

# Organização e Arquitetura de Computadores

## Trabalho I: Programação Assembler

Aluno: Gabriel Martins de Miranda

Matricula: 13/0111350

Turma: C

### 1. Objetivos

- Ler nome de uma imagem 512x512 (lena.bmp) como entrada do usuário.
- Mostrar o que foi lido na tela (Bitmap Display).
- Remover componentes R, G ou B da imagem.

### 2. Funções

→ *.data : declaração de variáveis*

- imagem = string que armazena o nome da imagem (entrada do usuário)
- prompt = string de interface com o usuário

→ *main: printa interface com o usuário e espera por entrada.*

*Dependendo da entrada chama os labels correspondentes*

- \$s3 = armazena o endereço inicial do heap
- syscall 4 = mostra 'prompt' na tela
- syscall 5 = armazena entrada do usuário

→ *ler\_nome: cursor esperando entrada do nome da imagem a ser aberta. Digite lena.bmp (sem pressionar enter) para carregar a imagem proposta*

- syscall 8 = ler string de input
- \$s0 = armazena o nome da imagem
- \$s2 = armazena quantidade total de pixels da

imagem(512x512 cada um com 3 bytes)

→ *c\_imagem: abre imagem para leitura, escreve seu conteúdo no endereço do heap e fecha (libera) a imagem carregada*

- syscall 13 = Abre (para leitura) o arquivo representado pela variável imagem

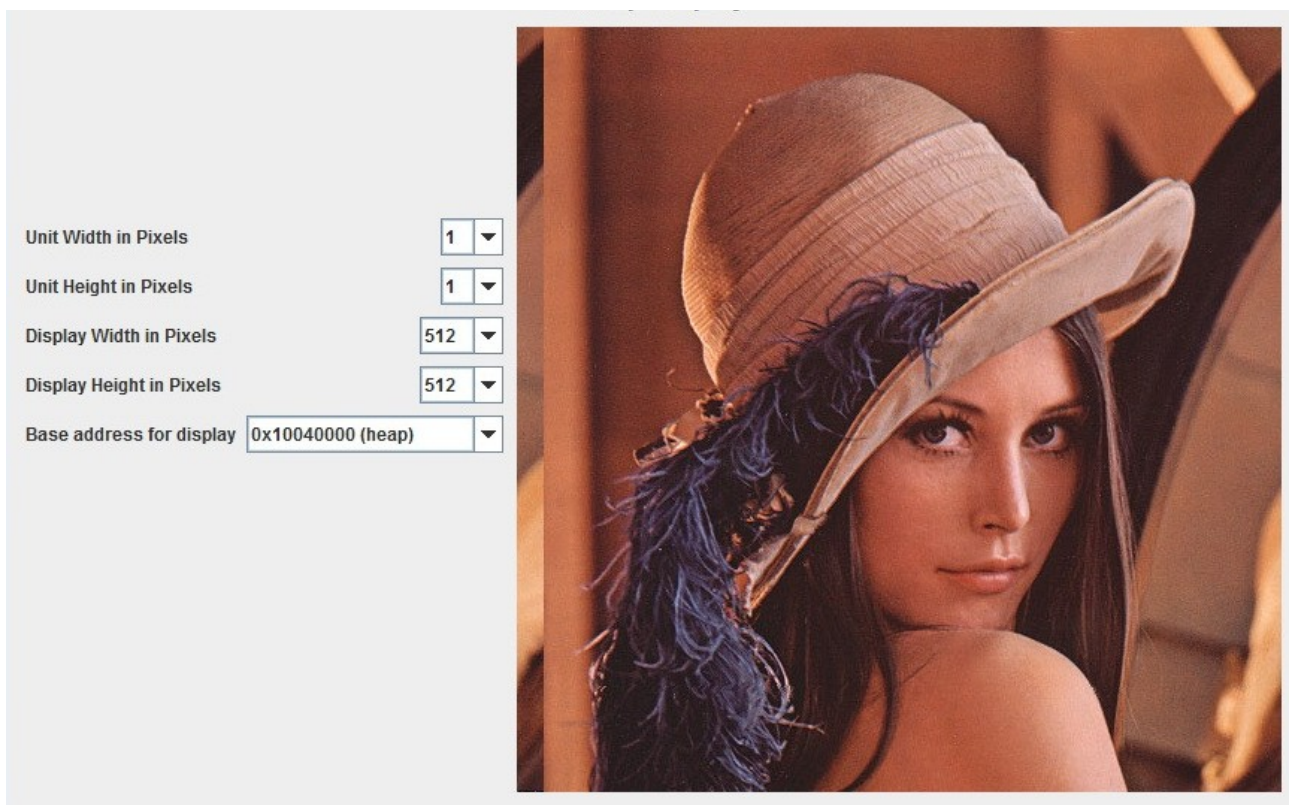
- \$s1 = armazena ponteiro do arquivo aberto
- syscall 14 = Lê do arquivo recém-aberto
- syscall 16 = Fecha arquivo

→ *tira\_r*: percorre o espaço de memória do heap destinada à imagem e zera todas as componentes de pixels que representam a cor vermelha. Utilizou-se uma *andi* com argumento `0x0000FFFF` em cada um deles.

→ *tira\_g*: percorre o espaço de memória do heap destinada à imagem e zera todas as componentes de pixels que representam a cor verde. Utilizou-se uma *andi* com argumento `0x00FF00FF` em cada um deles.

→ *tira\_b*: percorre o espaço de memória do heap destinada à imagem e zera todas as componentes de pixels que representam a cor azul. Utilizou uma *andi* com argumento `0x00FFFF00` em cada um deles.

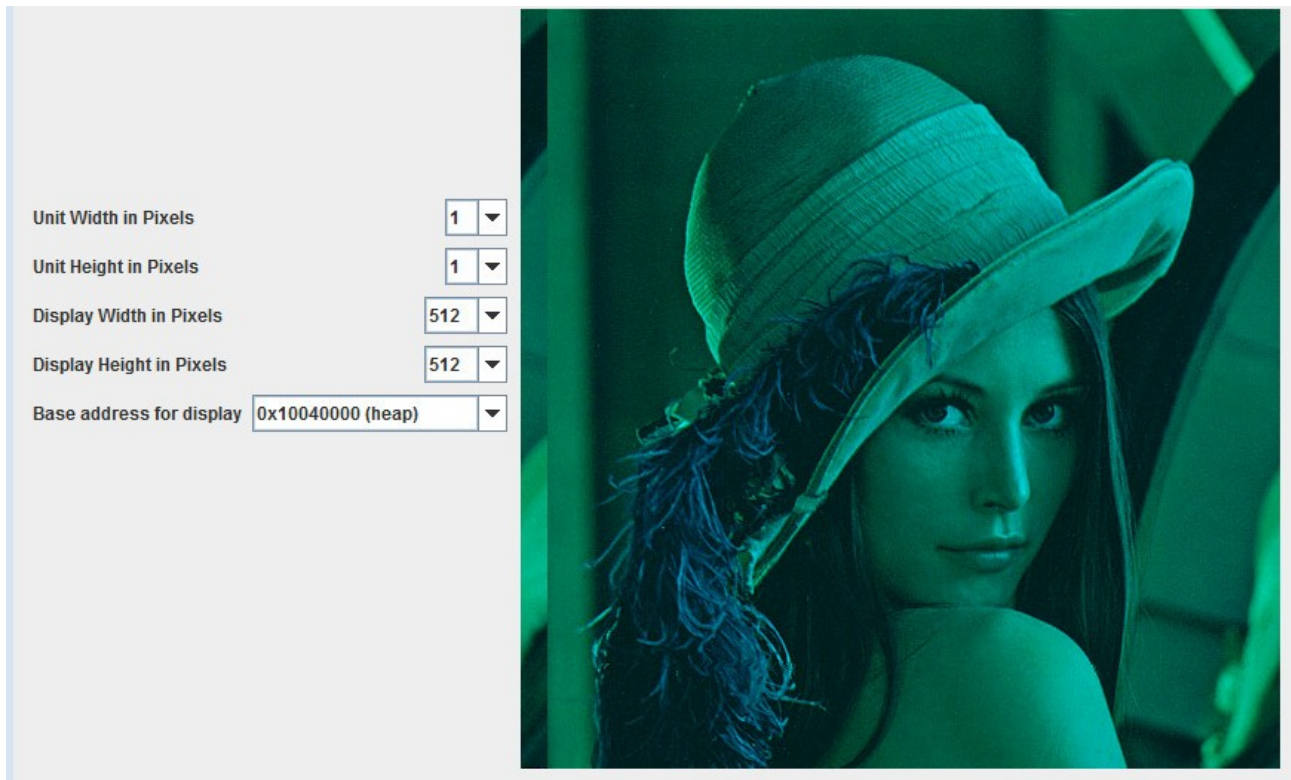
- *encerra*: opção que sai do programa
- syscall 10 = sai do programa



Im1: Imagem lena.bmp carregada .

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10040000	0x00894c37	0x00965843	0x008a513d	0x00894f3b	0x008b513d	0x00874d39	0x00884b38	0x008c4f3c
0x10040020	0x00925542	0x008f503e	0x00945039	0x0095513a	0x0095513a	0x0095513a	0x00945039	0x00945039
0x10040040	0x00945039	0x00945039	0x00c16f4a	0x00c6744f	0x00c3714c	0x00c36f4b	0x00c87450	0x00c97350
0x10040060	0x00c7714e	0x00ca7451	0x00c5704b	0x00c6714c	0x00c46f4a	0x00c06a47	0x00c06a47	0x00c16b48
0x10040080	0x00be6847	0x00ba6443	0x00ba6540	0x00b8633e	0x00b6603d	0x00b35f3d	0x00b05b3c	0x00aa5739
0x100400a0	0x00a55135	0x00a14d33	0x008b402b	0x008b422f	0x007f3a2a	0x0078392a	0x0071372c	0x0069342c
0x100400c0	0x00673631	0x005e2e2a	0x005d2d29	0x005b2c26	0x00562721	0x005a2b25	0x0066352e	0x0066352e
0x100400e0	0x00613029	0x0064342a	0x0070352f	0x00733830	0x006d322a	0x006a2f27	0x0070362b	0x0071372c

Im2: Memória heap que representa a imagem acima. Note que o formato dos pixels é 0x00?????.

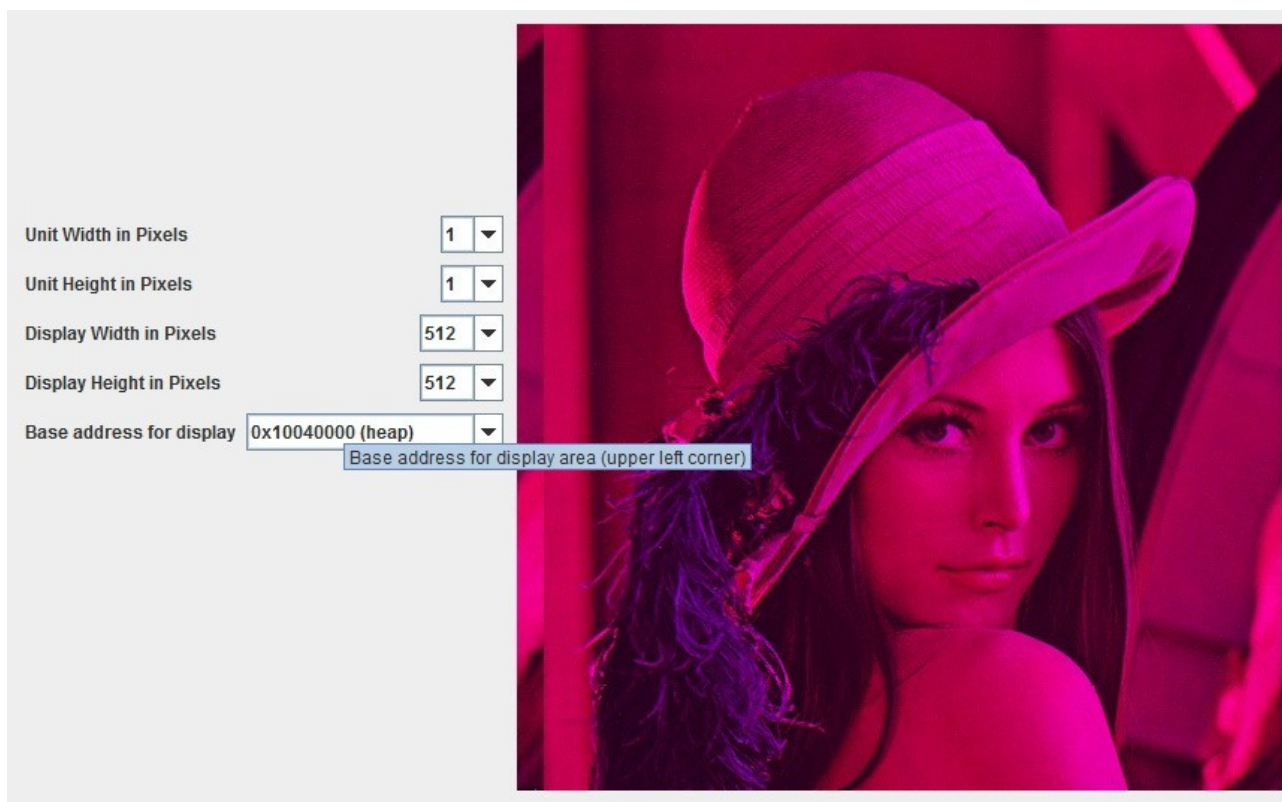


Im3: Imagem original sem componente R (vermelha).

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10040000	0x00004c37	0x00005843	0x0000513d	0x00004f3b	0x0000513d	0x00004d39	0x00004b38	0x00004f3c
0x10040020	0x00005542	0x0000503e	0x00005039	0x0000513a	0x0000513a	0x0000513a	0x00005039	0x00005039
0x10040040	0x00005039	0x00005039	0x00006f4a	0x0000744f	0x0000714c	0x00006f4b	0x00007450	0x00007350
0x10040060	0x0000714e	0x00007451	0x0000704b	0x0000714c	0x00006f4a	0x00006a47	0x00006a47	0x00006b48
0x10040080	0x00006847	0x00006443	0x00006540	0x0000633e	0x0000603d	0x00005f3d	0x00005b3c	0x00005739
0x100400a0	0x00005135	0x00004d33	0x0000402b	0x0000422f	0x00003a2a	0x0000392a	0x0000372c	0x0000342c
0x100400c0	0x00003631	0x00002e2a	0x00002d29	0x00002c26	0x00002721	0x00002b25	0x0000352e	0x0000352e
0x100400e0	0x00003029	0x0000342a	0x0000352f	0x00003830	0x0000322a	0x00002f27	0x0000362b	0x0000372c

Im4: Memória heap que representa a imagem acima. Observe que o formato dos pixels é 0x0000?????. A componente R (vermelha) foi zerada.

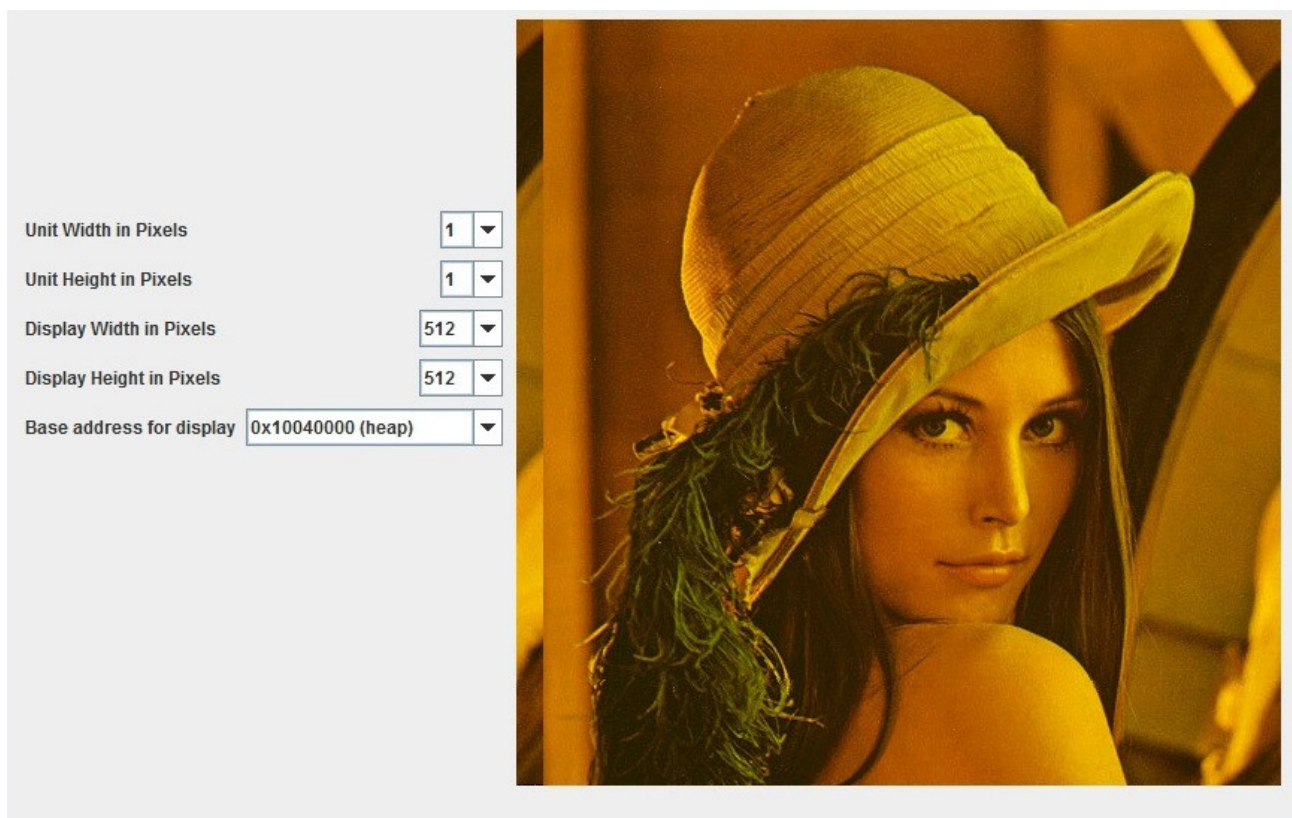




Im5: Imagem original sem a componente G (verde).

Data Segment								
Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10040000	0x00890037	0x00960043	0x008a003d	0x0089003b	0x008b003d	0x00870039	0x00880038	0x008c003c
0x10040020	0x00920042	0x008f003e	0x00940039	0x0095003a	0x0095003a	0x0095003a	0x00940039	0x00940039
0x10040040	0x00940039	0x00940039	0x00c1004a	0x00c6004f	0x00c3004c	0x00c3004b	0x00c80050	0x00c90050
0x10040060	0x00c7004e	0x00ca0051	0x00c5004b	0x00c6004c	0x00c4004a	0x00c00047	0x00c00047	0x00c10048
0x10040080	0x00be0047	0x00ba0043	0x00ba0040	0x00b8003e	0x00b6003d	0x00b3003d	0x00b0003c	0x00aa0039
0x100400a0	0x00a50035	0x00a10033	0x008b002b	0x008b002f	0x007f002a	0x0078002a	0x0071002c	0x0069002c
0x100400c0	0x00670031	0x005e002a	0x005d0029	0x005b0026	0x00560021	0x005a0025	0x0066002e	0x0066002e
0x100400e0	0x00610029	0x0064002a	0x0070002f	0x00730030	0x006d002a	0x006a0027	0x0070002b	0x0071002c

Im6: Memória heap da imagem acima. Aqui pode-se observar que os pixels estão no formato 0x00??00??. A componente G (verde) foi zerada.



Im7: Imagem original sem a componente B (azul).

Data Segment									
Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)	
0x10040000	0x00894c00	0x00965800	0x008a5100	0x00894f00	0x008b5100	0x00874d00	0x00884b00	0x008c4f00	▲
0x10040020	0x00925500	0x008f5000	0x00945000	0x00955100	0x00955100	0x00955100	0x00945000	0x00945000	
0x10040040	0x00945000	0x00945000	0x00c16f00	0x00c67400	0x00c37100	0x00c36f00	0x00c87400	0x00c97300	=
0x10040060	0x00c77100	0x00ca7400	0x00c57000	0x00c67100	0x00c46f00	0x00c06a00	0x00c06a00	0x00c16b00	
0x10040080	0x00be6800	0x00ba6400	0x00ba6500	0x00b86300	0x00b66000	0x00b35f00	0x00b05b00	0x00aa5700	
0x100400a0	0x00a55100	0x00a14d00	0x008b4000	0x008b4200	0x007f3a00	0x00783900	0x00713700	0x00693400	
0x100400c0	0x00673600	0x005e2e00	0x005d2d00	0x005b2c00	0x00562700	0x005a2b00	0x00663500	0x00663500	
0x100400e0	0x00613000	0x00643400	0x00703500	0x00733800	0x006d3200	0x006a2f00	0x00703600	0x00713700	▼

Im8. Memória que representa a imagem acima. Note que os pixels possuem o formato 0x00????00. Aqui a componente B (azul) de cada pixel foi zerada.