Standart Output

This article provides useful information about how the standard output is handled on the SJ2 platform.

`printf`

The standard output is connected to UART0. In a bare metal system without the operating system providing means of outputting data to a console, responsibility lies on the developer to connect `printf()` to your way of outputting data.

In GCC, the function `_write()` is invoked for all data output related to file handles. On the SJ2 platform, the function is implemented to output data to UART0, which is connected to the USB to serial chip that is interfaced to a computer (such as windows, linux) to see the serial console.

`system_calls.c` can be referenced to see the full implementation.

```
int _write(int file_descriptor, const char *ptr, int bytes_to_write) {
    // ...
    if (rtos_is_running && transmit_queue_enabled && !is_standard_error) {
      system_calls__queued_put(ptr, bytes_to_write);
    } else {
      system_calls__polled_put(ptr, bytes_to_write);
    }
    return bytes_to_write;}
```

`fprintf`

When `fprintf(stderr, "...")` is utilized, the `system_calls.c` does not deposit data to an RTOS queue in which case the data would have been sent out "later" depending on the speed of the UART. The `stderr` is the key that differentiate polled vs. queued data output.

When the `stderr` is utilized, this "file handle" triggers the branch statement to output the data using polled

UART driver. This means that the CPU cycles will be compromised, and we will waste cycles waiting for data to be sent, so this should not be used in "production code".

`printf` inside of an ISR

Inside of an interrupt, you never want to "block" using any RTOS API. If we use standard `printf()`, it may try to enqueue the data to be sent out of the UART0 peripheral, and therefore may crash the system when the UART transmission queue becomes full (as it will then try to sleep on the queue to be not full). Because of this, `fprintf(stderr, "...")` may be utilized inside of an ISR as it would not enqueue the data or try to "block" through the RTOS API.

In "production intent" code, there should be no printfs inside of an ISR.

Revision #2
Created 5 years ago by Preet Kang
Updated 5 years ago by Preet Kang