Projeto de Bloco: Arquitetura de Computadores, Sistemas Operacionais e Redes
Teste de Performance 3

Principais diferenças entre computadores de 32 bits e 64 bits

Sistema Operacional

Nessa frente, uma característica importante é o fato de as versões de 64 bits serem capazes de reconhecer uma quantidade maior de memória RAM do que as versões de 32 bits.

Portanto, para que o sistema operacional possa apresentar um melhor desempenho de processamento, não basta apenas que seu processador seja compatível com uma arquitetura superior, mas também que opere na versão de 64 bits.

Processador

Em suma, as terminologias "computador de 32 bits" ou "computador de 64 bits" correspondem à arquitetura do processador e do sistema operacional de uma determinada máquina. A maior parte dos processadores atuais são capazes de processar dados tanto de 64 ou 32 bits. Portanto, é comum que os programas desenvolvidos contem com versões compatíveis com ambas arquiteturas.

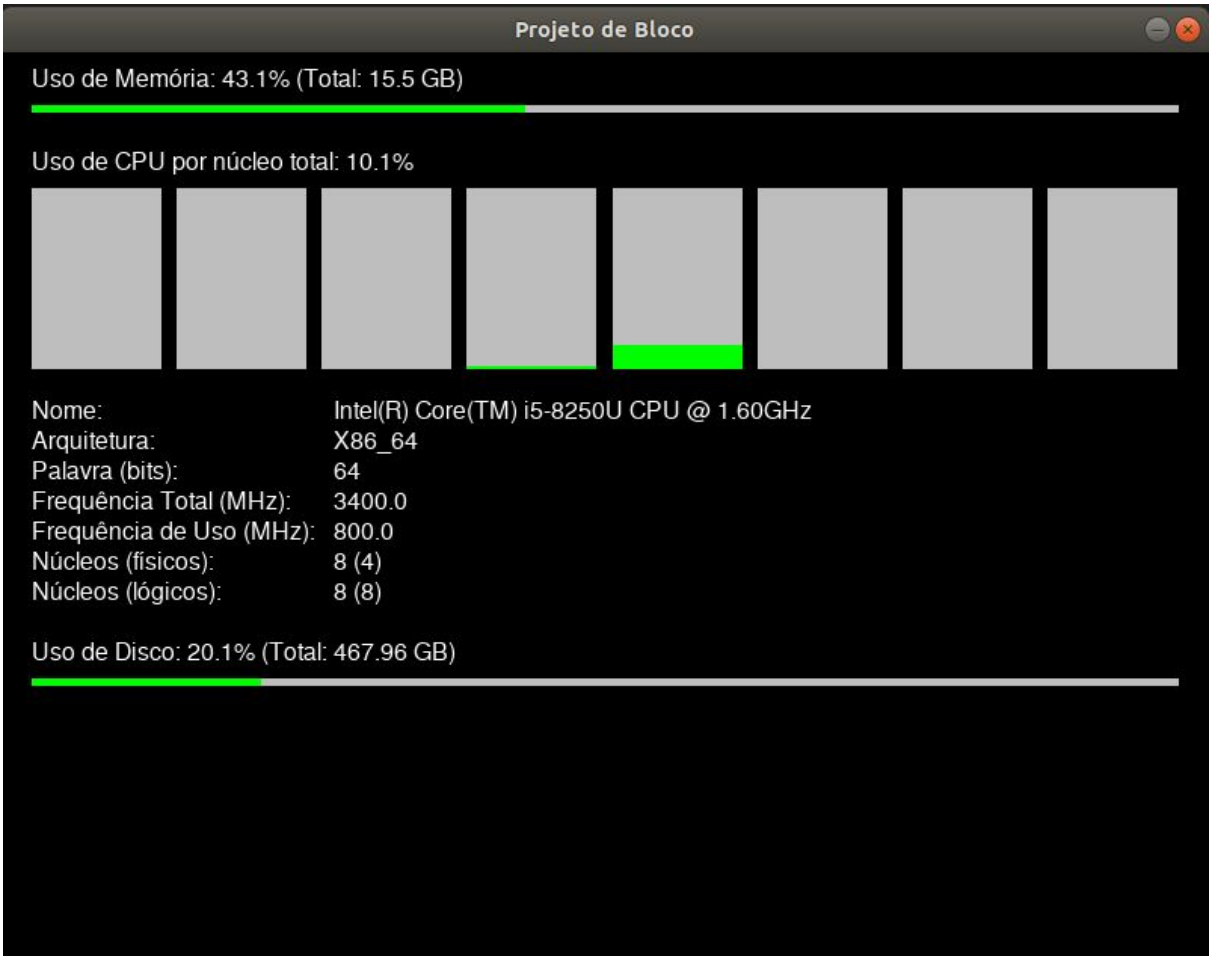
Parte Técnica

Simplificando, processadores de 32 bits são capazes de processar sequências de até 32 bits. Em contrapartida, processadores de 64 bits podem trabalhar com sequências de até 64 bits. Ou seja, processadores de 64 bits têm a capacidade de trabalhar com o dobro de informações.

O que é a palavra do processador:
R: Em ciência da computação, **palavra** é a unidade natural de informação usada por cada tipo de computador em particular. É uma sequência de bits de tamanho fixo que é processado em conjunto numa máquina. A quantidade de dados transferidos entre os **processadores** e a memória é também geralmente uma **palavra**.
Diferença entre núcleos físicos e lógicos:
R: O **Núcleo Físico** é o **núcleo** real **do** processador. Ex: Um dual core = 2 **núcleos físicos**. O **Núcleo Lógico**: É um **núcleo** criado pelo próprio processador, um **núcleo** virtual criado para ajudar os **físicos**, se eles estiverem **em** um alto nível **de** uso.

O entregável possui 5 telas, a primeira tela mostra informações sobre a memória, a segunda tela sobre o uso da CPU por núcleo, a terceira exibe informações sobre a arquitetura do sistema, a quarta sobre o uso de disco e a quinta exibe informações sobre todas as telas.

Exibição de uma das telas com informações do sistema:



Código do sistema:

```
import pygame
```

```
import psutil
import sys
import cpuinfo

################################################################################
################################################
# Instruções de uso do programa:
# Função para chamar as funções e aplicar evento das setas do teclado

# Foram usadas 5 telas:

# Tela 1: Uso de Memória
# Tela 2: Uso de CPU
# Tela 3: Informações da CPU
# Tela 4: Uso do Disco
# Tela 5: Chama todas as telas anteriores

# Para trocar de tela, da Tela 1 até a Tela 5, basta usar a seta da
direita
# Para acessar a última tela, Tela 5, e sair dela de volta para a Tela
1, use barra de espaço.
# A seta da esquerda não foi desenvolvida.

################################################################################
################################################
# Função para capturar o uso de memória


def mostra_uso_memoria(x, y):
    memoria = psutil.virtual_memory()
    largura = largura_tela - 2 * 20
    pygame.draw.rect(sur_memoria, cinza, (20, 5, largura, 5))
    tela.blit(sur_memoria, (0, x))
    largura = largura * memoria.percent / 100
    pygame.draw.rect(sur_memoria, verde, (20, 5, largura, 5))
    tela.blit(sur_memoria, (0, x))
    total = round(memoria.total / (1024 * 1024 * 1024), 2)
    percentagem = memoria.percent
    texto_da_barra = (
        'Uso de Memória: {}% (Total: {} GB)'.format(percentagem, total))
    text = font.render(texto_da_barra, 1, branco)
    tela.blit(text, (20, y))
```

```
# Função para capturar o uso de CPU


def mostra_uso_cpu(s, l_cpu_percent, A, B, C, D, E):
    s.fill(preto)
    capacidade = psutil.cpu_percent(interval=1)
    num_cpu = len(l_cpu_percent)
    x = y = 10
    desl = 10
    alt = s.get_height() - 2 * y
    larg = (s.get_width() - 2 * y - (num_cpu + 1) * desl) / num_cpu
    d = x + desl
    for i in l_cpu_percent:
        pygame.draw.rect(s, C, (d, y + 20, larg, alt))
        pygame.draw.rect(s, D, (d, y + 20, larg, (1 - i / 100) * alt))
        d = d + larg + desl
    text = font.render("Uso de CPU por núcleo total: " +
                       str(capacidade) + "%", 1, E)
    s.blit(text, (20, A))
    tela.blit(s, (0, B))

# Função para mostrar os detalhes da cpu


def mostra_info_cpu(s, X):
    mostra_texto(s, "Nome:", "brand_raw", 10)
    mostra_texto(s, "Arquitetura:", "arch", 30)
    mostra_texto(s, "Palavra (bits):", "bits", 50)
    mostra_texto(s, "Frequência Total (MHz):", "freq_total", 70)
    mostra_texto(s, "Frequência de Uso (MHz):", "freq", 90)
    mostra_texto(s, "Núcleos (físicos):", "nucleos", 110)
    mostra_texto(s, "Núcleos (lógicos):", "nucleos_logicos", 130)
    tela.blit(s, (10, X))
    s.fill(preto)

# Função para mostrar as infos da cpu


def mostra_texto(sur, nome, chave, pos_y):
    text = font.render(nome, True, branco)
    sur.blit(text, (10, pos_y))
    if chave == "freq":
        s = str(round(psutil.cpu_freq().current, 1))
```

```
    elif chave == "freq_total":
        s = str(round(psutil.cpu_freq().max, 2))
    elif chave == "nucleos":
        s = str(psutil.cpu_count())
        s = s + " (" + str(psutil.cpu_count(logical=False)) + ")"
    elif chave == "nucleos_logicos":
        s = str(psutil.cpu_count())
        s = s + " (" + str(psutil.cpu_count(logical=True)) + ")"
    else:
        s = str(info_cpu[chave])
    text = font.render(s, True, branco)
    sur.blit(text, (210, pos_y))

# Função para capturar o uso do disco


def mostra_uso_disco(x, y):
    disco = psutil.disk_usage('.')
    largura = largura_tela - 2 * 20
    pygame.draw.rect(sur_disco, cinza, (20, 5, largura, 5))
    tela.blit(sur_disco, (0, x))
    largura = largura * disco.percent / 100
    pygame.draw.rect(sur_disco, verde, (20, 5, largura, 5))
    tela.blit(sur_disco, (0, x))
    total = round(disco.total / (1024 * 1024 * 1024), 2)
    percentagem = disco.percent
    texto_da_barra = (
        'Uso de Disco: {}% (Total: {} GB)'.format(percentagem, total))
    text = font.render(texto_da_barra, 1, branco)
    tela.blit(text, (20, y))


def slide():
    tela_1 = tela_2 = tela_3 = tela_4 = tela_5 = True

    while tela_1:
        tela.fill(preto)
        mostra_uso_memoria(270, 240)
        pygame.display.flip()
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
```

```
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_RIGHT:
                    tela_1 = False
                    tela_2 = True
                elif event.key == pygame.K_LEFT:
                    tela_1 = False
                    tela_4 = True
                elif event.key == pygame.K_SPACE:
                    tela_1 = False
                    tela_5 = True

    while tela_2:
        tela.fill(preto)
        mostra_uso_cpu(sur_cpu, psutil.cpu_percent(
            percpu=True), 5, 200, verde, cinza, branco)
        pygame.display.flip()
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_RIGHT:
                    tela_2 = False
                    tela_3 = True
                elif event.key == pygame.K_LEFT:
                    tela_2 = False
                    tela_1 = True
                elif event.key == pygame.K_SPACE:
                    tela_2 = False
                    tela_5 = True

    while tela_3:
        tela.fill(preto)
        mostra_uso_cpu(sur_cpu, psutil.cpu_percent(
            percpu=True), 5, 40, preto, preto, preto)
        mostra_info_cpu(sur_infocpu, 200)
        pygame.display.flip()
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_RIGHT:
```

```
                tela_3 = False
                tela_4 = True
            elif event.key == pygame.K_LEFT:
                tela_3 = False
                tela_2 = True
            elif event.key == pygame.K_SPACE:
                tela_3 = False
                tela_5 = True

while tela_4:
    tela.fill(preto)
    mostra_uso_disco(270, 240)
    pygame.display.flip()

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            sys.exit()
        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_RIGHT:
                tela_4 = False
                tela_5 = True
            elif event.key == pygame.K_LEFT:
                tela_4 = False
                tela_3 = True
            elif event.key == pygame.K_SPACE:
                tela_4 = False
                tela_5 = True

while tela_5:
    tela.fill(preto)
    mostra_uso_memoria(30, 10)
    mostra_uso_cpu(sur_cpu, psutil.cpu_percent(
        percpu=True), 5, 60, verde, cinza, branco)
    mostra_info_cpu(sur_infocpu, 220)
    mostra_uso_disco(410, 390)
    pygame.display.flip()

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            sys.exit()
        if event.type == pygame.KEYDOWN:
```

```
                if event.key == pygame.K_RIGHT:
                    tela_5 = False
                    tela_1 = True
                if event.key == pygame.K_SPACE:
                    tela_5 = False
                    tela_1 = True


# Criando a tela do sistema
largura_tela = 800
altura_tela = 600
info_cpu = cpuinfo.get_cpu_info()
tela = pygame.display.set_mode((largura_tela, altura_tela))
pygame.display.set_caption("Projeto de Bloco")
pygame.display.init()
pygame.font.init()
font = pygame.font.SysFont("arial", 16)

# Cores
branco = (255, 255, 255)
preto = (0, 0, 0)
cinza = (190, 190, 190)
verde = (0, 255, 0)
vermelho = (255, 0, 0)

# Surfaces
sur_infocpu = pygame.surface.Surface((largura_tela, altura_tela/4))
sur_memoria = pygame.surface.Surface((largura_tela, altura_tela/4))
sur_cpu = pygame.surface.Surface((largura_tela, altura_tela/4))
sur_disco = pygame.surface.Surface((largura_tela, altura_tela/4))

while True:
   for event in pygame.event.get():
       if event.type == pygame.QUIT:
           pygame.quit()
           sys.exit()
       slide()

   pygame.display.update()
```