Escola Politécnica da Universidade de São Paulo
Departamento de Engenharia de Sistemas Eletrônicos - PSI

PSI-2553- Projeto de Sistemas Integrados

Experiência 4: Desenvolvimento de um Sistema Embutido com Cálculo de Número de Fibonacci por Hardware (Prática)

M.S. / W.J.C / M.A.R.J/ E.C.V. (17)

Conteúdo:

1. Objetivos

Esta experiência visa o desenvolvimento do software e montagem de um hardware com o sistema Plasma utilizando, como coprocessador, o Processador de Cálculo de Número de Fibonacci.

2. Parte Prática

Para esta experiência, o(a) aluno(a) é solicitado(a) a implementar uma nova plataforma Plasma, contendo a CPU, memória, interfaces, além de um novo bloco FIBONACCI, construído na Exp. 1. Este bloco substituirá o PERIF_EXP3 utilizado na Exp. 3, utilizando o seu endereçamento (0x20000070).

Com este novo hardware, deve-se realizar duas opções de operação distintas no Plasma, uma com a função Fibonacci realizada por software, como na Exp.2, e outra por hardware, com o bloco FIBONACCI. Em ambas, o conjunto total de tarefas a serem realizadas é o mesmo: um laço em que a cada iteração, a função Fibonacci é chamada e uma série de escritas em uma interface UART é feita. As duas formas de operação são:

1) Simulação no sistema Plasma de um software em que o cálculo do número de Fibonacci é feito por meio de uma sub-rotina (semelhante ao viso na Exp2). Apesar de que o hardware já possuirá as interfaces com o módulo FIBONACCI (em hardware), esta parte não terá efeito nenhum na computação do software. Todo o cálculo do Fibonacci ocorrerá com as instruções do processador do Plasma.
2) O cálculo do número de Fibonacci é feito pelo circuito projetado pelo aluno na Exp1. O aluno modificará o software da opção 1) para chamar o coprocessador FIBONACCI e, enquanto o cálculo do número de Fibonacci é nele realizado, o processador Plasma continuará executando, em paralelo, outras tarefas do software, como a escrita na interface UART.

2.1 Ponto de Partida

É dado aos alunos o sistema Plasma em **Rede\ NEWSERVERLAB\psi2553\exp4**. Ele é diferente daqueles usados nas experiências anteriores, portanto, abra uma pasta sua **~\psi2553\exp4** para evitar misturas indevidas. O software e o hardware que são fornecidos aos alunos, são apresentados com mais detalhes nas seções seguintes.

2.1.1 Software

1) Todo o software inicial fornecido está no arquivo **Rede\ NEWSERVERLAB\psi2553\exp4\exp4_tools.zip**, que contem os códigos de programa, em forma similar aos das experiências anteriores.
2) Observações em **plasma_fibonacci.c**:
    - O programa *main* contém um laço que chama, por várias iterações, a rotina de cálculo do número de Fibonacci.
    - Repare que as rotinas de escrita de *strings* (puts) e de escrita de caracteres (putchar) são utilizadas. Entenda todas as rotinas que estão em **no_os.c**, inclusive a de interrupção.

- Observe que há uma rotina de conversão de número para string (para um número limitado de caracteres, implementada no mesmo arquivo **plasma_fibonacci.c**, e que é usada em main.
- Toda escrita (strings ou números) é feita para o porto serial (UART). Observe que as rotinas de escrita utilizam alguns endereços da UART – procure entendê-los com a teoria dada na Exp.3 e com o arquivo **plasma.h**.
- Uma característica importante da computação em software é a sua total sequencialidade quando executado pelo processador do Plasma, pois esta execução é feita instrução a instrução. Entenda toda a sequência de eventos previstos no software.
- Nas iterações do laço, o número de Fibonacci é calculado várias vezes para diferentes valores de entrada. O programa usa dois números de referência que o(s) aluno(s) da dupla deverá(ão) **trocar** pelos resultantes de operação com seu(s) número(s) USP, da seguinte forma:

  number_m= número_USP_aluno_1 mod 41.

  number_n= número_USP_aluno_2 mod 41.

  Sendo dois alunos, number_m é o maior e number_n, o menor.

  Depois, aplique a seguinte regra:

  a) se (number_m < 30) = TRUE, faça number_m = number_m +10 até que FALSE;

  b) se (number_n > 15) = TRUE, faça number_n = number_n -10 até que FALSE.

2.1.2 Hardware

Todo o hardware fornecido está em **Rede\ NEWSERVERLAB\psi2553\exp4\hw_arquivos**. Trata-se de mais uma versão do sistema Plasma, modificado para ser utilizado com o módulo Fibonacci projetado pelos alunos.

O arquivo **plasma.vhd** apresenta diversas modificações, sendo as mais importantes:

1) Introdução do módulo FIBONACCI- a figura do sistema Plasma é mostrada na Figura 1 abaixo, detalhando-se como com o registrador PERIF_EXP4 do módulo FIBONACCI (0x20000070) é acessado. Na realidade, no caso do FIBONACCI, este registrador é "virtual", no sentido de que quando uma escrita ou leitura é requisitada neste endereço, ocorre uma tarefa de decodificação e, com ela, a geração dos sinais de controle *enable_perif_exp4_read* e *enable_perif_exp4_write*, que solicitam a leitura e escrita, respectivamente, ao periférico, dos dados nos buses *data_r* ou *data_w*. O registrador corresponde às saídas e entradas dos módulos FIBONACCI construídos pelos alunos;
2) Há o acréscimo de três adaptadores (adaptador0, adaptador1 e adaptador2) que introduzem registradores para acerto de sincronização, como mencionado na apostila de teoria. O adaptador0 instancia o bloco wrapper0, que cuida do registro do dado que vem da CPU, no bus *data_w* e que será consumido pelo módulo FIBONACCI. A figura 2 mostra o registrador já com os nomes reais encontrados no **plasma.vhd**.
3) Os adaptador1 e adaptador2 instanciam o bloco wrapper1, que cuida do registro dos sinais de controle que vem da CPU e que serão consumidos pelo módulo Fibonacci. A figura 3 mostra o registrador já com os nomes reais encontrados no **plasma.vhd**.

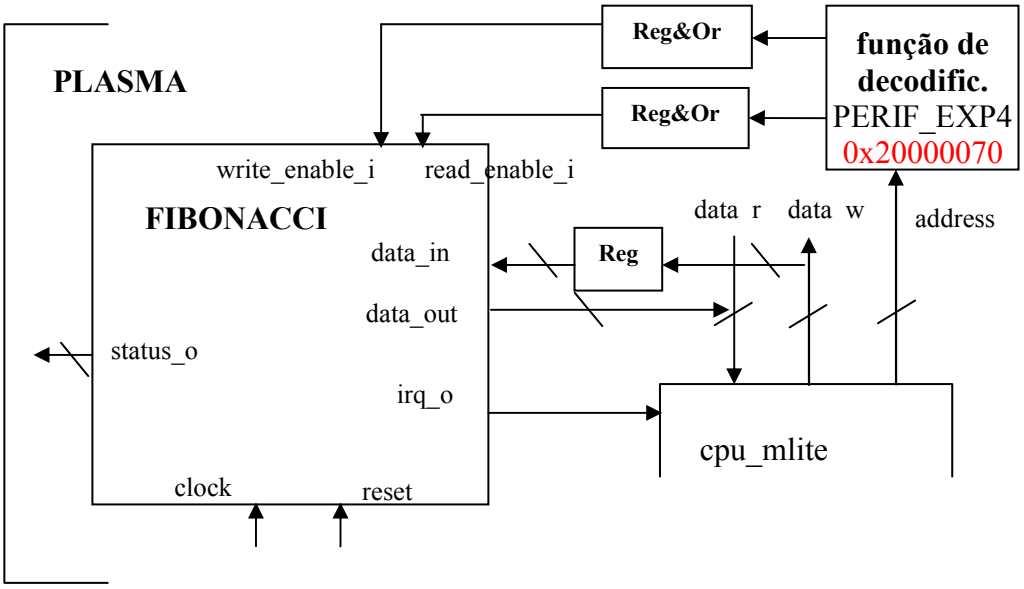


Figura 1. Esquema com a inclusão do Modulo FIBONACCI

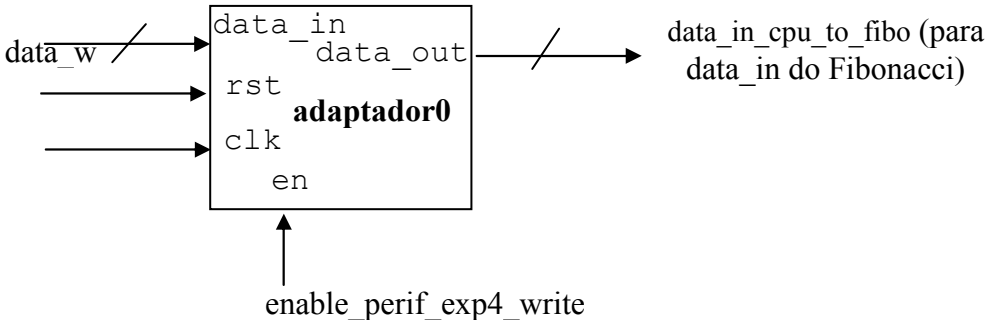


Figura 2. Adição de flip-flop para write no FIBONACCI

4) Mudanças foram necessárias para o porto serial (UART) na escrita. Existe um sinal que chega do módulo UART (**uart.vhd**), *uart_write_busy*, afetando diretamente os sinais *mem_pause* e *irq_status*, que indica que o UART está já em escrita e não aceita nova escrita (até terminar a sua tarefa). Para simplificar (já que não temos um módulo externo recebendo os dados da UART), assumimos que será sempre `0`, ou seja a porta serial sempre está disponível para receber novos dados.

   Atenção: *uart_write_busy* sempre ativado significa que a UART está também em todos os momentos solicitando interrupção, pelo IRQ_STATUS. Como forma de compensar esta situação, o IRQ_MASK não poderá conter ´1´ no bit referente ao IRQ_UART_WRITE_AVAILABLE (2o. bit da direita para a esquerda), senão o sinal irq do Plasma estará sempre ativado.

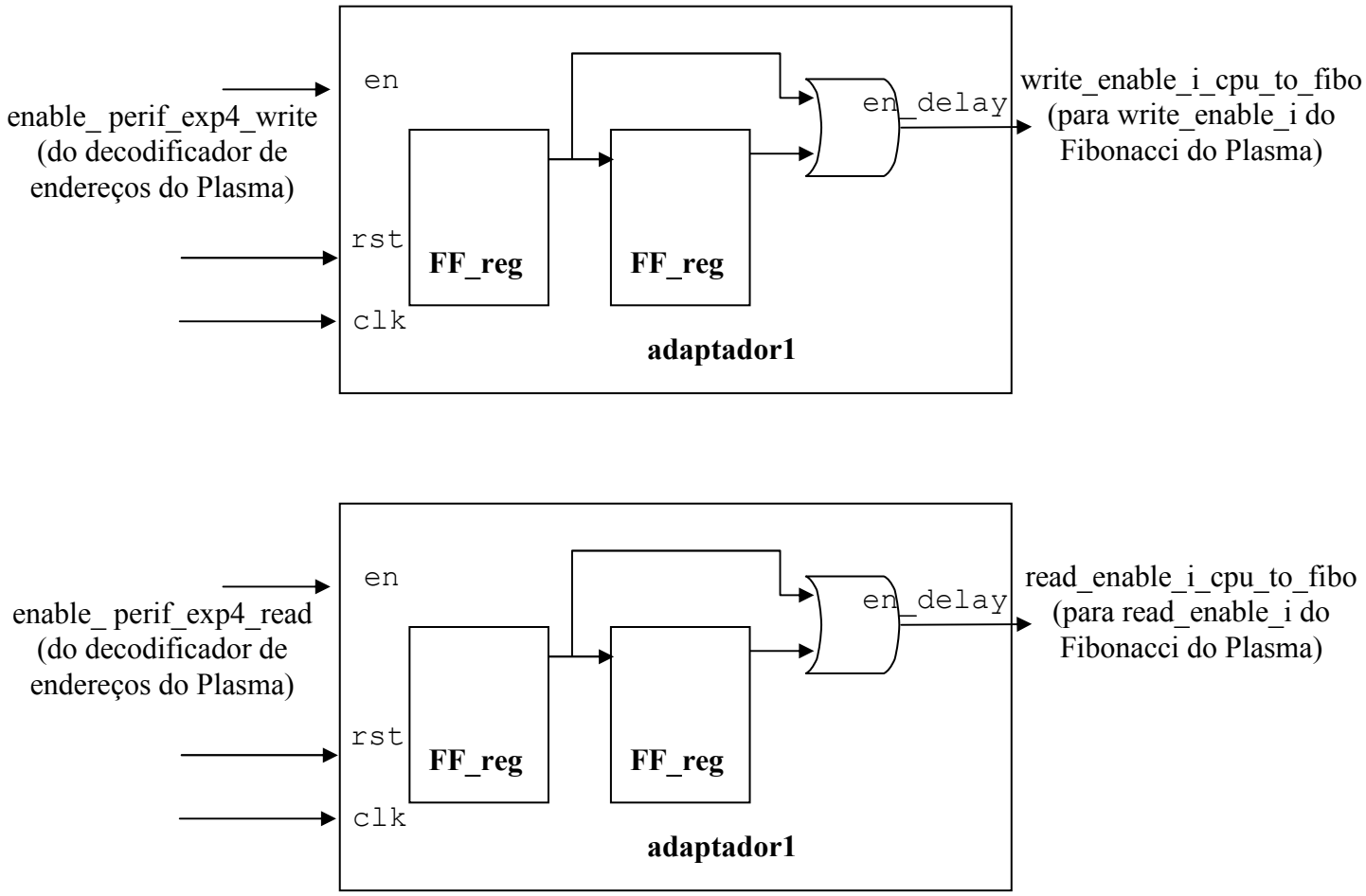


Figura 3. Adição de flip-flops para write e read

2.2. Adicionar o Módulo Fibonacci

O sistema Plasma já está pronto para incluir o seu módulo FIBONACCI. Os alunos deverão no plasma.vhd adicionar o componente e instanciá-lo na arquitetura. Os sinais que acessam o módulo FIBONACCI têm os nomes abaixo dentro do Plasma (associados aos prováveis nomes que você usou para o seu módulo FIBONACCI)

- irq_o_fibo_to_cpu ⇔ irq_o
- write_enable_i_cpu_to_fibo⇔ write_enable_i
- read_enable_i_cpu_to_fibo⇔ read_enable_i
- data_in_cpu_to_fibo ⇔ data_in
- data_o_fibo_to_cpu ⇔ data_o
- OPEN ⇔ status_o

Depende como o(a) aluno(a) implementou as interfaces de seu módulo, poderá haver alguma incompatibilidades de tipos. Caberá a ele(a) ajustar a interface.

2.3. Primeira Simulação- Fibonacci por Software

O aluno deverá realizar a simulação do código em software fornecido com o hardware fornecido, com o objetivo de se realizar os cálculos de número de Fibonacci por software. Quando rodar o sistema no Quartus, observe os dados de Fibonacci na carta do tempo e os resultados. Veja na Seção 2.5 o que você precisará apresentar. Observações:

- Recomenda-se aos alunos que criem, desde o começo, duas pastas tools separadas para os seus softwares: uma para o caso de software "puro", deste item, e outro para o caso do software usando o coprocessador Fibonacci projetado. Ao final da experiência, uma comparação deverá ser feita entre os dois casos.
- Para a simulação com testbench é dado um arquivo *inicio.txt* de configuração para a simulação. Este arquivo já contém sinais para observação da simulação para o caso de software visando a interrupção com o coprocessador FIBIONACCI. Estes sinais não terão significado nesta primeira simulação.

2.4. Novo projeto

2.4.1 Mudanças no Hardware

Todas as modificações em relação ao Plasma original foram realizadas de acordo com o **Guia de Modificações no Plasma**, fornecido na exp3, porém adaptadas para o processador FIBONACCI. Demais modificações estão listadas na Seção 2.1.2.

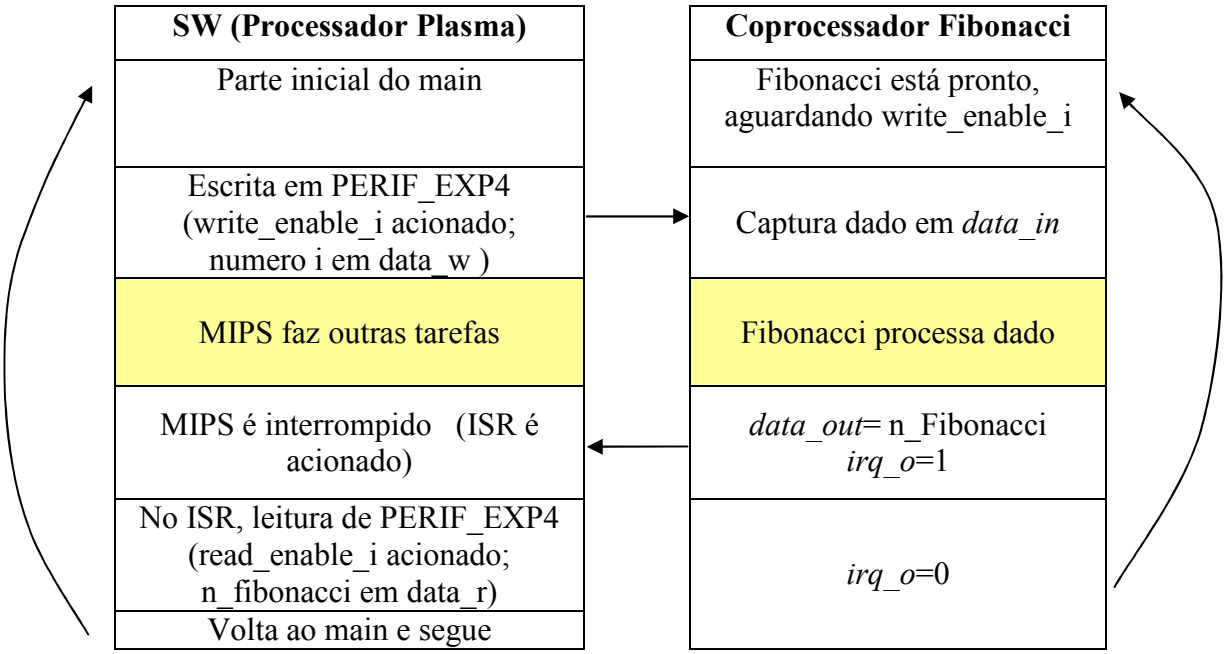
2.4.2 Mudanças no Software

1) Observe o código da rotina de interrupção *OS_InterruptServiceRoutine*, em **no_os.c**. Siga-o passo a passo para ter certeza que a entendeu.
2) O software original do *main* deverá ser modificado para que exista uma "chamada" ao coprocessador FIBONACCI ao invés de chamada à sub-rotina. A "chamada" ao coprocessador poderá ser feita por simples escrita no endereço de memória associado ao periférico.
3) A modificação acima deve ser feita de forma que haja concorrência entre o cálculo do número de FIBONACCI e as escritas diversas de strings via UART (são as células com fundo sombreado do diagrama abaixo. Estude bem a sequência de escritas na UART para saber quais são aquelas que dependem e as que independem do resultado do cálculo do número de Fibonacci. As que dependem são sequenciais à operação do coprocessador enquanto as que independem podem ser concorrente a tal operação.
4) Veja que a OS_InterruptServiceRoutine é chamada quando o módulo FIBONACCI termina a sua tarefa de cálculo de número Fibonacci e a interrupção é gerada. Dentro da ISR é realizada a leitura do resultado do cálculo do número de Fibonacci pelo endereço PERIF_EXP4.
5) Após a leitura acima, o valor deverá ser guardado em posição de memória dada por

   #define PERIF_EXP4_RAM_WRITE 0x0000Fxyz ,

   onde xyz deve ser os três últimos algarismos de seu número USP (o maior dos dois alunos) (se necessário, z deve ser aproximado para o correto alinhamento de dados na memória).

   Após retornar ao main, o valor de n_fibonacci guardado em 0xFxyz é lido e consumido.



6) O aluno deverá utilizar um **flag** para implementar um esquema de sincronização entre o *main* e o ISR para cuidar do caso em que as tarefas de escrita de *string* do *main* terminem antes que o número de Fibonacci seja calculado. Quando isto ocorre o main deve ficar em um estado de espera (idle) até que o coprocessador FIBONACCI termine de calcular o número de Fibonacci e a interrupção seja gerada. No caso oposto não haveria problema, pois enquanto o *main* ainda estivesse realizando a escrita na UART, a interrupção ocorreria, e o valor de n_fibonacci seria capturado em PERIF_EXP4_RAM_WRITE na ISR. A interação entre o **main** e o **OS_InterruptServiceRoutine** deve ser da forma descrita na figura 4, onde respectivas máquinas de estados são apresentadas.

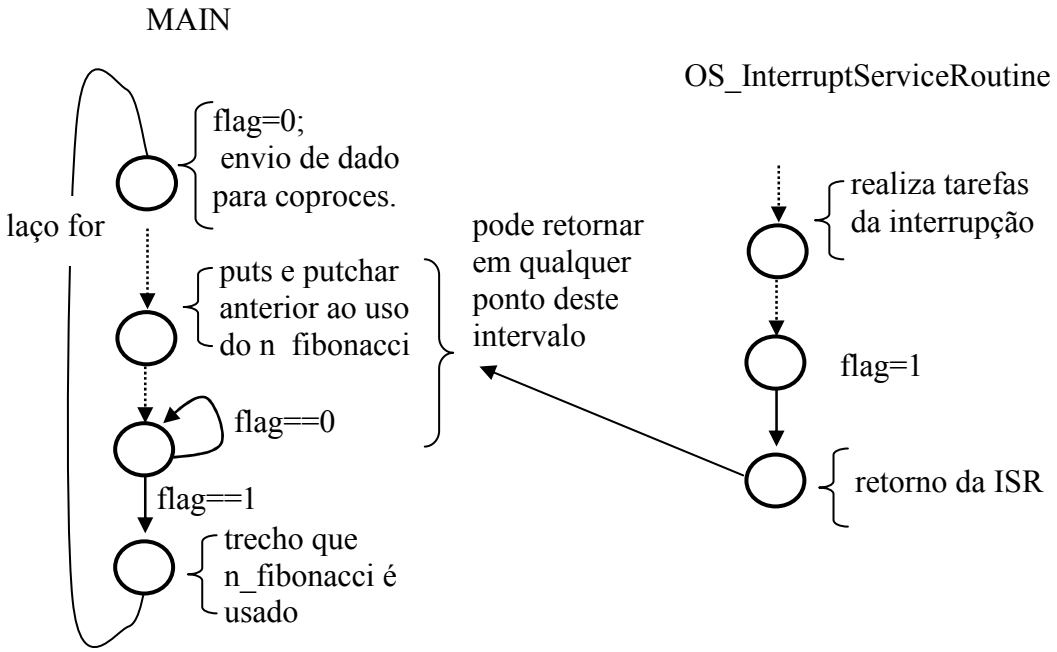


Figura 4. Adição de flag para sincronizar o uso do resultado do coprocessador Fibonacci

Neste caso deve haver um flag (um dado em memória; o aluno pode designar, por exemplo, um endereço anterior ao PERIF_EXP4_RAM_WRITE do item 5 (0x0000Fxyz-4) acionado na ISR que indicará que a interrupção já tenha sido feita e o número Fibonacci já tenha sido armazenado. Quando as tarefas de escritas de string terminam, este *flag* será checado –caso ainda inativo (não ocorrera a interrupção ainda), o programa *main* deve ficar em laço até que o *flag* fique ativado.

2.5 O que apresentar

Cada aluno/dupla deverá realizar as mudanças todas, compilar/montar o software e realizar as simulações no Quartus demonstrando que a computação esteja sendo feita corretamente. Não há necessidade de relatório impresso. Entretanto, os alunos deverão **incluir as anotações** e **explicações solicitadas**, da forma que achar mais conveniente. O aluno deverá enviar todos os dados por email.

2.5.1 Hardware

Envie todos os arquivos ***.vhd** referentes ao seu coprocessador fibonacci que tenha incorporado à pasta hw_arquivos. Além disso, envie o **plasma.vhd** com a inclusão do seu módulo fibonacci. Tenha certeza de que são os arquivos corretos, pois serão testados para a simulação.

2.5.2 Primeira Simulação- Fibonacci por software

1) O arquivo **plasma_fibonacci.c** modificado com os números USP. Tenha certeza de que seja o correto, pois a sua compilação será testada.
2) Os arquivos **test.map** e **test_lst** gerados.
3) Os executáveis code_0 a code_3.
4) Arquivo inicio.txt, caso tenha feito mudanças.
5) Cartas de tempos onde fique evidente (faça **marcações/anotações**):
   a. início do main (primeira vez)- anote o tempo.
   b. início da chamada à rotina fibonacci, na primeira vez.
   c. identifique o envio do number_n inicial à rotina fibonacci.
   d. identifique o envio do number_m à rotina fibonacci
   e. identifique na rotina fibonacci, o número Fibonacci calculado (do item c).
   f. identifique na rotina fibonacci, o número Fibonacci calculado (do item d).
   g. mostre a escrita UART (string) do resultado do item d (mostre a sequencialidade de toda a escrita).
   h. Saída do main (última vez)- anote o tempo.

2.5.3 Segunda Simulação- Fibonacci por hardware

1) os módulos **plasma.h**, **no_os.c** e **plasma_fibonacci.c** modificados. Tenha certeza de que sejam os corretos, pois a sua compilação será testada**.**
2) Os arquivos **test.map** e **test_lst** gerados.
3) Os executáveis code_0 a code_3.
4) Arquivo inicio.txt, caso tenha feito mudanças.
5) Cartas de tempos onde fique evidente (faça **marcações/anotações**):
   a. início do main (primeira vez)- anote o tempo.

b. início da chamada do módulo FIBONACCI, na primeira vez.
c. mostre o início de uma interrupção.
d. identifique o envio do number_n inicial ao co-processador FIBONACCI
e. identifique o envio do number_m ao co-processador FIBONACCI
f. identifique na ISR, a leitura do número Fibonacci calculado (do item d)
g. identifique na ISR, a leitura do número Fibonacci calculado (do item e)
h. identifique em main, a leitura do número Fibonacci calculado (do item d), em PERIF_EXP4_RAM_WRITE.
i. identifique em main, a leitura do número Fibonacci calculado (do item e), em PERIF_EXP4_RAM_WRITE.
j. mostre a escrita UART (string) do resultado do item d (aponte a interrupção da sequência de toda a escrita devido ao acesso ao número de fibonacci por hardware).
k. Saída do main (última vez)- anote o tempo.

2.5.4 Comparação e Justificativas

Em um arquivo ou anotação em separado, anote o tempo de computação do item a) até a ocorrência do item 2.5.1.g e 2.5.2.j e compare-os. Justifique os resultados obtidos (lembre-se do problema de troca de contexto com a interrupção).