Temperature monitor

Table of Contents

[List of Figures i](#_Toc194748675)

[List of Tables i](#_Toc194748676)

[1. Code 2](#_Toc194748677)

## List of Figures

No table of figures entries found.

## List of Tables

**No table of figures entries found.**

To implement this temperature monitor system, I followed the setup guidelines outlined in the FreeRTOS Windows simulator documentation. As recommended by the FreeRTOS wiki, I created a new project based on one of the demos provided in their source code. I selected the Blinky demo, as it includes working examples of key FreeRTOS concepts such as software timers, multiple tasks, queues and critical sections.

# Assumptions

Addresses:

* PCAL6408: 0x20 as specified when ADDR is tied to GND:
  + <https://www.nxp.com/docs/en/data-sheet/PCAL6408A.pdf>
* TMP101: 0x49 as specified when ADD0 is left floating (ie x in the image)
  + <https://www.ti.com/lit/ds/symlink/tmp100.pdf?ts=1743870402830&ref_url=https%253A%252F%252Fwww.google.com%252F>

Buffers:

* Only one task using them at a time as there is only one control register
* In buffer from Data: Bits are moved starting with lowest bit first (bit 0) into the buffer, then are pushed onto the bus starting with the MSB first.
* In buffer from Bus: Bits are received, starting with MSB first into the FIFO buffer, they are then copied into the data register.

Control Register bits:

* Execution bit: **Automatically resets** upon completion
* Data Load bit: **Automatically resets**, once completing a load from the data register into the write FIFO.
* Data size bit: This is **only** the amount to receive for one read, does **not** specify the total number of bits to be read across the entire operation.
  + Data size is also assumed to be specified in **bits,** however cannot exceed over 1 byte in size due to I2C only taking 8 bits at a time.

TMP101

* The temperature is calculated based on TMP101’s 12-bit resolution. Where each bit receives represents 0.0625C.
* The temperature reading frequency ( **every second** ) was also ensured to be larger then the conversion time (320ms) so that each read returns an updated temperature.
* To simulate the output, I have created a function which starts within range, increases in

A white and black rectangular box with black text

AI-generated content may be incorrect.

PCAL6408

* Bits 0 and 5 are seen to be connected to the LED’s
* It is assumed that each led is connected through and NPN transistor meaning
  + When pcal sets a pin to 0 (low), it is connected to GND internally, letting current flow through the LED (ON)
  + When pcal sets a pin to 1 (high), it is an open circuit, indicating that no current is passing through the LED (OFF)

Tasks

* In order of priority

1. A I2C queueing task, which employs a critical section for a tasks attempting to do a read / write operation. This ensures that each FIFO and the Control/Status/data registers will not get mixed up between tasks.
2. A temperature check task. Gets put to sleep for 1s the wakes up to indicate a new write should occur. Greater priority then Status writes as it is the main alert.
3. Status led task. Activates on a 2hz frequency to flash the LED.

Below I have provided an example of output and used portions of the code. I did not add entire code as it includes functions which are only used in the blinky demo.

Hi Lyden and Neerav