

Revisiones	Fecha	Comentarios
0	03/06/05	
1	06/03/07	ampliación, soporte Ramtron

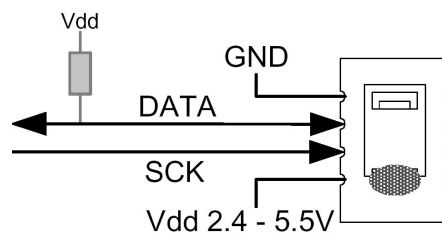
Si bien existen numerosas alternativas para la medición de temperatura ambiente, la medición de la humedad relativa ambiente resulta ser algo complicada. En ambos casos, lograr una determinada precisión implica disponer de sensores caros y mucho cuidado en la sección analógica y la placa de circuito impreso; hecho que se ve potenciado si el sensor debe estar a una cierta distancia del circuito de procesamiento, y/o en ambientes hostiles.

Les presentamos en esta oportunidad unos sensores combinados de humedad y temperatura ambiente: SHT-71, desarrollados por la firma Sensirion. Los mismos constan de un par de sensores y convertidores A/D, circuitería de calibración, y compensación. La información es presentada por una interfaz serie, de modo que el sistema de medición se desentiende del tema ruidos, linealización analógica, calibración, e impedancias. La información de temperatura se obtiene en 12 ó 14-bits en un rango de -40°C a +120°C, con una precisión (mejor caso) de 0.4°C. La información de humedad relativa ambiente porcentual se obtiene en 8 ó 12-bits, con una precisión del 3% (mejor caso). Los sensores funcionan tanto a 3,3V como a 5V, y su estabilidad a largo plazo viene garantizada por la calibración realizada internamente, a cada medición. Para aquellos preocupados porque el mismo ruido o interferencias electromagnéticas puedan alterar el valor de los datos, el mensaje que los lleva incluye un CRC (Cyclic Redundancy Check).



Hardware

La conexión es sumamente simple, requiere dos pines de I/O, uno con capacidad tristate u open collector para los datos; el otro entrega el clock que controla el timing de esos datos, para ambos sentidos de la comunicación



Software

El fabricante nos provee código en C como para ser compilado en un procesador compatible MCS51. Sin demasiado esfuerzo, es posible portarlo a cualquier otra arquitectura, tarea la cual hemos desarrollado para Rabbit y para VRS51L3074 de Ramtron, que si bien es MCS51 compatible, presenta algunas diferencias en cuanto a sus I/O.

Rabbit

El esquema propuesto para Rabbit es clásico y aprovecha sus *costates* y *cofunctions* para evitar ciclos de espera. En el archivo adjunto encontrará el código.

Ramtron

El esquema propuesto para Ramtron parte una de las funciones provistas por el fabricante de modo de aprovechar la CPU en los ciclos de espera, dado que desde que se inicia la medición hasta que el sensor está listo para entregar los datos, pueden transcurrir hasta más de 200ms:

```
char s_measure_start(unsigned char mode)
{
    unsigned char error=0;

    s_transstart();                // transmission start
    switch(mode){                  // envía comando
    case TEMP: error+=s_write_byte(MEASURE_TEMP); break;
    case HUMI: error+=s_write_byte(MEASURE_HUMI); break;
    default: break;
    }
    return(error);                 // el sensor queda trabajando
}

char s_measure_get(unsigned char *p_value, unsigned char *p_checksum)
{
    if(DATA==0) {                 // terminó ?
        *(p_value+1)=s_read_byte(ACK); // lee primer byte (MSB)
        *(p_value)=s_read_byte(ACK); // lee segundo byte (LSB)
        *p_checksum =s_read_byte(noACK); // lee checksum
        return(0);
    }
    return(1);                    // no
}
}
```

La forma de realizar una medición entonces, es la siguiente

```
if(!(error=s_measure_start(TEMP))) { // inicia medición de temperatura
    if(!s_measure_get((unsigned char*) &temp_val.i,&checksum)) {
        // procesa medición
    }
    else {
        // espera, maneja timeout
    }
}
}
```

Un detalle a tener en cuenta es que la respuesta del sensor de humedad es ligeramente cuadrática; esto significa que para obtener la máxima precisión, es necesario realizar una operación de corrección muy simple del tipo $y=ax^2+bx+c$, donde x es el valor medido e y corresponde al valor corregido. De igual modo, si la temperatura se aleja bastante de 25°C, es aconsejable realizar una segunda corrección para compensar la dependencia del sensor respecto a la temperatura, en este caso lineal. Por este motivo, y por el hecho de que regularmente no se toman mediciones con mayor frecuencia que una vez por segundo, es que hemos decidido mantener el código en C para esta nota de aplicación, tanto para Rabbit como para Ramtron. En este último caso, para minimizar la necesidad de uso de coma flotante, hemos partido las funciones en dos archivos: *SHTxx.c* y *SHTxx_highlevel.c*. El primero tiene todas las funciones de soporte necesarias para realizar una medición y poder reportar a un sistema de mayor jerarquía, o comparar con límites prefijados por éste, sin emplear coma flotante. De ser necesario un procesamiento de los datos, el usuario puede utilizar las funciones contenidas en *SHTxx_highlevel.c*, que lo hacen utilizando coma flotante, o escribir su propio código con aritmética entera, tomando éste como guía. Dichas funciones son:

```
calc_sth11(&humi_val.f,&temp_val.f); // procesa y corrige
dew_point=calc_dewpoint(humi_val.f,temp_val.f); //calcula punto de condensación
```

En el archivo adjunto encontrará el código original, el port para Rabbit, y el port para Ramtron VRS51L3074, con un ejemplo de uso de las funciones.

Holtek, Microchip, otros micros

Por supuesto que también hemos portado a assembler a otros micros de nuestra cartera de productos, como PIC y Holtek. El código se encuentra en otras notas de aplicación, como por ejemplo CAN-047 y CAN-057, respectivamente.

CAN-028, Sensores de Humedad y Temperatura SHT-71

Para aquellos que deseen escribir código para Rabbit o Ramtron en assembler, o portar a otro micro, pueden utilizar el C como guía o tomar la información de la hoja de datos del SHT-71. En el caso de C, debe tenerse cuidado con la forma de almacenar enteros, dado que el código original (provisto por el fabricante) es big-endian, y tanto Ramtron con SDCC como Rabbit son little-endian.

La interfaz de comunicación es de 2-hilos, similar pero **no compatible con I²C, la relación clock-data corresponde al gráfico siguiente:**

