Sistemas Embarcados I - Laboratório 6
Guilherme Goes Zanetti 2019107824 / Luiza Batista Laquini 2019107786
03/02/2022

Objetivos

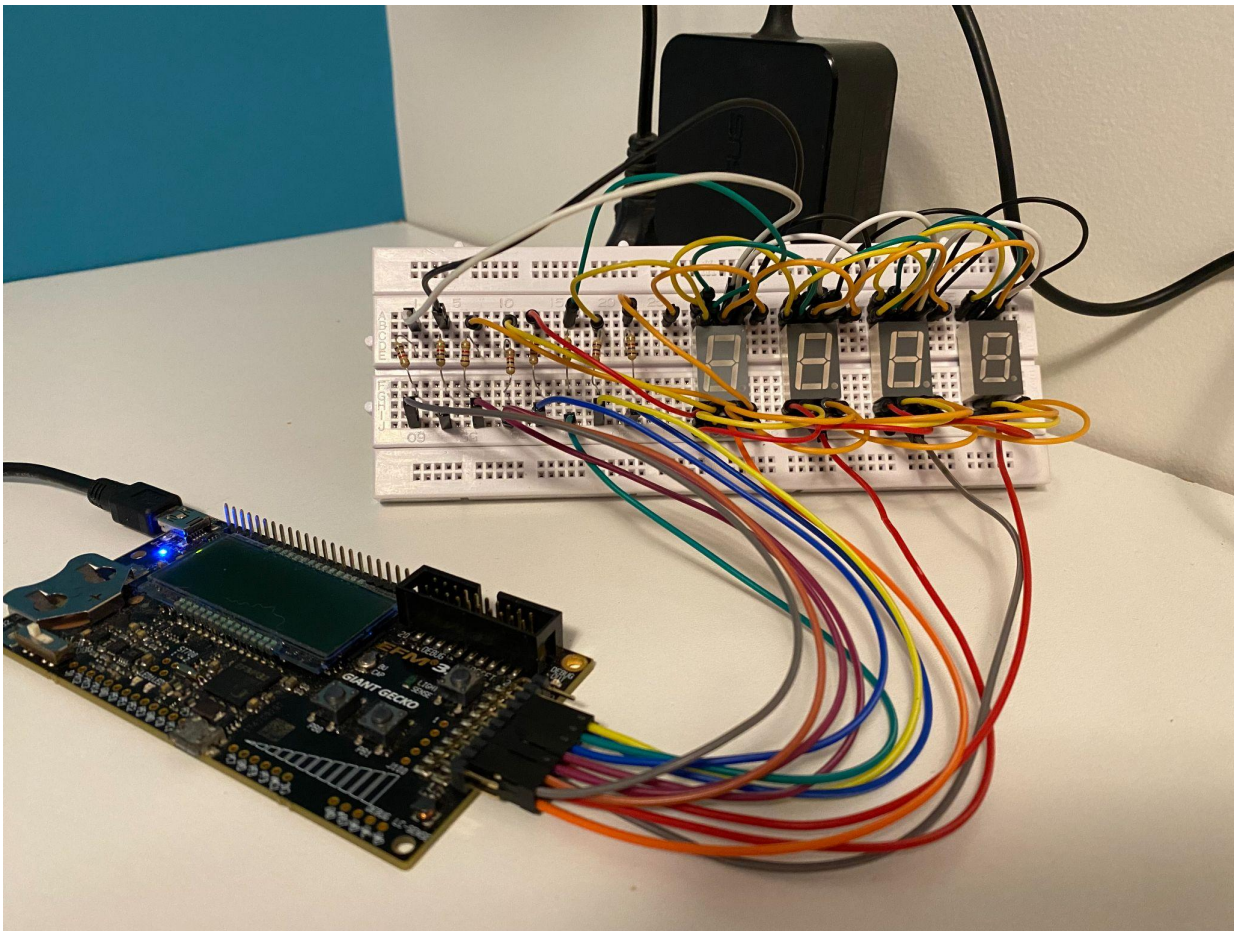
Utilizando a EFM32 Giant Gecko disponibilizada pelo professor:

- Aplicar os conceitos de acionamentos de cargas simples usando linguagem de programação C para microcontroladores (ARM).
- Realização de funções de Delay para visualização dinâmica de displays 7 segmentos.

Resultados

Para realizar a experiência proposta no roteiro 6, utilizamos 4 displays de 7 segmentos cátodo comum, ou seja, conectamos o pino "*COM*" de cada um dos displays com um pino diferente da placa de desenvolvimento EFM32. Para ativar o display em questão, devemos colocar tensão 0 na saída do pino equivalente.

IMAGEM DA MONTAGEM



Ligamos também 7 pinos da EFM32 em sete barramentos que alimentam os segmentos dos displays. Com isso, e uma lógica de ativação sequencial dos displays, podemos criar uma ilusão de estarem todos os displays ativados simultaneamente, mesmo com dígitos diferentes.

Para isso, criamos a função display_4digits() que faz uma sequência de desligar um display, definir os segmentos a serem acesos, religar o display e aguardar um curto período antes de passar para o próximo display, repetindo o processo.

```
void display_4digits(int d1, int d2, int d3, int d4){
    //Display first digit
    GPIOC->DOUT = 0xFF;
    GPIO_WritePins(GPIOD, ~INT_TO_SEG(d1), INT_TO_SEG(d1));
    GPIOC->DOUT = ~(BIT(0));

    Delay(1);

    //Display first digit
    GPIOC->DOUT = 0xFF;
    GPIO_WritePins(GPIOD, ~INT_TO_SEG(d2), INT_TO_SEG(d2));
    GPIOC->DOUT = ~(BIT(3));

    Delay(1);

    //Display first digit
    GPIOC->DOUT = 0xFF;
    GPIO_WritePins(GPIOD, ~INT_TO_SEG(d3), INT_TO_SEG(d3));
    GPIOC->DOUT = ~(BIT(4));

    Delay(1);

    //Display first digit
    GPIOC->DOUT = 0xFF;
    GPIO_WritePins(GPIOD, ~INT_TO_SEG(d4), INT_TO_SEG(d4));
    GPIOC->DOUT = ~(BIT(5));

    Delay(1);
}
```

Para o funcionamento geral do programa, temos um loop aninhado dentro de outro loop, sendo o loop maior a mudança dos dígitos a serem exibidos (passagem de um segundo/minuto para os cronômetros). Já o loop interno vai garantir que o tempo em que cada unidade seja exibida seja constante, ou seja, ele apenas exibe nos displays os dígitos atuais, sendo necessário o loop para manter a ilusão explicada anteriormente.

```c
int vetDigitos[4] = {0, 0, 0, 0};
/* loop */
while (1) {
    //Logic to define the next digits to be shown
    switch (modo){
    case 0: // Apertando somente PB0 (Contador de corrida livre)
        corrida_livre(vetDigitos);
        break;
    case 1: // Apertando somente PB1 (Relógio 24h)
        relogio24h(vetDigitos);
        break;
    case 2: // Apertando PB0 e PB1 juntos (Relógio 12h)
        relogio12h(vetDigitos);
        break;
    default:
        clearArray(vetDigitos);
        break;
    }

    int i=0;
    for (i=0; i<TIME_UNIT; i++){
        //Function that displays the 4 char numbers given
        display_4digits(vetDigitos[0], vetDigitos[1], vetDigitos[2], vetDigitos[3]);
        checkButtons(vetDigitos);
    }
}
```

O que vai ser exibido nos displays depende do modo escolhido pelo usuário através dos botões nativos da placa de desenvolvimento. Quando PB0 é pulsado, a função de contador de corrida livre é ativada (de 0 até 9999). No caso de pulsar o PB1, a função de relógio 24h é ativada (00:00h até 23:59h). Se o PB0 e PB1 são apertados simultaneamente, a função de relógio 12h será ativada (00:00 a.m. até 11:59 a.m. e 00:00 p.m. até 11:59 p.m.).

Nesse último modo, além de exibir o horário do relógio, dois leds da placa podem ser acesos para indicar se estamos durante a manhã (a.m.) ou durante a tarde (p.m.).

Segue vídeo com os resultados alcançados: LAB6_Sistemas_Embarcados_I.mp4

Conclusão

A experiência proposta foi concluída com êxito, já que com os testes que realizamos, foi possível verificar que os diferentes modos estão funcionando corretamente, inclusive nas condições de contorno (24h para o relógio de 24h por exemplo). Foi possível também por meio do experimento ter um maior entendimento sobre a utilização de HALs (Hardware Abstraction Layer) que usamos para inicialização das portas dos GPIO e para a escrita nas portas.