Aula 4 - SPI com IT

Revisão da aula 3

- Finalizada a escrita por blocos (iniciada na aula 2)
- iniciado o gerenciamento do buffer, para modo pooling apenas
- Pendências da aula 3:
    - Possibilitar o usuário configurar os parâmetros iniciais
    - Possibilitar ele escolher o modo de transmissão

Passando a transmissão para modo não bloqueante

- O processador dispara a transmissão
- O chip select deve subir só depois de toda a transmissão ser completa!

Gerenciamento do buffer

Ao implementar a transmissão por SPI_IT, o buffer corre o risco de ser sobreescrito, o que retorna o erro de `void HardFault_Handler(void)`

Isso é evitado da seguinte forma. Quando uma função (como o write ou o write_str) é chamada, ela retorna um status de HAL_OK caso ela tenha sido executada com sucesso. Isso significa que a transmissão dos dados foi disparada. Quando a transmissão acaba, o estado do buffer volta a FREE

No início de cada função há um `if(buf.status==B_BUSY) return HAL_BUSY;` que impede a execução da função enquanto o buffer não estiver livre.

Com isso, a função só é executada caso o buffer esteja disponível!

`HAL_OK` -- Função executada / Transmissão disparada

`B_BUSY` -- Setado no início de cada função, o que impede o uso do buffer pelas demais

`B_FREE` -- Setado no final de todas as transmissões, deixa o buffer disponível

Exemplo testado na main

Neste caso, o código ainda está num modo bloqueante. Porém é possível testar o funcionamento do buffer.

```
    while(LCD5110_set_XY(0, 0)!=HAL_OK){};
    while(LCD5110_write_str("Hello World")!=HAL_OK){};
    while(LCD5110_set_XY(0, 2)!=HAL_OK){};
    while(LCD5110_write_str("Estou Vivo")!=HAL_OK){};
```

A primeira função é executada, e logo ela retorna HAL_OK. A próxima não será executada enquanto o buffer não estiver livre, devido ao if no início da função. Assim, ela só retornará HAL_OK quando o buffer for liberado.