Embedded Systems Project

Lorenzo Pinosa

Mat. 852231 - Computer Science and Engineering

STDOUT redirect to screen

For STM32F429I board

# Specification

This project relies on the STM32F429I board and the Miosix kernel. Its goal is to redirect the standard output to the display. The log of the Miosix boot phase should be printed as well.

Printing to the display is done by using the mxgui library.

# Implementation

## IRQDisplayPrint

The main part of the project is the IRQDisplayPrint class. It inherits from the miosix class Device and implements the methods *IRQwrite(const char \* to\_print)* and *writeBlock(const void \* buffer, size\_t size, off\_t where)* that are called when stdout is flushed.

### Caching strings to print

When those methods are called, the received string is cached in a Queue (input\_queue). This queue will then be processed by another thread, from now on referred to as the print thread. This is to minimize as much as possible the time spent when the stdout is flushed, since this is done on the main thread.

The use of a Queue instead of a Vector is to exploit the blocking nature of the get() method in the printing thread, to avoid dealing with mutex and Interrupts enabling/disabling.

### The print thread

The printing thread is an endless while cycle that waits for new strings to print and processes them.

First of all, it checks if the string is null or empty, and eventually discards it. Then, it uses the *carriage\_return\_fix* method to deal correctly with carriage returns: since every string received is treated already as a new line and carriage returns are received as a stand-alone string, printing them as they are would result in one additional new line every time. The solution is to remove the first carriage return of every sequence of carriage returns received.

Now the strings are processed and stored in a Vector, and then printed.

#### String processing

For each string, *process\_string(string str)* is called. This method measures the string and cuts it into parts, if necessary, that fit into the screen. Then it adds the chunks to a string Vector (*print\_lines*). Thus, print\_lines contains text ready to be printed on the screen without further concerns.

The *process\_string* method starts by gathering information about the font being used and the display width. Then, with two nested cycles, it cuts the string in parts which fit the screen and does that until the string is finished. The outer cycle is designed to automatically skip the internal one if the string is already shorter than the screen width; to improve performance.

#### Screen vertical overflow

After a string has been processed, *check\_array\_overflow()* is called. This method checks if the lines to be printed contained in *print\_lines* are too much with respect to the screen height and, if needed, removes the oldest ones. This will generate a scrolling effect, like a console.

#### Printing on the screen

*internal\_print()* actually prints the text contained in *print\_lines* vector to the screen. Instead of clearing each time the screen and drawing everything again, it prints every line and clears the blank space at the right of the string. This greatly improves the visual appearance of a screen refresh.

## Changes to Board Support Package

To integrate the previously discussed class into Miosix kernel, it was necessary to edit the Board Support Package for the stm32f4discovery board, located in bsp.cpp and bsp.h files in folder miosix\arch\cortexM4\_stm32f4\stm32f429zi\_stm32f4discovery\interfaces-impl.

### IRQset

The main point was to replace the default serial device with an instance of the IRQDisplayPrint class, and save a reference to it.

### Creating the print thread

Then, inside the *bspInit2()* method, it was possible to create the print thread. The thread is initialized with a dummy method that does no more than calling the *printIRQ()* one of the IRQDisplayPrint class, that never returns since it is an endless loop.

# Remarks

To get the modified Miosix kernel compiling it was necessary to modify the Makefile since miosix and mxgui are now referencing each other.