UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS ARARANGUÁ

# RELATÓRIO DO TRABALHO 4

IGOR FIEDLER DE BASTOS E VINÍCIUS DOMINGOS GABRIEL

2025

# INTRODUÇÃO

Neste relatório, temos como objetivo abordar o processo e problemas que enfrentamos ao produzir e simular o trabalho 4 de linguagem e descrição de hardware. O trabalho é dividido em 4 partes, sendo:

1. Simular o programa de soma de vetores  no processador MIPS_S_Prt com memórias do tipo BRAM.
2. Escrever um software contador de segundos que conte de 000,0s a 999,9s.
3. Acrescentar um periférico de hardware ao sistema MIPS_S_withBRAMs.
4. Prototipar o sistema completo

A seguir detalharemos todo o caminho para realizarmos esses 4 processos.

**Nota sobre a Organização dos Arquivos**: Para facilitar a avaliação e a replicação dos testes, os arquivos VHDL de suporte, como o driver de display (**dspl_drv**) e o componente de periférico (**periferico_display.vhdl**), foram agrupados no diretório **vhdls_auxiliares**. Adicionalmente, as diferentes versões do componente **memory.vhd**, geradas a partir do software **contador.asm** com os respectivos valores de delay para o teste rápido de validação e para a contagem precisa em hardware, foram preservadas e estão disponíveis no diretório **t4_p2**.

# PROJETO 1

O propósito fundamental desta etapa foi a familiarização com o fluxo de trabalho para simulação de software em um processador descrito em VHDL.

Utilizamos o software MARS e com o auxílio do documento **"SIMULAÇÃO VHDL DO PROCESSADOR MR2"**, a simulação foi executada, permitindo a observação do comportamento do processador ciclo a ciclo. A validação foi concluída ao verificar que os resultados da soma foram corretamente armazenados na memória de dados, confirmando o funcionamento adequado do *datapath* e da unidade de controle do processador.

Esta fase foi concluída com sucesso e serviu como base essencial para as etapas seguintes, garantindo o domínio sobre o ambiente de simulação e o processo de carga de software na memória do sistema.

## PROJETO 2

O objetivo era desenvolver um software em Assembly MIPS para implementar um contador com precisão de décimos de segundo, operando no intervalo de 000,0s a 999,9s.

Utilizando novamente o MARS, foi desenvolvido o código Assembly para o contador. O principal desafio foi a implementação da lógica de controle e exibição em Assembly, uma linguagem de baixo nível que exige gerenciamento manual de registradores e memória. Inicialmente, houve dúvidas sobre o funcionamento das saídas do sistema. Com o auxílio do professor, foi esclarecido que o software deveria escrever os dados a serem exibidos em endereços de memória específicos, que o hardware do periférico (a ser desenvolvido na etapa 3) ficaria responsável por ler e interpretar.

A simulação no MARS foi crucial para depurar a lógica do contador antes mesmo de sua integração com o hardware VHDL.

## PROJETO 3

O objetivo central desta etapa foi projetar e implementar um periférico em VHDL para servir como interface entre o processador MIPS e o hardware de exibição. A função essencial deste componente é permitir que o software, executado no processador, controle o que é exibido no display de forma indireta e organizada.

Para alcançar o objetivo, foi desenvolvido um componente de hardware descrito no arquivo periferico_display.vhdl.

O funcionamento do periférico se baseia nos seguintes passos:

1. Monitoramento do Barramento: O periférico monitora continuamente o barramento de endereços do sistema MIPS.
2. Detecção de Endereço: Ele foi projetado para reconhecer e responder a operações de escrita em endereços de memória específicos. Conforme programado no software em Assembly, os endereços designados para esta comunicação são:
    - 0x1008000
    - 0x1008001
    - 0x1008002
3. Captura de Dados: Quando o processador MIPS executa uma instrução para escrever um valor em um desses endereços, o periférico captura o dado correspondente do barramento de dados do sistema.
4. Armazenamento e Transmissão: O valor capturado é armazenado em um registrador interno do periférico e, em seguida, transmitido para o driver do display, que o converte nos sinais necessários para a exibição física.

O fluxo de informação pode ser resumido da seguinte forma:

- O software no MIPS calcula o valor a ser exibido.
- O software escreve esse valor em um endereço de memória conhecido (ex: 0x1008000).
- O periférico VHDL, que "escuta" esse endereço, lê o valor.
- O periférico joga esse valor para o driver do display.

O resultado foi um módulo de hardware funcional que se integrou com sucesso ao processador MIPS. Ele cumpriu seu papel de ler os dados dos registradores mapeados em memória e repassá-los ao destino final. Esta etapa foi crucial para validar o conceito de co-design hardware/software, onde ambos os componentes são projetados para interagir de forma coesa e eficiente dentro de um mesmo sistema.

# PROJETO 4

O objetivo era sintetizar e implementar o design VHDL completo em uma placa FPGA Nexys e validar seu funcionamento no mundo real.

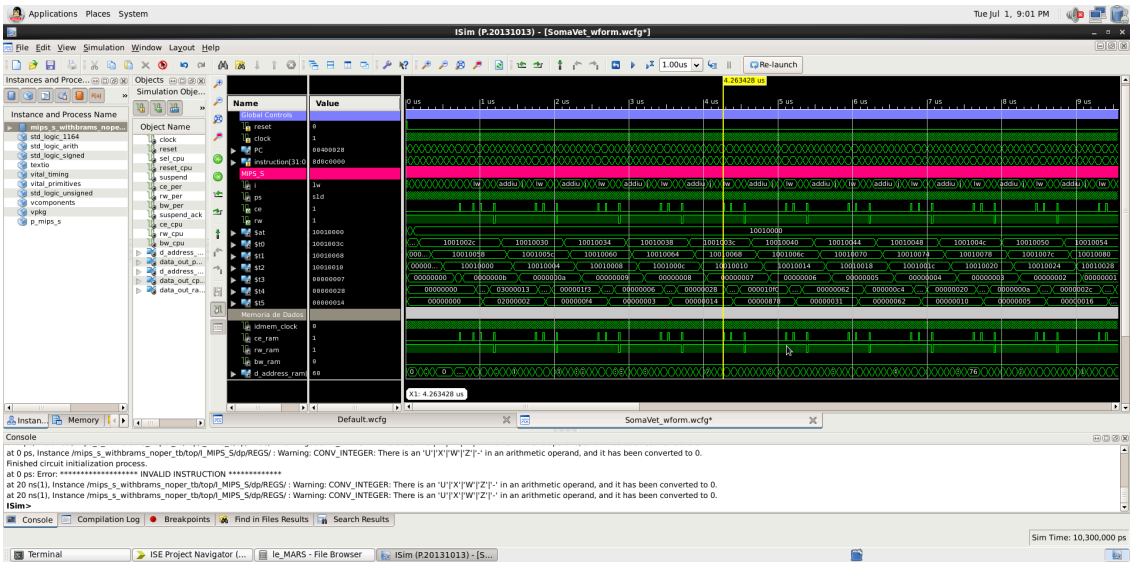
- **Arquivo de Restrições (.ucf)**: Um arquivo .ucf foi criado para mapear as portas da entidade de topo (clock, reset, an, dec_ddp) aos pinos físicos correspondentes da placa.
- **Depuração Hardware vs. Simulação:** A validação inicial na placa revelou que o display não acendia, apesar do sucesso na simulação. Um processo de depuração sistemático foi iniciado. Um teste simplificado ("Hardware Hello, World") foi criado para isolar o problema. Utilizando um projeto funcional como referência, foi descoberto que o driver de display (dspl_drv) esperava um "protocolo" específico em suas entradas de dados de 6 bits, onde o bit mais significativo (bit 5) funcionava como um sinal de habilitação para cada dígito. O design top_conts.vhd foi corrigido para enviar este bit de habilitação, resolvendo o problema.

Após a correção final e a geração de um novo arquivo .bitstream, o sistema foi programado com sucesso na placa FPGA. O protótipo operou exatamente como o projetado: os displays de 7 segmentos exibiram o cronômetro, iniciando em 000.0 e incrementando a cada 0.1 segundo real, parando em 999.9, demonstrando a correta e funcional integração entre o software executado no processador soft-core e o periférico de hardware customizado.

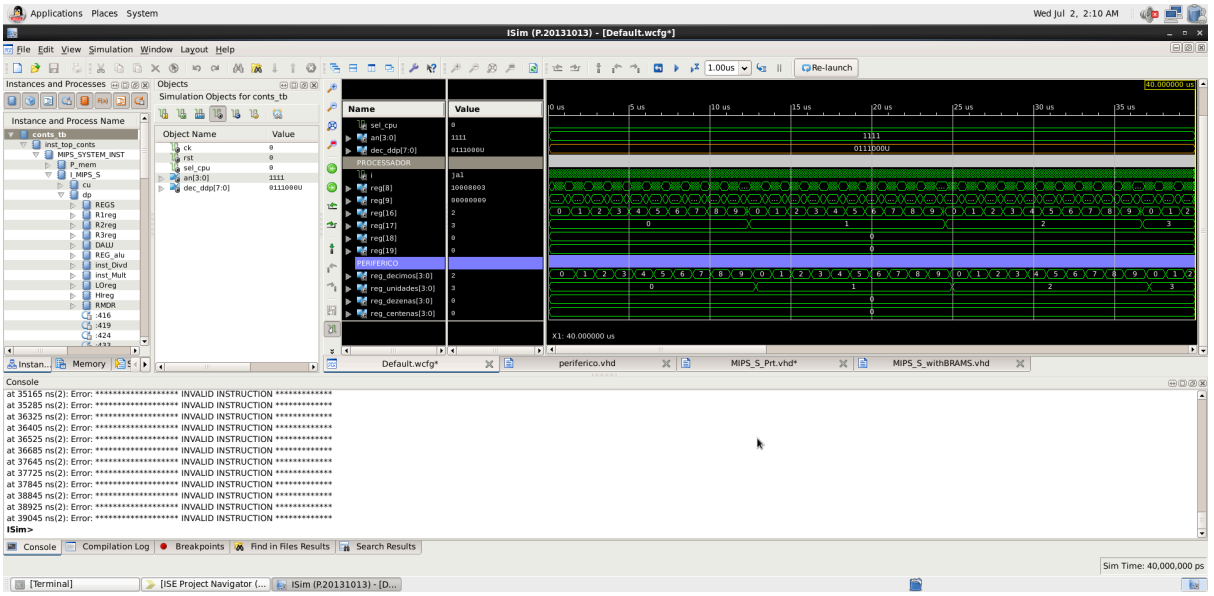
# SIMULAÇÃO E PROTOTIPAÇÃO

As imagens a seguir demonstram o funcionamento dos projetos.

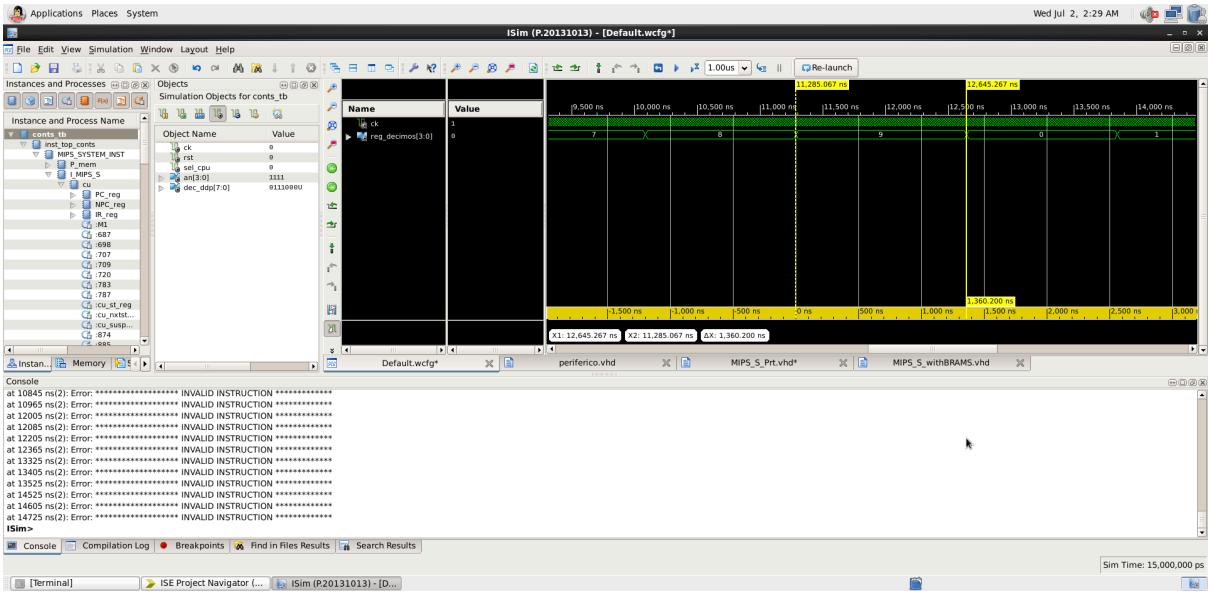
## Projeto 1:



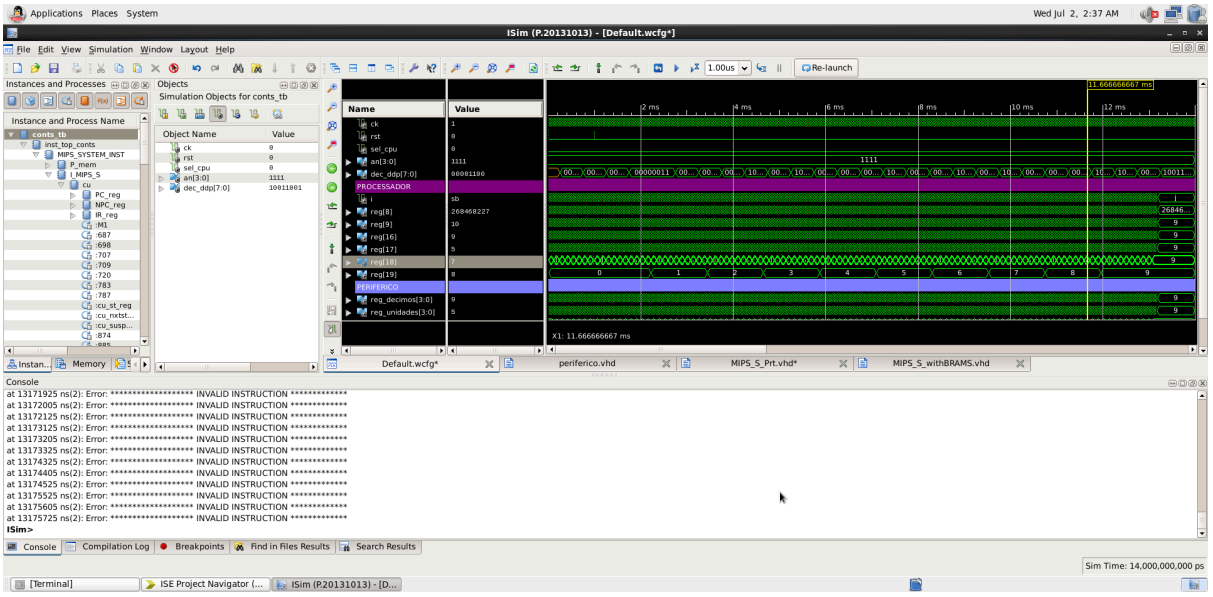

## Projeto 3:



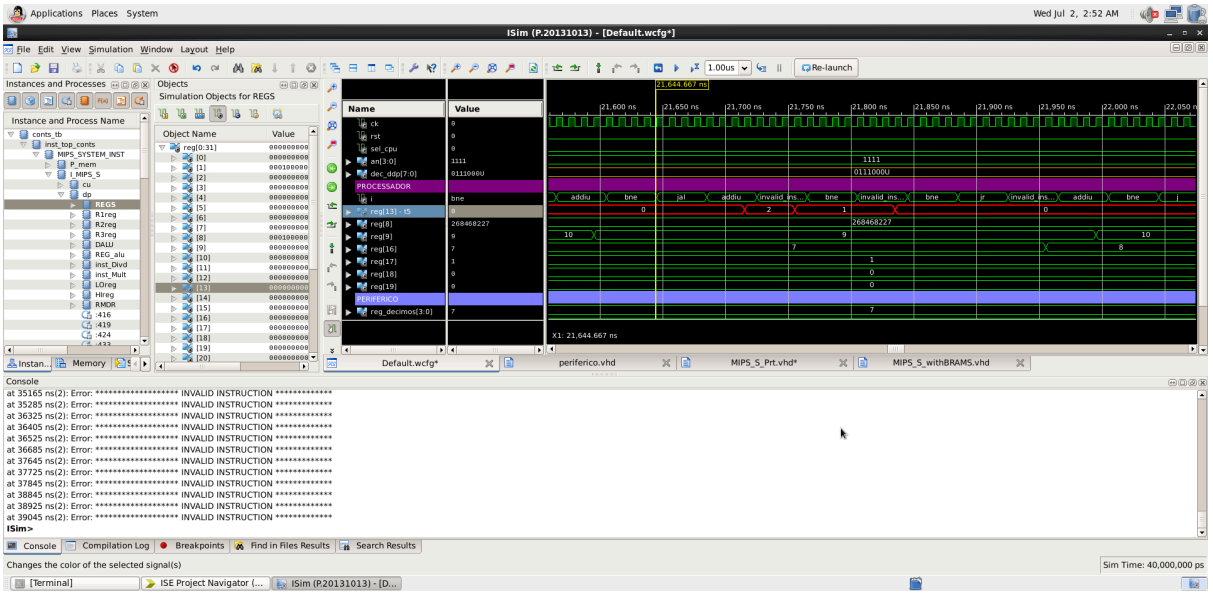
Clock de décimo:




Simulação final do p3:



Projeto 4:




Um vídeo com o funcionamento do projeto 4 na placa foi gravado e upado no youtube e na pasta do trabalho 4.

Link: https://www.youtube.com/shorts/vUkZkmK2gvE

# CONCLUSÃO

A execução dos quatro projetos proporcionou uma experiência completa e prática no ciclo de vida de um sistema embarcado. O trabalho solidificou conceitos de programação em Assembly MIPS, projeto de hardware síncrono em VHDL e, de maneira mais significativa, as técnicas e desafios envolvidos na depuração da interface software-hardware. A superação dos problemas encontrados, desde o ajuste fino de tempo do software até a depuração de protocolos de comunicação em nível de hardware, resultou na entrega de um protótipo funcional e no cumprimento de todos os objetivos propostos.