1 TRABALHO:

1.1 Introdução

O objetivo desta tarefa é fazer com que o aluno crie um simples sistema de arquivos que deve utilizar uma memória i2c EEPROM. O sistema de arquivos tem um layout apresentado em aula (Fig.1) e cada entrada de arquivo é representada como um inodo (Fig.2).

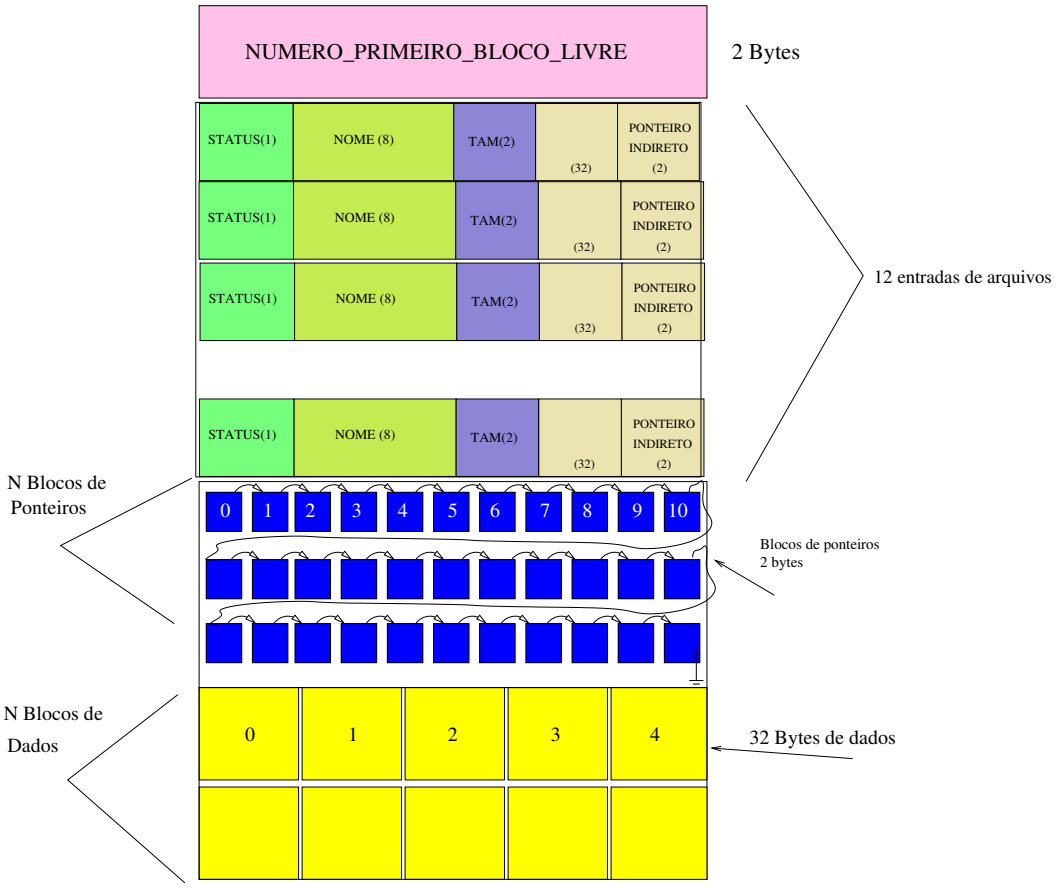

Figure 1: Layout

1.1.1 Como funciona:

Existe um cabeçalho de 2 Bytes que indica qual é o número do primeiro bloco de dados livre. Após, existem 12 entradas de arquivos (inodos). Cada entrada de arquivo é composta por campos (Status - Indica se a entrada está livre ou ocupado, Nome - 8 letras para o nome do arquivo, tamanho - quantos bytes o arquivo ocupa, bloco direto (32 bytes de dados), bloco indireto (é o endereco de um bloco cujo conteúdo são enderecos para blocos de dados) . Em azul temos os blocos de ponteiros e abaixo destes os blocos de dados. Os blocos de ponteiros possuem 2 Bytes cada e iniciamente foram organizados para criar uma lista encadeada. Um bloco armazena nos seus 2 Bytes o endereço o próximo bloco livre. Existe uma relação entre cada bloco de ponteiro e o bloco de dados. O bloco de ponteiro 0 está relacionado ao bloco de dados 0.

1.1.2 Criação de uma entrada de arquivo:

Deve-se procurar por uma entrada livre (entrada com o campo status valendo 0) Quando for encontrada, marca-se a entrada como ocupada (status valendo 1).

1.1.3 Alocação:

Toda a vez que um bloco precisa ser alocado, verifica-se o cabeçalho e descobre-se qual é o primeiro bloco livre que seja utilizado. Depois, acessando o número desse bloco na lista encadeada, descobre-se qual o número do próximo bloco e esse número será o novo primeiro bloco livre no cabeçalho.

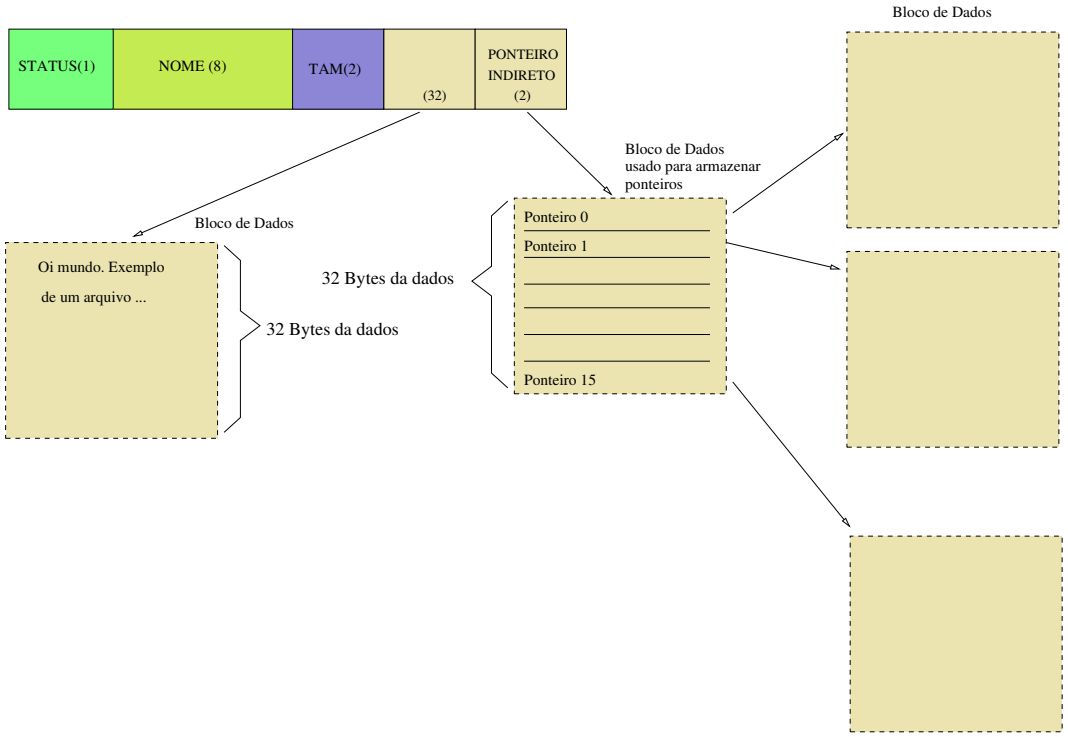

Figure 2: inodo

1.1.4 Remoção de um arquivo:

Para remover um arquivo é preciso desalocar todos os blocos de dados pertencentes a um arquivo e depois disso, tornar a entrada de arquivo disponível. Os blocos desalocados devem ser inseridos na lista encadeada.

O sistema de arquivos pode ser imaginado como um sistema em camadas, tal como mostrado na figura 3.

1.2 O que é solicitado ?

Implemente o código do sistema de arquivos para torná-lo funcional e mais especificamente as funções mostradas na tabela abaixo. Teste usando o exemplo em C mostrado:

PS: Você pode implementar melhorias no sistema de arquivo, como por exemplo um mecanismo de cache para evitar perda de tempo acessando posições de memória que já se encontram num buffer.

Função	comentários
fopen	implemente apenas os modos de abertura r+ e w
fread / fwrite	implemente essas funções para ler e escrever vários bytes de uma só vez. Note que você não pode fazer chamadas sucessivas a fgetc/fputc para implementar estas funções em virtude do overhead envolvido
feof	implemente esta função
ftell	implemente esta função
fclose	implemente esta função

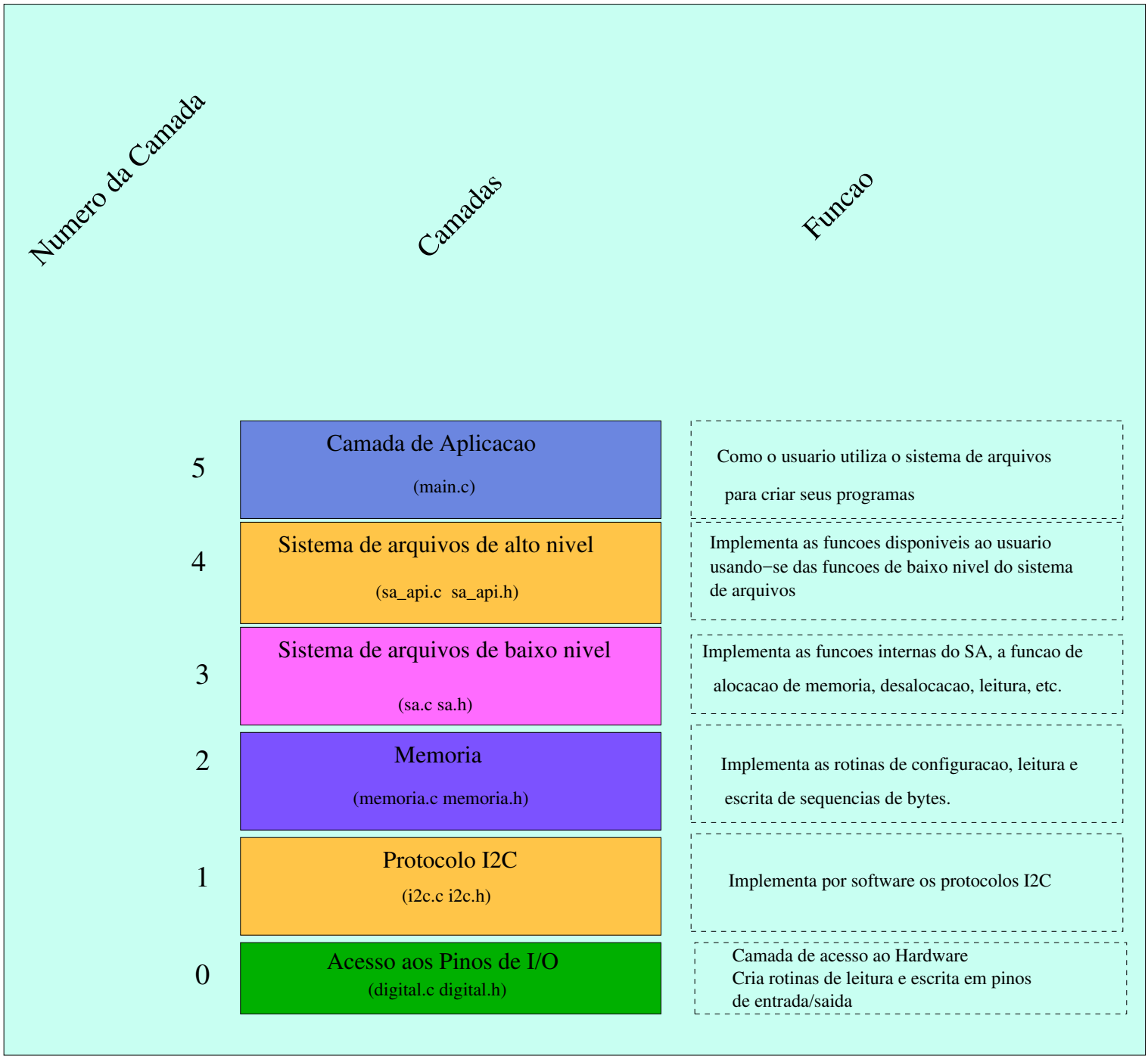

Figure 3: Sistema visto em camadas

```
1
2  #include <stdio.h>
3  #include <string.h>
4  #include "serial.h"
5  #include "digital.h"
6  #include "sa_api.h"
7
8
9
10 int main (void)
11 {
12
13
14   serial_configura (PIN_1, PIN_2); // vai usar porta serial
15   printf("RODANDO...\r");
16
17
18   char buffer[70],a;
19   ARQUIVO* A=NULL;
20
21
```

```
  printf("Tecle [ENTER] para iniciar\r");
  while (1)
  {
    a=getchar();
    if (a==13) break;
  }
  printf("OK \r");

  inicializa_SA();              // vai usar o SA


  cria_SA(FORMATA_LOGICO);    // formata (cria) o sistema de arquivos
                 //
  A = meu_fopen ("UFSC","w");
  strcpy(buffer, "Oi mundo, teste de mensagemXXX");
  meu_fwrite(A,(uint8_t *)buffer, strlen(buffer));
  meu_fclose(A);

  A = meu_fopen ("Fabio","w");
  strcpy(buffer, "Oi mundo, teste de mensagemXXX");
  meu_fwrite(A,(uint8_t *)buffer, strlen(buffer));

  strcpy(buffer, " Prof: FABIO");
  meu_fwrite(A,(uint8_t *)buffer, strlen(buffer));

  strcpy(buffer, " de la Rocha UFSC Ararangua");
  meu_fwrite(A,(uint8_t *)buffer, strlen(buffer));



  meu_fclose(A);


  A=meu_fopen("Fabio","r");
  meu_fseek(A,19,SEEK_SET);
  meu_fread(A,(uint8_t *)buffer,8);
  buffer[8]=0;
  printf("Lido=%s\r",buffer);



  lista_todos();

  while(1)
  {
  }
  return 0;

}
```

1.3 Apresentação:

A apresentação pode ser realizada em grupos de no máximo 4 alunos que defenderão seu projeto no laboratório para o professor. A defesa será usando o Linux e também usando o kit ARM.

Data: até dia 29/06/2018.