



Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica
e de Computação



EA871 - Laboratório de Programação Básica de Sistemas Digitais

Roteiro 8 - Interface Paralela: LCD

Autores: Fernando Teodoro de Cillo e Rafael Silva Cirino

RA: 197029 223730

Campinas
Maio de 2022

Introdução:

O intuito deste experimento é programar microcontrolador MKL25Z128 para configurar um LCD utilizando sinais GPIO.

Experimento:

1 Os registradores que devem ser configurados para cada módulo são:

- **SIM:** $0x40048038u = (1 < 11)$ // habilitar o bit 11 que é equivalente ao PORTC
- **PORTC_PCR (0 ao 7)** - Dados: $0x4004B000u = (0xFFFFF8FF) - 0x00000100$ // Zera os bits 10, 9 e 8 e depois habilita somente o bit 8
- **PORTC_PCR (8 ao 10)** - RS, E, LE: $0x4004B000u = (0xFFFFF8FF) - 0x00000100$ // Zera os bits 10, 9 e 8 e depois habilita somente o bit 8
- **GPIOC_PDDR:** $0x400FF094u = 0x7FF$ // Ativar bits do 0 ao 10

2

2.a

No LCD foi observado que uma letra tenta sobrescrever a outra e assim não é possível definir o que está escrito, o led D0 fica aceso. Foi possível observar que as letras acabaram se sobrescrevendo no LCD, impossibilitando definir o que foi escrito. O led D_0 fica aceso. Segundo as especificações técnicas para RS, LE e Enable temos 40ns de ciclo, o que foi obtido aproximadamente para os 3 sinais, como podemos ver pela figura 1.

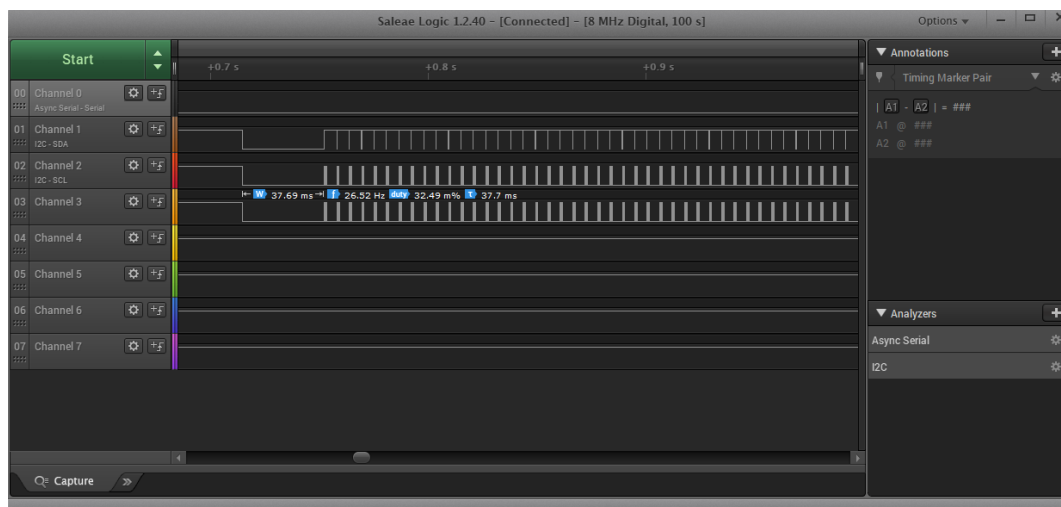


Figura 1: Analisador lógico sem função *delay*

2.b

Adicionando um *delay* foi possível visualizar a alternância entre as letras do alfabeto e os *bitmaps*. O led D_0 fica apagado.

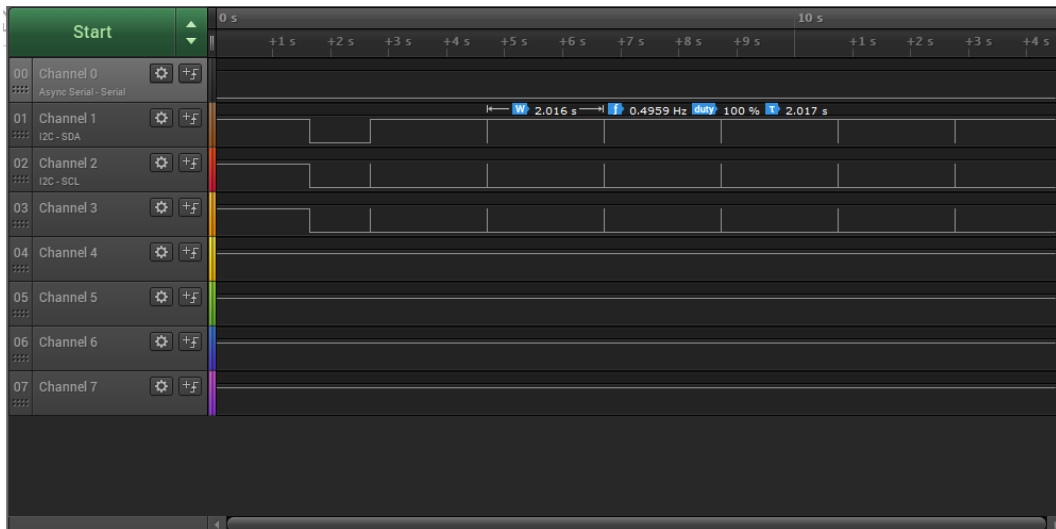


Figura 2: Analisador lógico com função *delay*

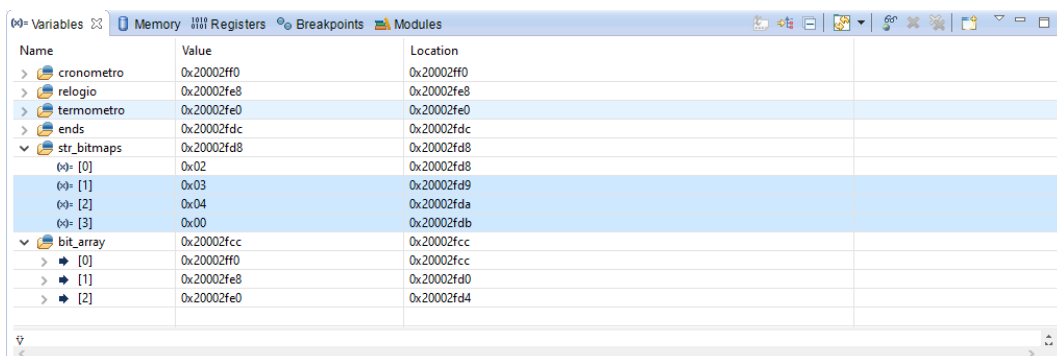
2.c

A segunda linha ficou completamente apagada, pois o `for` que adicionava os *bitmaps* foi comentado.

5

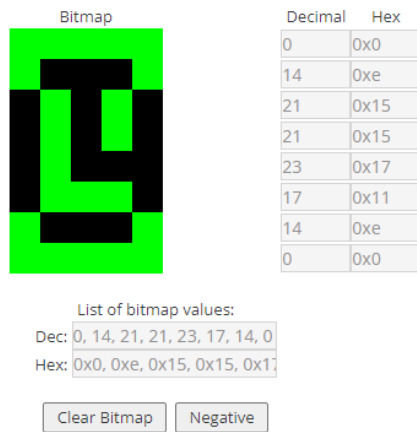
Na figura 3 pode-se observar que `str_bitmap` contém os endereços onde foram salvos os *bitmaps* na CGRAM e em `bit_array` os endereços de cada *array* utilizados para salvar o *bitmap* na posição correta, apontada por `ends` na função `constroiBitmap`.

Já na figura 4 podemos ver os *bitmaps* dos módulos relógio, termômetro e cronômetro (4a, 4b e 4c, respectivamente).

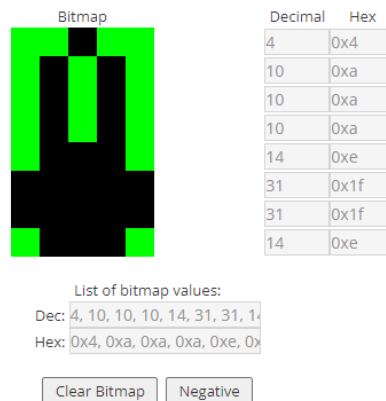


Name	Value	Location
> cronometro	0x20002ff0	0x20002ff0
> relógio	0x20002fe8	0x20002fe8
> termometro	0x20002fe0	0x20002fe0
> ends	0x20002fdc	0x20002fdc
▼ str_bitmaps	0x20002fd8	0x20002fd8
0- [0]	0x02	0x20002fd8
0- [1]	0x03	0x20002fd9
0- [2]	0x04	0x20002fda
0- [3]	0x00	0x20002fdb
▼ bit_array	0x20002fcc	0x20002fcc
> [0]	0x20002ff0	0x20002fcc
> [1]	0x20002fe8	0x20002fd0
> [2]	0x20002fe0	0x20002fd4

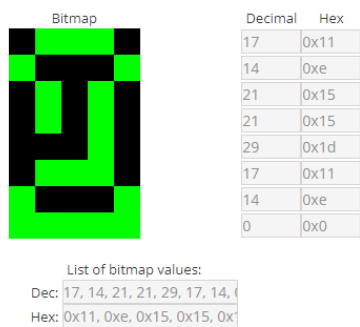
Figura 3: print



(a) Bitmap do relógio



(b) Bitmap do termômetro



(c) Bitmap do cronômetro

Figura 4: Bitmaps

Com uso dos recursos de depuração do IDE é possível enviar os `bitmaps` para o LCD, como mostrado na figura 5.



Figura 5: Visualização dos *bitmaps* no LCD

6 Foram realizados testes na placa em que o nome da cor mostrada no LCD correspondeu ao representado pelo led RGB, da forma que esperávamos, como mostrado na figura 6.

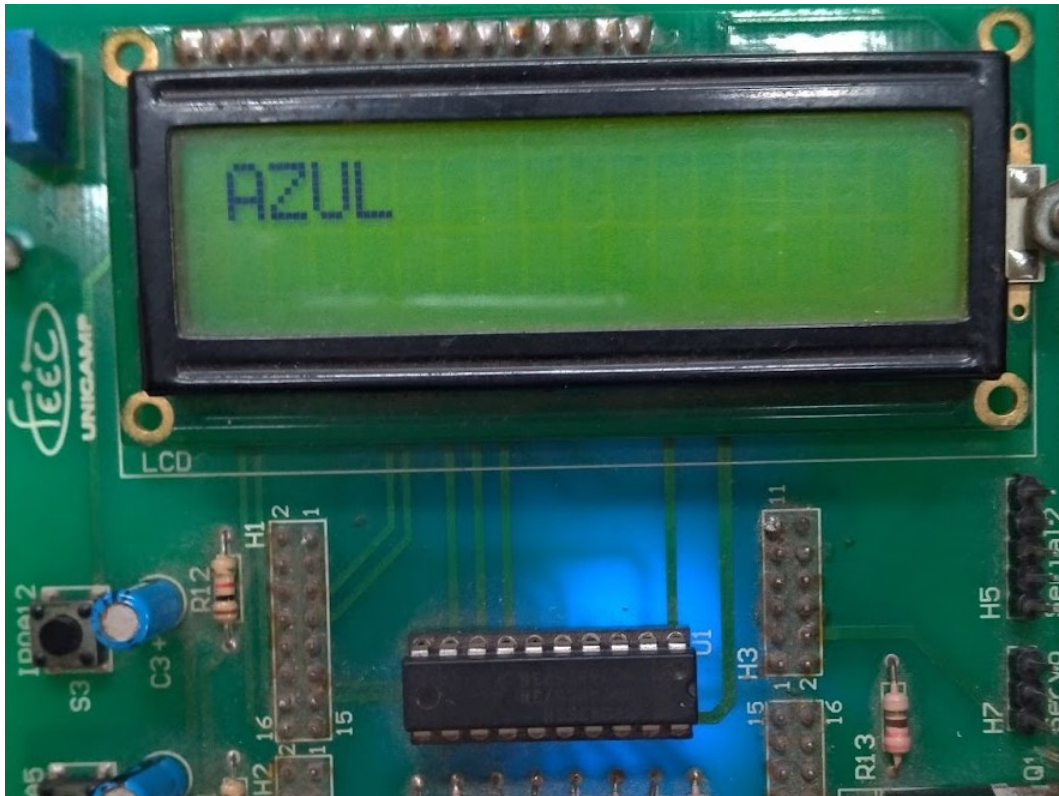


Figura 6: LCD mostrando o nome correspondente à cor do led (azul)

7 O teste foi realizado apertando as botoeiras e parando no *breakpoint*, quando verificou-se o estado atual das variáveis. As figuras 7, 8 e 9 mostra os valores registrados em cada instante.

Name	Value	Location
estado_anterior	0	0x20002fef
estado_corrente	0	0x20002fed
paleta_anterior	0	0x20002fee
paleta_corrente	0	0x20002fec

(a) Estado inicial



(b) LCD mostrando o bitmap do relógio

Figura 7: Estado das variáveis e bitmap do relógio no LCD

Name	Value	Location
(-) estado_anterior	3	0x20002fef
(-) estado_corrente	3	0x20002fed
(-) paleta_anterior	1	0x20002fee
(-) paleta_corrente	1	0x20002fec

(a) Após pressionar IRQA5



(b) LCD mostrando o bitmap do cronômetro

Figura 8: Estado das variáveis e bitmap do cronômetro no LCD

Name	Value	Location
(-) estado_anterior	6	0x20002fef
(-) estado_corrente	6	0x20002fed
(-) paleta_anterior	2	0x20002fee
(-) paleta_corrente	2	0x20002fec

(a) Após pressionar IRQA12



(b) LCD mostrando o bitmap do termômetro

Figura 9: Estado das variáveis e bitmap do termômetro no LCD

8

INÍCIO

definição dos bitmaps

constroiBitmaps (endereços, vetor de bits)

vetor de cores

atribuição do nome de cada cor

loop:

atualiza o estado atual

se estado anterior != estado atual:

estado anterior = estado atual

atualiza a paleta atual

se paleta anterior != paleta atual:

paleta anterior = paleta atual

acende o led correspondente ao estado atual

escreve a cor do led no LCD

mostra o bitmap no LCD

função delay

FIM

Observação: A função delay foi utilizada para que o processador não considere dois acionamentos da botoeira ao apertar apenas uma vez.

9

Podemos ver, na figura 10, o LCD mostrando o *bitmap* no canto superior direito e o nome da cor no canto inferior direito.



Figura 10: Bitmap no canto superior direito

10

A figura 11 mostra o estado das variáveis antes de carregar o programa para a placa e depois de carregá-lo e pressionar a botoeira NMI.

Name	Value	Location
> cronometro	0x20002fe4	0x20002fe4
> relógio	0x20002fdc	0x20002fdc
> termometro	0x20002fd4	0x20002fd4
> ends	0x20002fd0	0x20002fd0
> str_bitmaps	0x20002fcc	0x20002fcc
> bit_array	0x20002fc0	0x20002fc0
> cor	0x20002fb4	0x20002fb4
> cor_nome	0x20002f90	0x20002f90
<- estado_anterior	0	0x20002fef
<- paleta_anterior	0	0x20002fee
<- estado_corrente	0	0x20002fec
<- paleta_corrente	0	0x20002fed
> write_bitmap	0x20002f8c	0x20002f8c

(a) Estado das variáveis antes de carregar o programa para a placa

Name	Value	Location
> cronometro	0x20002fe4	0x20002fe4
> relógio	0x20002fdc	0x20002fdc
> termometro	0x20002fd4	0x20002fd4
> ends	0x20002fd0	0x20002fd0
> str_bitmaps	0x20002fcc	0x20002fcc
> bit_array	0x20002fc0	0x20002fc0
> cor	0x20002fb4	0x20002fb4
> cor_nome	0x20002f90	0x20002f90
<- estado_anterior	1	0x20002fef
<- paleta_anterior	0	0x20002fee
<- estado_corrente	1	0x20002fec
<- paleta_corrente	0	0x20002fed
> write_bitmap	0x20002f8c	0x20002f8c

(b) Depois de pressionar a botoeira NMI

Figura 11: Estado das variáveis

12

As tabelas 1 e 2 mostram testes funcionais de duas sequências de acionamento das botoeiras e as respectivas cores acesas na led RGB.

Tabela 1: Primeira sequência

Estado	[i,j]	Botão pressionado	cor
0	[0,0]	-	azul
1	[0,1]	NMI	ciano
4	[0,2]	IRQA5	ciano
7	[1,0]	IRQA12	magenta
1	[1,1]	NMI	ciano
2	[1,2]	NMI	magenta
5	[2,0]	IRQA5	amarelo
8	[2,1]	IRQA12	amarelo
6	[2,2]	IRQA12	vermelho

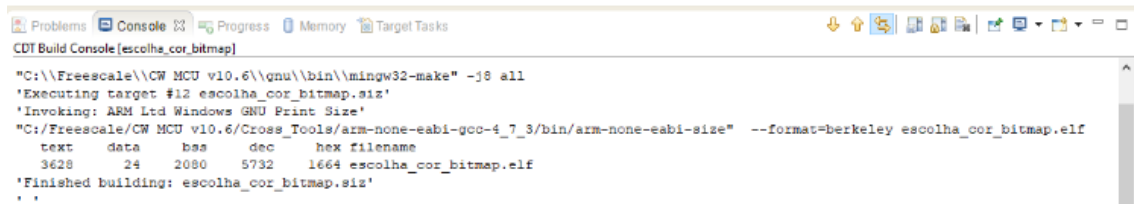
13

Na figura 12 temos o print size obtido no experimento 8, comparando com a figura 13, referente ao experimento 7, podemos perceber que há um aumento somente no arquivo .txt, que corresponde ao tamanho das instruções e dados constantes armazenados na memória flash, o que é um aumento esperado, tendop em vista que foram adicionadas novas funções. Como .data se refere a valores com inicialização armazenada na memória RAM e .bss a variáveis sem

Tabela 2: Segunda sequência

Estado	[i,j]	Botão pressionado	cor
0	[0,0]	-	azul
3	[0,1]	IRQA5	verde
4	[0,2]	IRQA5	ciano
5	[1,0]	IRQA5	amarelo
8	[1,1]	IRQA12	amarelo
2	[1,2]	NMI	magenta
8	[2,0]	IRQA12	amarelo
6	[2,1]	IRQA12	vermelho
0	[2,2]	NMI	azul

inicialização armazenadas na memória RAM, comparando com o último experimento é esperado que não haja aumento, visto que não foram adicionadas variáveis globais ao programa.



```

CDT Build Console [escolha_cor_bitmap]

"C:\Freescall\CW MCU v10.6\gnu\bin\mingw32-make" -j8 all
'Executing target #12 escolha_cor_bitmap.siz'
'Invoking: ARM Ltd Windows GNU Print Size'
"C:/Freescall/CW MCU v10.6/Cross_Tools/arm-none-eabi-gcc-4_7_3/bin/arm-none-eabi-size" --format=berkeley escolha_cor_bitmap.elf
text  data  bss   dec   hex filename
3628   24   2080  5732  1664 escolha_cor_bitmap.elf
'Finished building: escolha_cor_bitmap.siz'

```

Figura 12: *Print size* do projeto do roteiro `escolha_cor_bitmap`



```

CDT Build Console [escolha_cor_interrupt]

'Executing target #11 escolha_cor_interrupt.siz'
'Invoking: ARM Ltd Windows GNU Print Size'
"C:/Freescall/CW MCU v10.6/Cross_Tools/arm-none-eabi-gcc-4_7_3/bin/arm-none-eabi-size" --format=berkeley escolha_cor_interrupt.elf
text  data  bss   dec   hex filename
2136   24   2080  4240  1090 escolha_cor_interrupt.elf
'Finished building: escolha_cor_interrupt.siz'

```

Figura 13: *Print size* do projeto do roteiro `escolha_cor_interrupt`