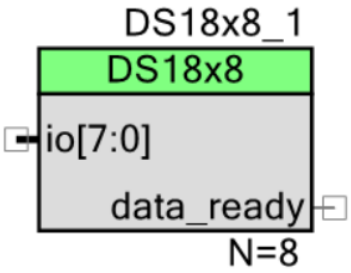
DS18x8 Digital Temperature Sensor
0.0

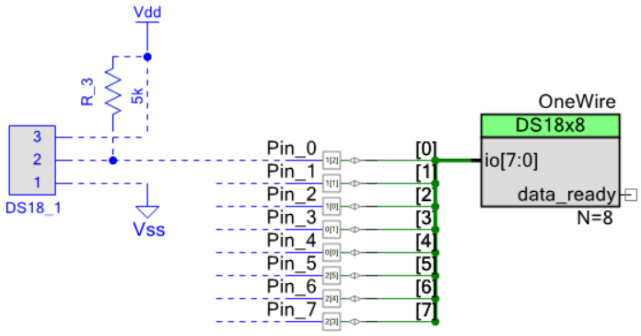
Features



- Implements 1-Wire communication protocol to read Maxim Integrated DS18B20 digital temperature sensor.
- Simultaneously samples data from up to 8 sensors.
- Temperature resolution 12 bit (better than 0.1°C)
- Returns temperature in float, integer and string data formats.

General description

The DS18x8 component provides 1-wire communication with Maxim DS18B20 digital temperature sensors [Ref. 1] and allows synchronous sampling temperatures from up to 8 sensors with resolution better than 0.1°C. Component provides API functions that return temperature value and sensor presence state.



When to use DS18x8 component

Component is developed for use in distributed temperature-sensing applications where multiple sensors must be sampled at the same time. Typical applications for this component are: testing heat load at multiple points in electronic system, HVAC environmental control with several comfort zones, checking temperature uniformity in the refrigeration chamber, TEC temperature control.

Input-output connections

The component has up to 8 digital bidirectional terminals for communication with temperature sensors and one interrupt output.

io[N:0] – Input/Output (bidirectional)

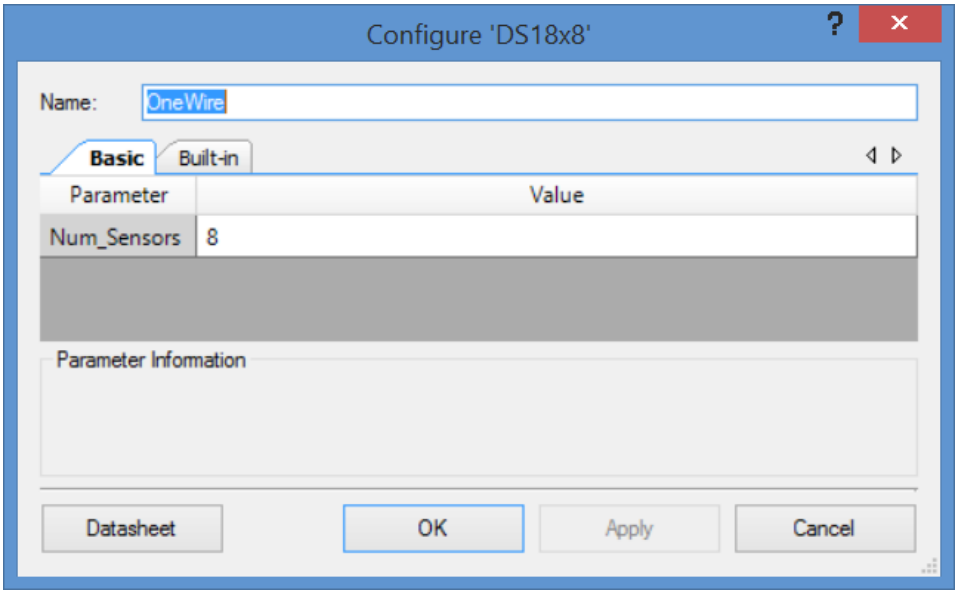
This input sweeps the individual 1-Wire communication terminals into a single bus terminal. This pin is always visible. N-is the number of sensors - 1. This input should be directly connected to the external bidirectional pins, and pins number should correspond to number of sensors declared in the configuration dialog. Note that PSoC allows only 4 bidirectional pins per port, so 8 sensors will occupy two ports.

data_ready – Output

This pin provides interrupt source when temperature conversion has been completed and sensors are ready to be parsed for temperature data. It is driven by hardware timer inside the component. This pin is always shown on the symbol. The pin doesn't have to be connected.

Parameters and Settings

Basic dialog provides following parameters and settings:



Num_Sensors

The Num_Sensors parameter corresponds to number of sensors attached to the component. Valid range is 1 through 8.

Application Programming Interface

Function	Description
`DS18x8_Start()`	Enables component
`DS18x8_Stop()`	Disables component
`DS18x8_CheckPresence()`	Function to check sensors presence
`DS18x8_SendTemperatureRequest()`	Function to initialize temperature measurement and data conversion by ADC in the digital sensor.
`DS18x8_ReadTemperature()`	Function to read temperature data from sensor memory.
`DS18x8_BusState()`	Bus state: bit 1-sensor present, bit 0-no sensor found.
`DS18x8_GetTemperatureAsFloat (uint8 index)`	Function to convert temperature from encoded values to degrees °C, returned in float format.
`DS18x8_GetTemperatureAsInt100 (int16 index)`	Function to convert temperature from encoded values to degrees °C x100, returned in integer format.
`DS18x8_GetTemperatureAsString (uint8 index);`	Function to convert temperature from encoded values to degrees °C, returned in string format.
`CYBIT DS18x8_DataReady;`	Semaphore flag, indicates that ADC conversion complete and temperature data can be retrieved from thermometer

void DS18x8_Start ()

Description: Initializes the component.

Parameters: void.

Return Value: void.

void DS18x8_Stop ()

Description: Stops the component.

Parameters: void.

Return Value: void.

void DS18x8_CheckPresence (void)

Description: Function to check sensors presence.

Parameters: void.

Return Value: Returns sensor presence state as 8-bit mask, where bit 1 corresponds to sensor present and bit 0 if sensor absent.

void DS18x8_SendTemperatureRequest (void)

Description: Asynchronous call to start temperature measurement. Function clears the bus and begins digitizing temperatures in all sensors attached.

Parameters: void.

Return Value: Returns sensor presence state as 8-bit mask, where bit 1 corresponds to sensor present and bit 0 if sensor absent.

uint8 DS18x8_ReadTemperature (void)

Description: Function to retrieve temperature data from sensors memory. In the result of this operation temperature values will be obtained in encoded format, altogether with sensor presence state. The result is stored in data array of the structure

```
DS18x8_Sensor[index].Temperature;
DS18x8_Sensor[index].present;
```

Parameters: void.

Return Value: Returns sensor presence state as 8-bit mask, where bit 1 corresponds to sensor present and bit 0-to sensor absent.

float DS18x8_GetTemperatureAsFloat (uint8 index)

Description: Function to convert sensor temperature reading from the encoded values to degrees °C.

Parameters: index: sensor index, valid range from 0 to (Num_Sensors-1).

Return Value: Returns sensor temperature measured in °C in float format. If sensor not present then out of limit value -256.0 is returned.

int16 DS18x8_GetTemperatureAsInt100 (uint8 index)

Description: Function to convert sensor temperature reading from the encoded values to degrees °C multiplied by 100 and rounded to nearest integer value.

Parameters: index: sensor index, valid range from 0 to (Num_Sensors-1).

Return Value: Returns sensor temperature measured in °C, multiplied by 100 in int16 format. For example 16.125°C will be converted into value 1613. If sensor not present then value -25600 is returned.

char* DS18x8_GetTemperatureAsString (uint8 index)

Description: Function to convert sensor temperature reading from the encoded values to degrees °C, returned in string format. This function provided as high-speed alternative to the standard output formatting function sprint(), and provides 10x performance improvement and code size reduction (“stdlib.h” not required).

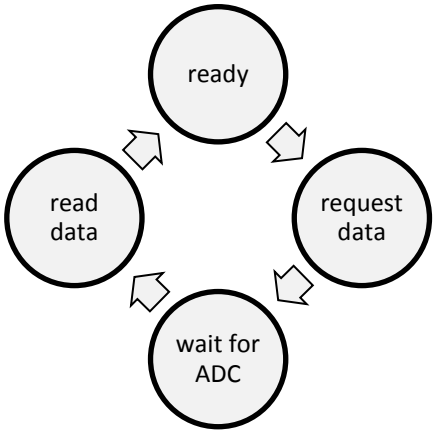
Parameters: index: sensor index, valid range from 0 to (Num_Sensors-1).

Return Value: Returns sensor temperature measured in °C, returned as string of type “%.2” format. For example temperature 16.125°C will be converted into string “16.13”. If sensor is not present, the return string will be “-“.

Functional Description

The DS18B20 digital temperature sensor provides temperature reading in range from -55°C to +125°C with up to 12-bit resolution (1bit = 0.0625°C). Communication with sensor provided through 1-Wire protocol [Ref. 2-4]. The process of measurement is asynchronous and requires several steps to complete. After receiving a request for a temperature from microcontroller, sensor ADC begins to convert thermistor reading and stores result in scratchpad memory for later retrieval. It takes sensor at least 750 ms to convert data with 12-bit precision [Ref. 1]. After conversion completed the temperature data can be retrieved by reading it byte-by-byte using 1-Wire protocol.

Asynchronous operation of the DS18x8 component implemented as a state machine using both software and hardware. The 1-Wire communication protocol is implemented in software, while long 750ms delay implemented using non-blocking hardware timer. Temperature reading is initiated by the `SendTemperatureRequest()` function, which clears bus, starts temperature conversion and triggers one-shot Timer with delay time of 750 ms. The time delay is set to ensure complete conversion of the data by ADC. The time-delay is non-blocking, so that CPU is allowed to perform other tasks. On timer expiration, the "data_ready" interrupt trigger is issued, flag `_DataReady` is set, and CPU can retrieve data from sensor using `ReadTemperature()` function. The data extracted from the sensor are encoded in proprietary format [Ref. 1], and can be converted into degrees °C using several "helper" functions provided in component API.



The DS18x8 component is currently designed to operate only at 12-bit resolution for few reasons: default resolution of the DS18B20 sensor on power-up is 12-bit, and many practical applications require temperature reading with precision of 0.1°C, which requires 12-bit resolution.

Some practical notes on using DS18B20 sensor: sample requests to the sensor can't be too frequent; give ADC enough time to complete conversion before the next request. The ADC conversion time 750 ms limits sensor sampling highest rate to about 1 sample/sec. As another drawback, as at this sample rate sensor self-heating causes temperature readout elevation to

about 0.1°C above the steady value. Keep in mind that DS18B20 sensor body thermal equilibration time is about 3-5 sec, which renders faster temperature polling meaningless.

Lastly, the `ReadTemperature()` function returns temperature samples which took place about 750 ms prior to the call (actual sampling event occurs at point B on timing diagram, Figure 2), which may be important for data synchronization in some applications .

Features not implemented in current version of the component:

- Communicating to multiple sensors on single line.
- Powering sensor from data line (“parasit power”).
- Programmable resolution (9 to 12 bits).
- Polling sensor’s unique 64-bit serial code.
- Alarm signaling.
- CRC check.

Sample Firmware Source Code

Two demo projects for PioneerBoard (“DS18x8_demo” and “DS18x8_demo2”) are provided together with the component. In DS18x8_demo project, all sensors are simultaneously sampled at 4 seconds interval, and Temperature readings are sent to terminal by UART. In DS18x8_demo2 project, PSoC waits for input command from the terminal and returns temperature readings from selected sensor in response.

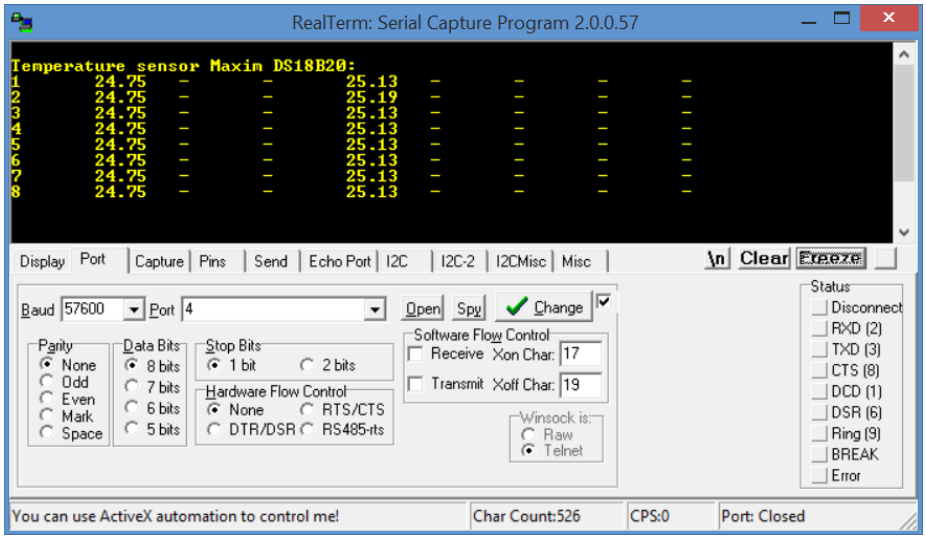


Figure 1. DS18x8_demo project terminal output. Only sensor 1 and 4 are attached.

Resources

Component Technology Mapping Summary tested using PSoC 4200 (Pioneer Board) is provided below.

Resource Type	Used	Free	Max	% Used
Digital clock dividers	1	3	4	25.00%
Pins	3	33	36	8.33%
UDB Macrocells	10	22	32	31.25%
UDB Unique Pterms	13	51	64	20.31%
UDB Total Pterms	14			
UDB Datapath Cells	2	2	4	50.00%
UDB Status Cells	2	2	4	50.00%
Status Registers	1			
StatusI Registers	1			
UDB Control Cells	4	0	4	100.00%
Control Registers	4			
Interrupts	1	31	32	3.13%

Performance

The performance of the component only slightly depends on CPU speed, optimization level, debugger mode, or number of sensors attached because all 1-Wire calls rely on fixed timing delays, specified by the manufacture of the DS18B20 sensor, and all sensors are being sampled simultaneously. The performance of the have been gathered using PSoC4200, CPU speed 24 MHz, with the associated compiler configured in Debug mode with no optimization. These numbers should be treated as approximations and used to determine necessary trade-offs.

The component timing diagram with at least one sensor present is shown on Figure 2. Current implementation of 1-wire calls to the sensor is software-based, which blocks CPU making it unavailable to other tasks during time intervals A-B and C-D per each temperature sampling cycle. Interval A-B duration is 1.79 ms, during this time CPU resets the bus and sends temperature request (actual temperature sampling event is located at point B). Extended interval B-C, during which ADC digitizes temperature, is hardware-based and is non-blocking, and CPU is available to other tasks. Minimal duration of this interval (for 12-bit resolution) is 750 ms, at the end of this interval (point C) interrupt is fired. Interval C-D duration is 2.42 ms, during this time CPU reads temperature data from sensors scratchpad memory.

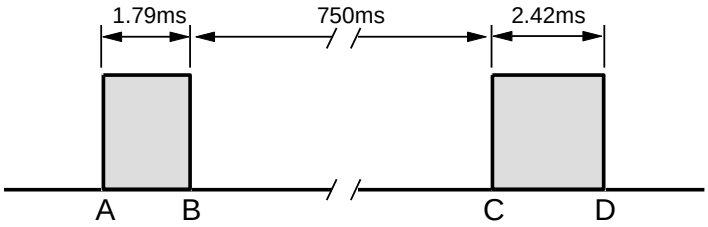


Figure 2. DS18x8 timing diagram with sensors present. A-Send request, B-sample temperature and begin ADC conversion, C- data ready for retrieval (interrupt fired), D- data retrieved.

If component finds no sensors attached, then several time-consuming steps (including ADC conversion) are skipped and result (which is empty bus state) is available sooner.

Figure 3. DS18x8 timing diagram with no sensors present. A-Send request, B-data retrieved = empty bus.

Component Changes

Version	Description of changes	Reason for changes/impact
0.0	Version 0.0 is the first beta release of the DS18x8 component	

References

1. Maxim Integrated**.** DS18B20 Programmable Resolution 1-Wire Digital Thermometer
2. Maxim Integrated. Application Note 162. Interfacing the DS18X20/DS1822 1-Wire® Temperature Sensor in a Microcontroller Environment.
3. Maxim Integrated. Application Note 126. 1-Wire Communication Through Software.
4. Cypress. OneWire Datasheet. One wire V 1.1 (for PSoC1).
5. Cypress AN2163. Interfacing to 1-Wire/Two-Wire Digital Temperature Sensors using PSoC® 1.