технологии. 2013. № 2.
3. Долгушин С. Графический интерфейс пользователя на базе готовых дисплейных модулей компании 4D Systems // Компоненты и технологии. 2013. № 4.
4. http://mymcu.ru/content/files/ftdi/silabs.rar