UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA ELÉTRICA E COMPUTAÇÃO

28/04/2016

EA871 - LAB. DE PROGRAMAÇÃO BÁSICA DE SISTEMAS DIGITAIS

EXPERIMENTO 5 – Interface Paralela

Profª Wu Shin-Ting
Aluno: Tiago Eidi Hatta RA: 160388

Proposta do Experimento

Controlar a frequência de piscadas e as cores dos leds do kit FRDM-KL25Z, aplicando conceitos de armazenamento de dados na memória, através da linguagem C, e utilizando os push-buttons da placa auxiliar [4]. Além disso, programar o LCD e o MCU para processar os sinais GPIO, de forma que o visor mostre as informações relacionadas a cor e a frequência selecionada.

Metodologia:

Para a programação do LCD, foram obedecidas as normas determinadas pelo fabricante. Mas conceitos relacionados ao display foram alvos de estudo para resolução do problema, apresentados a seguir:

DDRAM – Display Data RAM: É a memória que grava as os caracteres de 8-bits a serem mostrados no lcd. Para o experimento, o visor tem disponibilidade para 16 caracteres em cada uma das 2 linhas.
CGROM – Character Generator ROM: Memória onde está armazenada os bitmaps de 5x8 pixels de caracteres ASCII já definidos.
CGRAM – Character Generator RAM: Memória disponível para que o programador possa criar novos caracteres. Para isso, é necessário que os dados referentes a cada uma das 5 linhas do bitmap sejam gravados como um número de acordo com cada pixel de cada uma das 8 colunas disponíveis (CGRAM Creating Custom Character [5]).

Além disso, outros conceitos importantes para entender o funcionamento do LCD e posteriormente programá-lo foram:

Pulso: De acordo com o manual do LCD [6], o sinal “Enable” deve ser colocado em alto e, após um breve tempo, retornar ao seu valor anterior para o processamento das instruções. Essa temporização não é igual para todas elas (conforme pode ser visto na seção 9 de [6]).

ASCII: Como a intepretação de dados e instruções de um microcontrolador é apenas a partir do sistema binário (0 e 1), um modo eficiente para que os carateres pudessem ser digitalizados em um lcd, por exemplo, é através da tabela ASCII. Nela, cada caractere está ligado a sua representação em número binário (que pode ser convertida em hexadecimal) [7].

Proposta de Solução

Módulos e Registradores Configurados [1]

Módulo SIM: Especificamente o registrador SCGC5 (System Clock Gating Control Register 5), responsável por habilitar o clock para cada uma das portas relacionadas aos pinos da placa utilizados no experimento.

Módulo PORT: Responsável por dar suporte ao controle das portas da placa principal do kit usado. Registradores:

- PCRn: de acordo com a porta utilizada do pino n, configura a multiplexação para GPIO.
Foram configuradas:
- Porta A, pino 5 e 12: push-buttons da placa auxiliar.
- Porta B, pino 18 e 19: leds vermelho e verde.
- Porta D, pino 1: led azul
- Porta C, pinos 0 a 7: Dados GPIO
Os pinos do LCD são ligados ao da porta C da placa.
- Porta C, pinos 8 a 10: RS, E do LCD e LE do latch, respectivamente.

Módulo GPIO: Comunicação digital do processador do MCU. Registradores:
- PDDR: configura os pinos como saída (leds e lcd) ou entrada (push-buttons) para GPIO.
- PDOR: configura os pinos como saída (leds e lcd) ou entrada (push-buttons) para propósitos gerais.

Abaixo estão os pseudo-códigos das duas partes do experimento, bem como a explicação da solução baseada nos códigos fontes em C.

Pseudo-Código

```
MAIN:
INÍCIO
        CONFIGURE registrador de controle SIM_SCGC5[10] de forma a habilitar clock do módulo GPIO
        CONFIGURE registrador de controle PORTB_PCR18[8:10] de forma que o pino 18 da porta B
        sirva GPIO
        CONFIGURE registrador de controle PORTE_PCR23[8:10] de forma que o pino 23 da porta E
        sirva GPIO

        CONFIGURE registrador de controle GPIOB_PDDR[18, 19] de forma que os pinos 18 e 19 da
        porta B sejam de saída
        CONFIGURE registrador de controle GPIOE_PDDR[23] de forma que o pino 23 da porta E seja
        de saída

        CONFIGURE registrador de controle GPIOC_PDDR[0:10] de forma que os pinos 0 a 10 da porta
        C sejam de saída
        CONFIGURE registrador de controle GPIOE_PDOR[] de forma que os pinos 0 a 10 da porta C
        seja de saída

        INICIA LCD DE ACORDO COM O INDICADO PELO FABRICANTE

        CRIA CARACTERE ê (chama criaCaractere)

        PARA LOOP VERDADEIRO
                SE PUSHBUTTON 12 APERTADO
                        INDICE: TEMPO DE ESPERA
```

```
                ACENDE LED DE ACORDO COM O VALOR DE INDICE

                SE PUSHBUTTON 5 APERTADO
                        NIVEL: TEMPO DE ESPERA

                TEMPO: INVERSO DA FREQUENCIA SELECIONADA POR NIVEL

                DELAY10US passado como parametro TEMPO determinado de delay

                CONVERTE PARA ASCII O VALOR FLOAT DA FREQUENCIA DE PISCADA

                LIMPAR O LCD

                MANDA STRING PARA O VISOR: “Cor: “
                MANDA STRING PARA O VISOR: NOME DA COR SELECIONADA DO LED
                MANDA STRING PARA O VISOR: “Frequ”
                MANDA STRING PARA O VISOR: “ê” (criado anteriomente e armazenado em CGRAM)
                MANDA STRING PARA O VISOR: “ncia: ”
                MANDA STRING PARA O VISOR: FREQUENCIA CONVERTIDA EM ASCII
        FIM PARA
FIM
```

Código Fonte

Enviado como anexo ao relatório pelo Ensino Aberto. Documentado de acordo com o especificado pela Doxygen.

Testes

Para os testes, foram realizadas as seguintes etapas:

1) Teste das cores dos leds, alternando-as em um certo tempo (visível ao olho humano);
2) Mudança das cores do led de acordo com o push-button da placa auxiliar;
3) Mudança da frequência de piscada do led com o push-button da placa auxiliar, em paralelo com a mudança das cores;
4) Teste com o LCD, mostrando alternadamente o nome das cores de acordo com o push-button da placa auxiliar;
5) Teste para mostrar se o caractere “ê” incluido na string “Frequência” foi criado na memória CGRAM com sucesso – Mostrando no visor a string citada;
6) Composição de todos os testes anteriores para concluir a resolução do problema.

Conclusão

A solução apresentada funcionou ao cumprir os objetivos de mostrar as informações de cores e frequência no lcd, de acordo com os tempos determinados pelos push-buttons da placa auxiliar. No entanto, houve um pequeno atraso, tanto na mudança no visor do LCD, quanto nas piscadas do led. Isso se deve ao fato de que o MCU, apesar de conseguir executar instruções com certa rapidez em relação à percepção do olho humano (em micro segundos), ainda resulta em atrasos. Para resolver isso deve-se pensar em uma forma de sincronizar mais de uma entrada para mais de uma saída.

Uma alternativa pode ser a utilização de um tratamento de interrupções, que não foi estudado (na prática) até o momento no curso de EA871.

Referências

[1] FRDM-KL25Z User's Manual.

ftp://ftp.dca.fee.unicamp.br/pub/docs/ea871/ARM/FRDMKL25Z.pdf

[2] KL25 Sub-Family Reference Manual

ftp://ftp.dca.fee.unicamp.br/pub/docs/ea871/ARM/KL25P80M48SF0RM.pdf

[3] ARMv6-M Architecture Reference Manual – ARM Limited.

ftp://ftp.dca.fee.unicamp.br/pub/docs/ea871/ARM/ARMv6-M.pdf

[4] EA871 – Descrição do Hardware da Placa Auxiliar.

ftp://ftp.dca.fee.unicamp.br/pub/docs/ea871/ARM/DescricaoDoHardware.pdf

[5] LCD Tutorial for Interfacing with Microcontrollers. Rickey's World.

http://www.8051projects.net/lcd-interfacing/index.php

[6] Datasheet do display LCD.

ftp://ftp.dca.fee.unicamp.br/pub/docs/ea079/datasheet/AC162A.pdf

[7] TABELA ASCII – UFRJ

http://equipe.nce.ufrj.br/adriano/c/apostila/tabascii.htm