HTU21D Xplained Pro Digital Humidity and Temperature Sensor

- **Measures relative humidity from 0% to 100%**
- **Measures temperature from -40°C to +125°C**
- **I²C communication**
- **Fully calibrated**
- **Fast response time**
- **Selectable resolution**
- **Very low power consumption**

GENERAL DESCRIPTION

The HTU21D Xplained Pro provides the necessary hardware to interface the HTU21D digital relative humidity and temperature sensor to any system that utilizes Atmel Xplained Pro compatible expansion ports configurable for I²C communication. The HTU21D sensor is a self-contained humidity and temperature sensor that is fully calibrated during manufacture. The sensor can operate from 1.5V to 3.6V, has selectable resolution, low battery detection and checksum capability. The HTU21D also has a low power stand-by mode for power-sensitive applications.

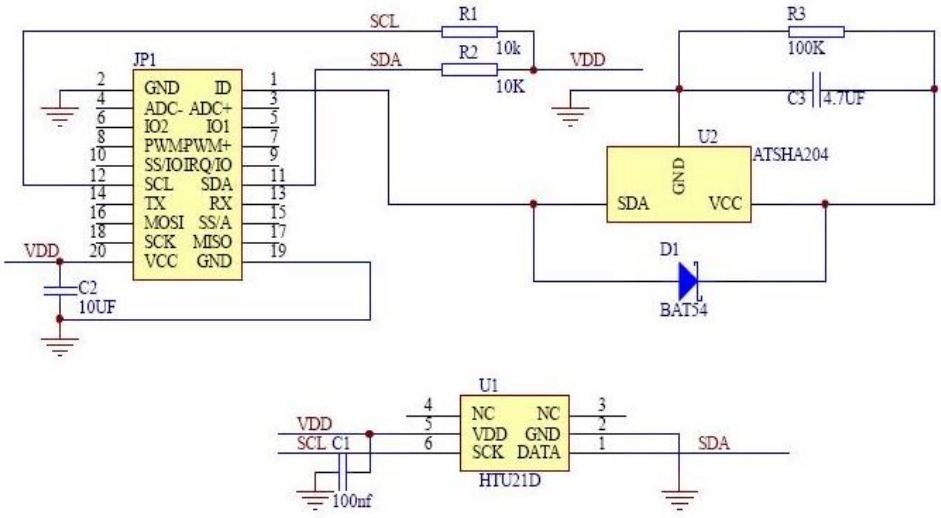
FEATURES

- 20-pin Xplained Pro compatible connector
- I²C interface
- Xplained Pro hardware identification chip
- Atmel Studio 6 Project available for download
- µC C code available for download
- Selectable 8-12 bit resolution for humidity
- Selectable 12-14 bit resolution for temperature

PERFORMANCE

- 0% to 100% relative humidity range
- -40°C to +125°C temperature range
- Very low power consumption
- Operates from 1.5V to 3.6V
- Fast response time – 5 seconds typical
- Built-in heater for fast recovery from saturation
- Recovers fully from condensation
- Fast conversion time – 14ms typical

SCHEMATIC



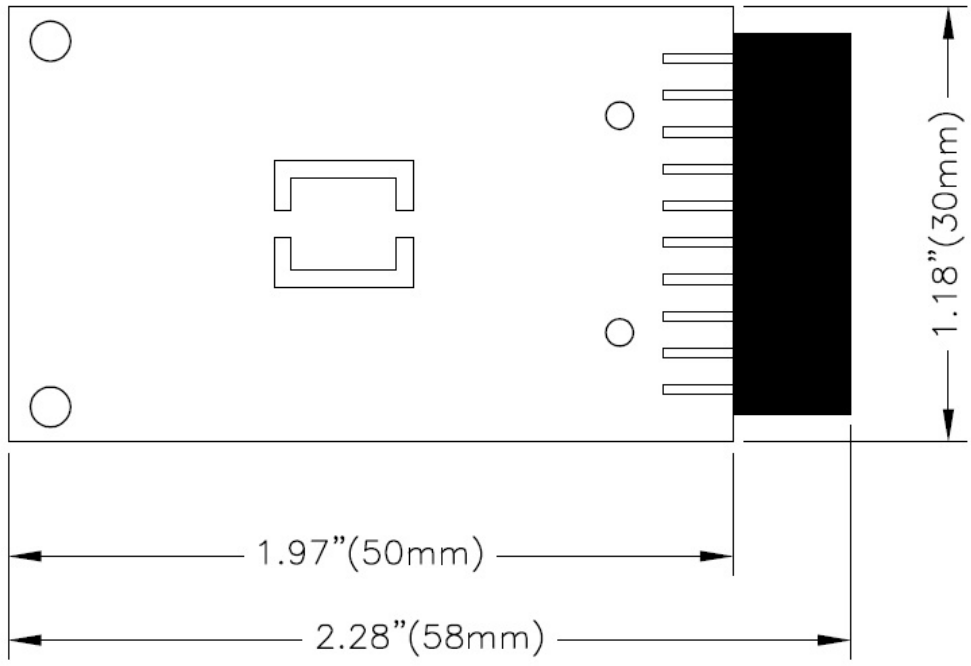
CONNECTOR PIN ASSIGNMENTS (I^2C COMMUNICATIONS)

SYSTEM PLUG

Connector JP1					
Pin No.	Signal	Description	Pin No.	Signal	Description
1	ID	Hardware identification	11	SDA	TWI Serial Data
2	GND	Ground	12	SCL	TWI Serial Clock
3	N/C	Not Connected	13	N/C	Not Connected
4	N/C	Not Connected	14	N/C	Not Connected
5	N/C	Not Connected	15	N/C	Not Connected
6	N/C	Not Connected	16	N/C	Not Connected
7	N/C	Not Connected	17	N/C	Not Connected
8	N/C	Not Connected	18	N/C	Not Connected
9	N/C	Not Connected	19	GND	Ground
10	N/C	Not Connected	20	Vdd	Power Supply

Table 1

DIMENSIONS [mm]



DETAILED DESCRIPTION

I²C Interface

The peripheral module can interface to the host being plugged directly into an Xplained Pro extension port (configured for I²C) through connector JP1.

External Control Signals

The IC operates as an I²C slave using the standard 2 wire I²C connection scheme. The IC is controlled either by the host (through the Xplained Pro connector). In cases where one or more of the SCL and SDA signals are driven from an external source, 10k resistors R1, R2 provide pull-up. However, this also increases the apparent load to the external driving source. If the external source is not capable of driving these loads (10k), they should be removed.

DRIVERS & SOFTWARE

Detailed example software and drivers are available that execute directly without modification on a number of development boards that support an integrated or synthesized microprocessor. The download contains several source files intended to accelerate customer evaluation and design. The source code is written in standard ANSI C format, and all development documentation including theory/operation, register description, and function prototypes are documented in the interface file.

Functions Summary

Enumerations

enum	**htu21_i2c_master_mode** {
	htu21_i2c_hold,
	htu21_i2c_no_hold
	}

enum	**htu21_status** {
	htu21_status_ok, **htu21_status_no_i2c_acknowledge**,
	htu21_status_i2c_transfer_error, **htu21_status_crc_error**
	}

enum	**htu21_resolution** {
	htu21_resolution_t_14b_rh_12b = 0, **htu21_resolution_t_12b_rh_8b**,
	htu21_resolution_t_13b_rh_10b, **htu21_resolution_t_11b_rh_11b**
	}

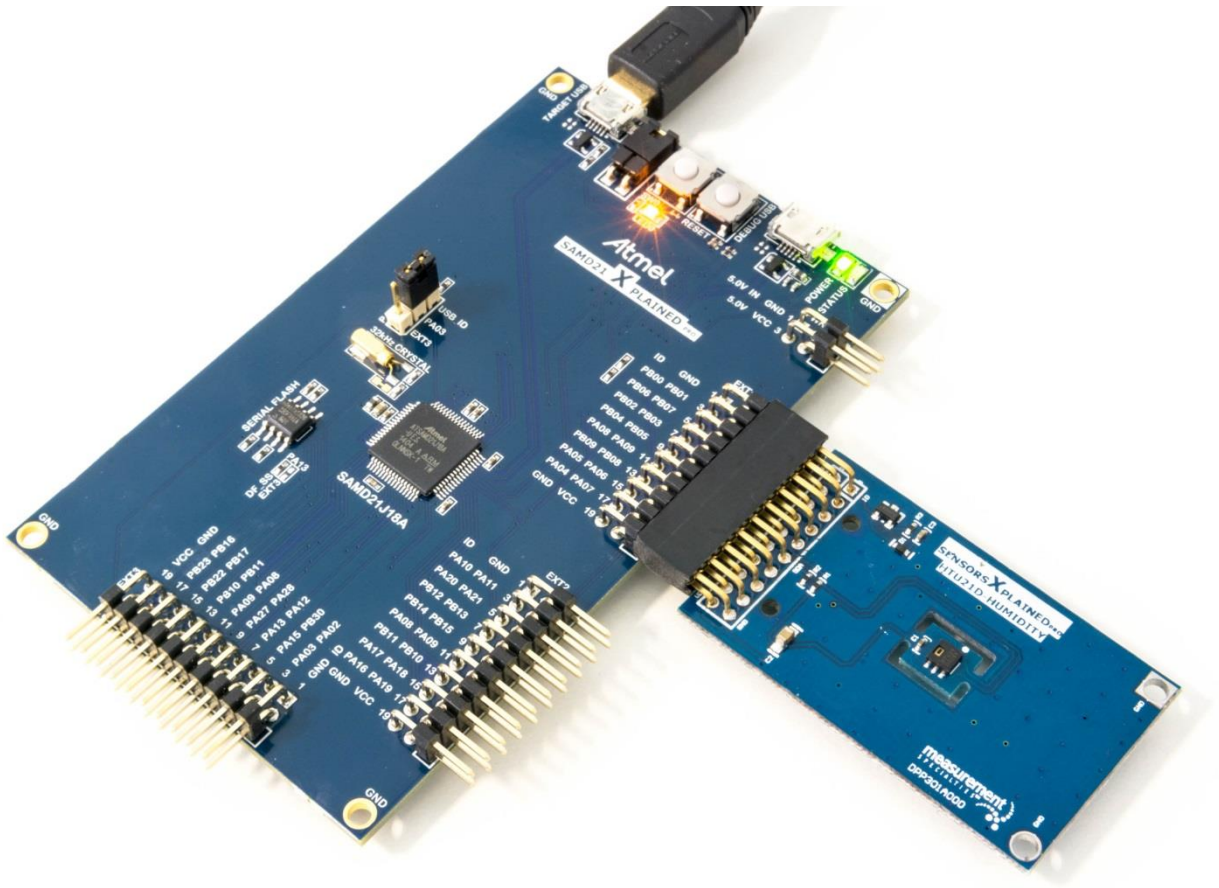
enum	**htu21_battery_status** { **htu21_battery_ok**, **htu21_battery_low** }
enum	**htu21_heater_status** { **htu21_heater_off**, **htu21_heater_on** }

Functions

void	**htu21_init** (void) Configures the SERCOM I2C master to be used with the HTU21 device.
bool	**htu21_is_connected** (void) Reset the HTU21 device.
enum htu21_status	**htu21_reset** (void) Reset the HTU21 device.
enum htu21_status	**htu21_read_serial_number** (uint64_t *) Reads the htu21 serial number.
enum htu21_status	**htu21_set_resolution** (enum htu21_resolution) Set temperature and humidity ADC resolution.
void	**htu21_set_i2c_master_mode** (enum htu21_i2c_master_mode) Set I2C master mode. This determines whether the program will hold while ADC is accessed or will wait some time.
enum htu21_status	**htu21_read_temperature_and_relative_humidity** (float *, float *) Reads the relative humidity value.
enum htu21_status	**htu21_get_battery_status** (enum htu21_battery_status *) Provide battery status.
enum htu21_status	**htu21_enable_heater** (void) Enable heater.
enum htu21_status	**htu21_disable_heater** (void) Disable heater.
enum htu21_status	**htu21_get_heater_status** (enum htu21_heater_status *) Get heater status.
float	**htu21_compute_compensated_humidity** (float, float) Returns result of compensated humidity.
float	**htu21_compute_dew_point** (float, float) Returns the computed dew point.

Project Setup

This project is based on ATSAMD20J18 board with Measurement Specialties Xplained Pro extension board connected to EXT1 pad as shown on figure below.



Running the Application

1. Download the HTU21D Xplained Pro example package here.
2. Decompress the archive file
3. Open the .cproj project file with Atmel Studio 6
4. You will now be able to build the HTU21D example project -
5. Finally, run the build result on your Atmel Xplained Pro board -

Application Code

This section is intended to provide a basic example of functionality.

```
/**
 * \file main.c
 *
 * \brief HTU21 Temperature & Humidity monitoring application file
 *
 * Copyright (c) 2014 Measurement Specialties. All rights reserved.
 *
 */

#include <asf.h>

uint64_t serial;
float temperature;
float relative_humidity;
float compensated_humidity;
float dew_point;
enum htu21_heater_status heater;

int main (void)
{
		enum htu21_status status;
		float last_temperature = 0;
		float variation = 0;
		uint8_t n=0;

		system_init();
		delay_init();

		// Configure device and enable
		htu21_init();

		if( !htu21_is_connected() )
			return -1;

		// Reset HTU21
		status = htu21_reset();
		if( status != htu21_status_ok)
			return -1;

		// Read serial number
		status = htu21_read_serial_number(&serial);
		if( status != htu21_status_ok)
			return -1;

		// Configure resolution
		status = htu21_set_resolution(htu21_resolution_t_12b_rh_8b);
		if( status != htu21_status_ok)
			return -1;

		// Monitor temperature every 500ms
		while (1) {

			// Enable heater for 10s
			if( (n==10) ) {
				status = htu21_enable_heater();
				if( status != htu21_status_ok)
					return -1;
			}
			// Disable heater after 20s
			if( (n==20) ) {
				status = htu21_disable_heater();
				if( status != htu21_status_ok)
					return -1;
			}

			// Check heater status
			status = htu21_get_heater_status(&heater);
			if( status != htu21_status_ok)
				return -1;

			// Alternate between w and w/o hold temperature read
			if( n&1 )
				htu21_set_i2c_master_mode(htu21_i2c_no_hold);
			else
				htu21_set_i2c_master_mode(htu21_i2c_hold);

			status = htu21_read_temperature_and_relative_humidity(&temperature, &relative_humidity);
			if( status != htu21_status_ok)
				return -1;

			compensated_humidity = htu21_compute_compensated_humidity(temperature,relative_humidity);
			dew_point = htu21_compute_dew_point(temperature,relative_humidity);

			variation += temperature - last_temperature;

			// Look for significant temperature variation
			if (  variation >= 0.5 ) {
				// Yes, so turn LED on.
				port_pin_set_output_level(LED_0_PIN, LED_0_ACTIVE);
```

```
                                variation = 0;
                                } else if ( variation <= -0.5 ) {
                                // No, so turn LED off.
                                port_pin_set_output_level(LED_0_PIN, LED_0_INACTIVE);
                                variation = 0;
                        }

                        delay_ms(500);
                        last_temperature = temperature;
                        n++;
                        if(n==50) n=0;
                }

                return 0;
}
```

Disclaimer

REFERENCE MATERIALS

Detailed information regarding operation of the IC:
HTU21D Datasheet

Detailed information regarding SAMD2x Driver:
HTU21D SAMD2x Driver

Complete software sensor evaluation kit for Xplained Pro:
HTU21D SAMD2x Software

Overview of Atmel's Sensor Hub:
Atmel Sensor Hub

Measurement Specialties products on Atmel's Sensor Hub:
Atmel MEAS Sensor Hub Products

ORDERING INFORMATION

DPP301A000 : HTU21D XPLAINED PRO BOARD

Customer Service

MEAS France
Impasse Jeanne Benozzi
CS 83 163
31027 Toulouse Cedex 3
FRANCE
Tel:+33 (0)5 820.822.02
Fax:+33 (0)5.820.821.51
Sales: sales.tlse.fr@meas-spec.com