UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA
SISTEMAS DIGITAIS E DISPOSITIVOS LÓGICOS RECONFIGURÁVEIS

DUAL-LOCKSTEP PIPELINE RISC-V

por

Felipe Viel

Florianópolis (SC), setembro de 2021

UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA
SISTEMAS DIGITAIS E DISPOSITIVOS LÓGICOS RECONFIGURÁVEIS

DUAL-LOCKSTEP PIPELINE RISC-V

por

Felipe Viel

Relatório de projeto final para a disciplina de Sistemas Digitais e Dispositivos Lógicos Reconfiguráveis da Pós-Graduação em Engenharia Elétrica.
Professor Responsável: Eduardo Augusto Bezerra

Florianópolis (SC), setembro de 2021

1 PROJETO

1.1 VISÃO GERAL

O processamento de dados nas atuais arquiteturas de processadores permite alcançar desempenho em diversas frentes, como economia de energia, processamento paralelo, aceleração de algoritmos, entre outros. Além disso, em decorrência de falhas na concepção ou dependência de arquiteturas em sua grande maioria proprietárias, levam empresas e até mesmo nações a buscar alternativas. Levando isso em conta, nascem novas propostas de arquiteturas que visão sanar diversos dos pontos citados acima, tendo eles como problemas. Nesse contexto, surge a arquitetura RISC-V, a qual ganha cada vez mais adeptos, tanto na indústria quanto de nações, diante dos contextos atuais de embargos tecnológicos.

A arquitetura RISC-V é uma ISA (Instruction Set Architecture) que vem sanar problemas principalmente de fragmentação que há hoje em arquiteturas, sem levar em conta os conceitos de arquiteturas CISC (Complex Instruction Set Computer) e RISC (Reduced Instruction Set Computer). A padronização, e união da comunidade, permite que haja uma melhor evolução da tecnologia e uma melhor logística de criação, desenvolvimento e integração de fornecedores globais. A ISA nasceu em 2010, sendo criada por Krste Asanović, Yunsup Lee e Andrew Waterman. Atualmente a ISA é mantida por uma organização, a RISC-V International, e fomentada por diversas empresas, como as gigantes Google, IBM, Cadence e Huawei, entre outras. A ISA RISC-V compete, hoje, diretamente com ISA ARM e, por assim se dizer, com Intel e AMD (dado pelo potencial futuro da arquitetura).

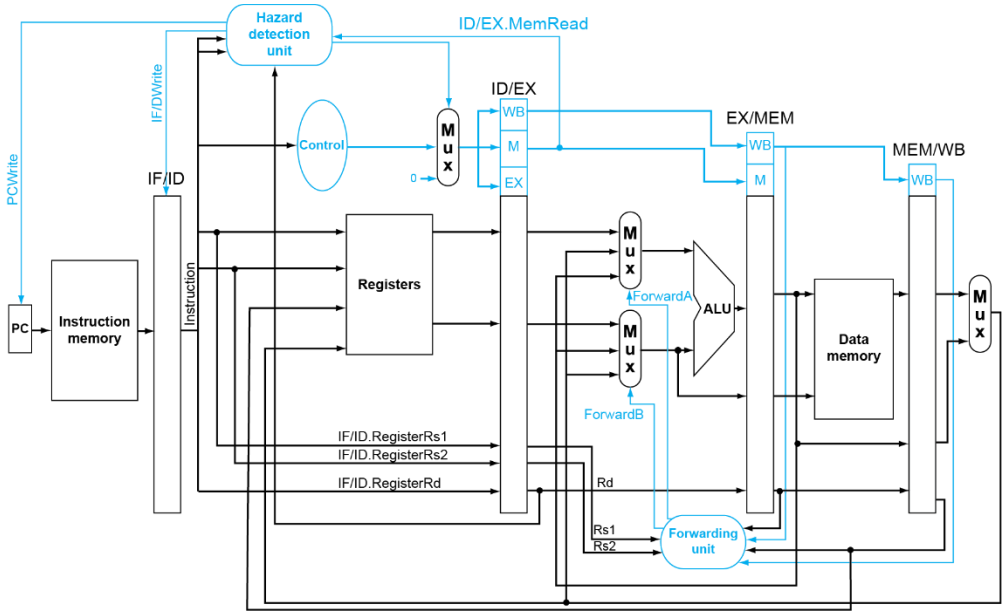

Figura 1. Organização do RISC-V completa (PATTERSON; HENNESSY, 2017)

A Figura 1 ilustra a organização do RISC-V com suporte a versão RV32I e implementação de *data hazards* Fowarding Unit (para suportar o redirecionamento de dados usados por instruções futuras) e Detection Hazard Unit (para detectar dependência de dados entre instruções novas e instruções em estágios avançados do pipeline).

1.2 PREMISSAS

As premissas são necessárias para nortear o projeto a ser desenvolvido.

1.2.1 Premissas para o SoC RISC-V

As premissas são:

- Será usando um núcleo RISC-V escrito em VHDL;
- O núcleo RISC-V será baseado na ISA RV32I;
- O núcleo usado implementa o tratamento de data hazard usando Forwarding Unit. Não é implementado a Hazard Detection Unit.
- Será usado dois núcleos replicados e iguais para implementar a técnica Dual-Lockstep;

- Será feito a detecção a partir da escrita e leitura das memórias e bacos de registradores, além do valor do PC (Program Counter);
- O detector de falha fará uma detecção *stateless*, onde o estado anterior e o futuro não importam, somente o valor atual dos dados monitorados;
- O detector compara os dados de mesma origem das duas replicações de núcleo;
- A detecção de valores desiguais acarreta em um reset dos processadores;
- Em caso de 7 detecções de erros de instrução, será usado multiplexado o valor de uma terceira memória que substituirá as duas memórias de instrução originais.

1.2.2 Análise de Requisitos

1.2.1 Requisitos funcionais para o SoC RISC-V

- RF01: O sistema deve suportar o conjunto de instruções RV32I;
- RF02: O sistema deve tratar *data hazards*;
- RF03: O sistema deve ser *dual-core*;
- RF04: O sistema deve ser detectar falhas de memorização nos núcleos;
- RF05: O sistema deve resetar os núcleos na presença de falhas;
- RF06: O sistema deve utilizar uma terceira memória de instrução em caso de 7 falhas detectadas;
- RF07: O sistema não deve guardar os estados dos dados nas operações monitoradas;

1.2.2 Requisitos não funcionais para o SoC RISC-V

- RNF01: O núcleo RISC-V usado é o Maestro;
- RNF02: O núcleo RISC-V é multi-ciclo com 5 estágios de pipeline;
- RNF03: A técnica de tratamento de *data hazards* é a Forwarding;
- RNF04: O SoC é escrito em VHDL;
- RNF05: A arquitetura do sistema é 32-bits;

- RNF05: Os núcleos estão sincronizados e compartilham o mesmo sinal de relógio (*clock*).

1.3.3 Regras de negócio para o SoC RISC-V

Nesta seção, é apresentada as regras de negócio do sistema a ser desenvolvido.

- RN01: Quando o detector indicar diferença entre um dos dados monitorados em ambos os processadores, deverá gerar uma sinalização de falha;
- RN02: A sinalização deverá, após detecção da falha, resetar os dois núcleos.
- RN03: Sempre que houver um reset por parte do detector, os núcleos deverão reiniciar todo o software.
- RN04: Com a ocorrência de 7 detecções de falha relacionados à memória de instrução, o detector deverá trocar, via multiplexação, para uma terceira memória de dados que servirá as instruções para os dois processadores.

1.3 Arquitetura e organização do SoC RISC-V

1.5.1 Organização do SoC RISC-V

A Figura 2 ilustra a organização básica de sistema Dual-Lockstep, sem considerar as implementações que são propostas neste trabalho. Na figura, pode-se observar os elementos básicos para a organização do SoC em questão. Os elementos presentes são os dois núcleos RISC-V, com as memórias de instrução e dados, e o detector/monitor de dados diferentes.

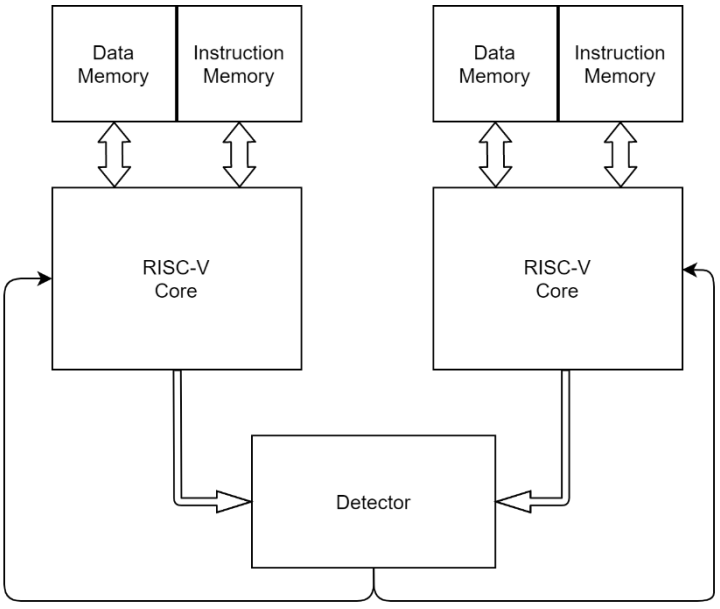

Figura 2. Sistema Dual-Lockstep para falhas com núcleos RISC-V

Visando atender uma gama de falhas, o sistema proposto neste trabalho visa detectar disparidades entre os núcleos, e suas memórias, no que diz respeito a dados que estão ou serão armazenados. Esse monitoramento acontece nas memórias de instrução e de dados, no banco de registradores e no PC. Na Figura 3, é apresentada a organização proposta neste trabalho.

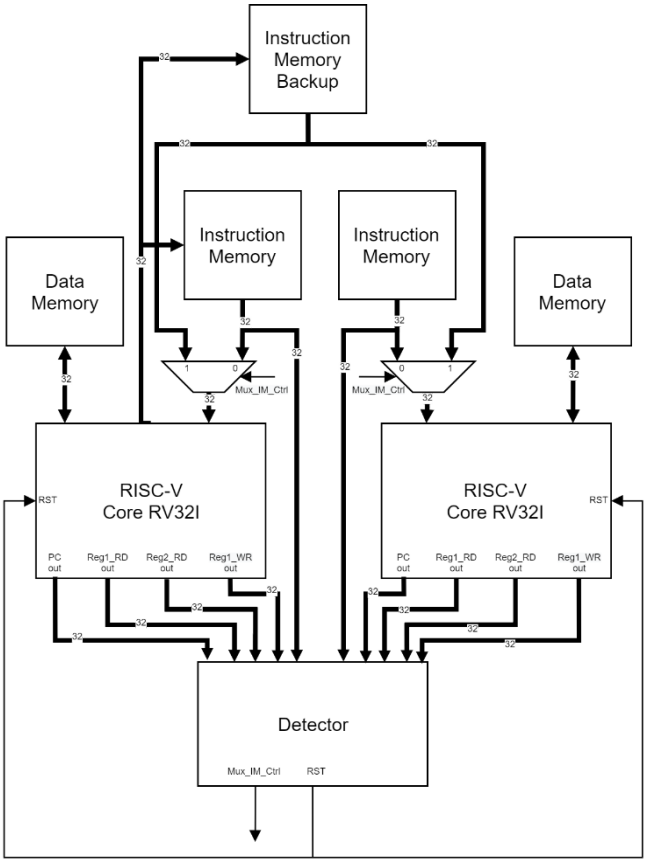

Figura 3. Sistema proposto com a abordagem Dual-Lockstep

Na Figura 3, apresenta-se a organização externa do sistema proposto, sendo possível observar o redirecionamento de dados vindos da comunicação com a memória e de internamento do processador. O detector monitora dados que são ou estão armazenados no banco de registradores, o valor do PC para busca de instruções e os dados que são buscados ou armazenados (no caso da memória de dados) nas memórias de cada núcleo do RISC-V. O detector ao verificar que há uma diferença entre os dados dos dois núcleos, irá gerar um sinal de reset (RST) em ambos os núcleos.

Além disso, como já citado, o detector irá observar se ocorrem mais de 7 ocorrências de diferença entre as instruções (armazenadas na memória de instrução) dos dois núcleos. Caso isso ocorra, ele irá utilizar as instruções armazenadas na memória de instruções de backup (INSTRUCTION MEMORY BACKUP) e colocar em “1” o seletor do multiplexador (Mux_IM_Ctrl).

A Figura 4 ilustra a organização do núcleo Maestro (CHRISÓSTOMO, 2018), o qual é um RISC-V de 5 estágios usado para este trabalho.

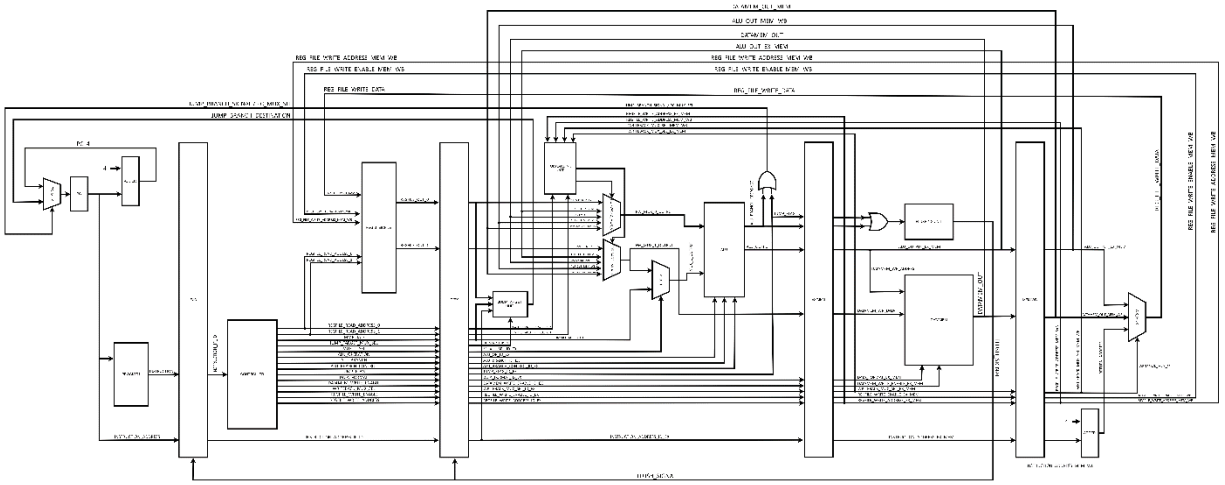

Figura 4. Núcleo RISC-V RV32I Maestro (CHRISÓSTOMO, 2018).

Os sinais que serão redirecionados para o detector, e presentes no projeto do Maestro, são:

- REGFILE_OUT_0;
- REGFILE_OUT_1;
- INSTRUCTION_ADDRESS;
- INSTRUCTION;
- DATAMEN_OUT; e
- DATAMEN_WR_DATA.

Deu-se prioridade para o tratamento de erros na memorização de dados por serem as primeiras e últimas etapas do processamento. O monitoramento dos registradores de pipeline não é implementado neste trabalho, já que os dados gerados no início e fim do processamento já cobrem falhas durante as etapas de pipeline. A Figura 5 apresenta, internamente, os sinais que são redirecionados para o detector. Essa figura é uma versão simplificada da Figura 4, visando apenas ilustrar a origem dos sinais utilizados pelo detector.

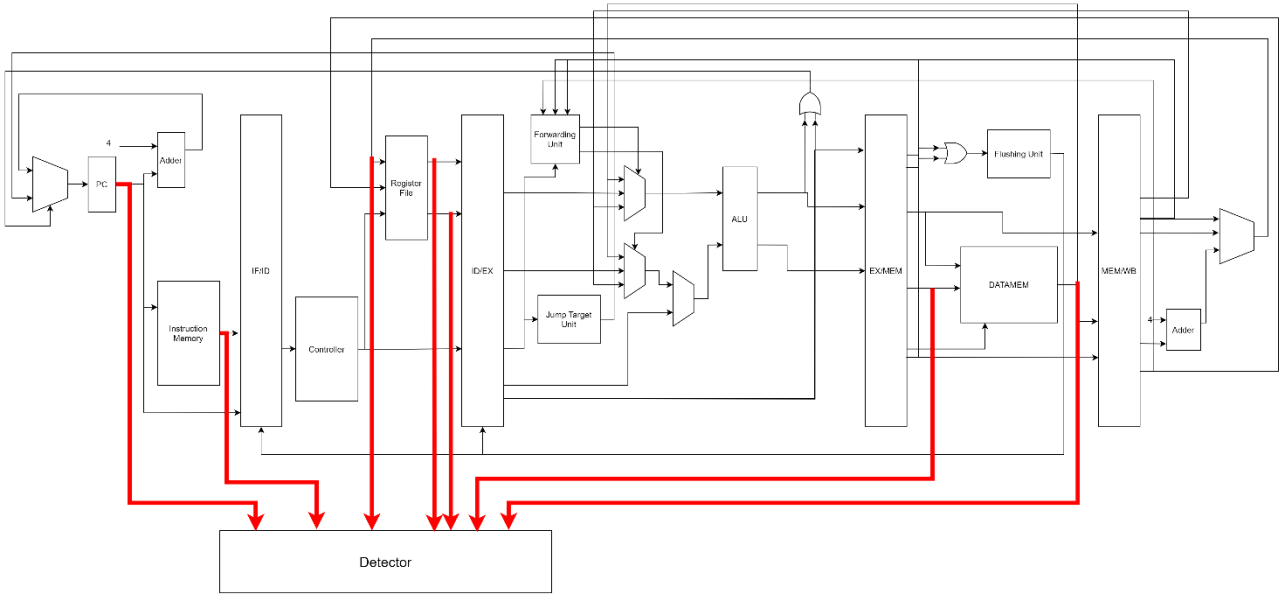

Figura 5. Núcleo RISC-V RV32I Maestro com os sinais de dados (em vermelho) usados pelo detector.

1.5.2 Arquitetura e organização do Detector

A Figuras 5 ilustra, por meio de máquina de estados, o comportamento do detector. Como o mesmo monitora quantidade de ocorrências falhas na memória de instrução, há uma necessidade uma maior inteligência no detector, o que exclui a possibilidade de o mesmo ser implementado via lógica puramente combinacional.

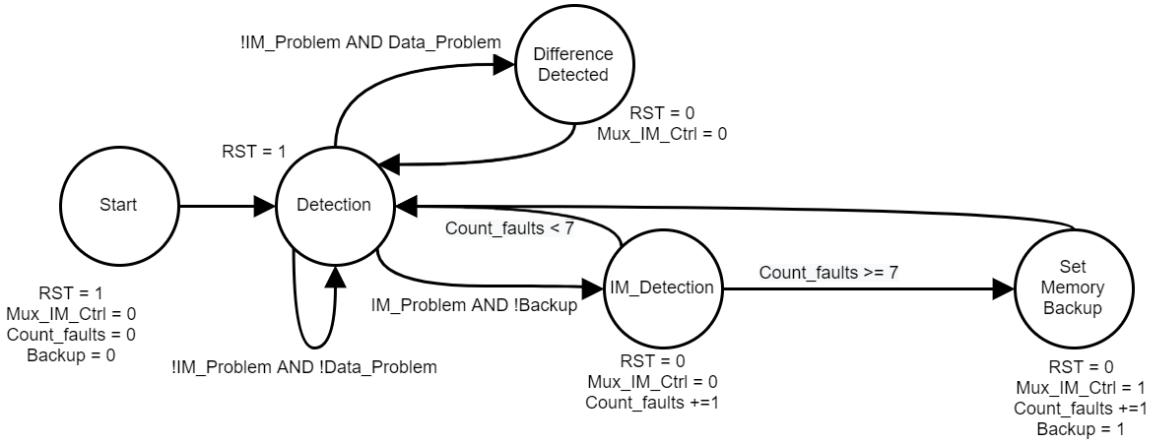

Figura 6. Máquina de estados finita para controle de execução do detector

O fluxo da máquina de estados indica um estado de inicialização (Start), um estado de constante monitoramento visando detectar problemas (Detection), um estado que trata problemas de erros em dados que não são instruções (Difference Detected), um estado que identifica erros de instrução (IM_detection) é um estado que identifica e trata erros mais de 7 ocorrências de erros em instruções (Set Memory Backup). A prioridade de execução entre os estados Difference Detected e IM_detection é do IM_detection. Caso haja 7 ocorrências de erros em instruções, uma flag é

levantada (Backup) e nunca mais entra-se nos estados IM_detection e Set Memory Backup, já que todos os dois núcleos receberão instruções da mesma memória.

A Figura 6 ilustra a organização interna do detector. Observa-se na figura a FSM (Máquina de Estados Finitos, bem como os registradores que permitem identificar e manter um controle de ocorrências dos erros nas instruções.

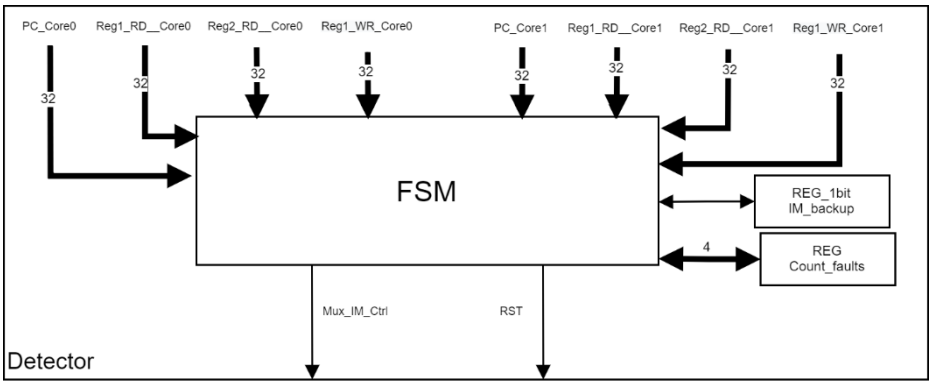

Figura 7. Organização Interna do detector

1.4 Ambiente de desenvolvimento e teste

1.4.1 Ferramentas

Buscando validar a implementação do modelo de sistema Dual-Lockstep proposto, será utilizado a ferramenta Siemens Modelsim HDL Simulator. A ferramenta permitirá validar e verificar todas as etapas de execução do sistema proposto.

A geração de softwares/instruções que serão usadas para validação do modelo será via software Ripes, o qual permite a compilação e geração de instruções para núcleos de diferentes estágios de pipeline e para diferentes arquiteturas.

1.4.2 Testes

Para realizar os testes, serão implementadas algumas injeções de falhas. Para falhas de instrução, será gerado dois softwares com diferentes instruções em alguns pontos do software. Nesse teste, também será testada a capacidade de chavear para a memória de backup. Para os testes de falhas de registradores, escrita e leitura, do banco de registradores e do PC, serão implementadas lógicas extras que permitiram alterar algum bit (ou bits) dos dados processados quando dado um comando gerado pelo testador. Já a falha de memória de dados, será também usado uma lógica que permita inverter o valor alterado tanto na escrita quanto na leitura. A Figura 7 ilustra as localizações lógicas de injeção de falhas.

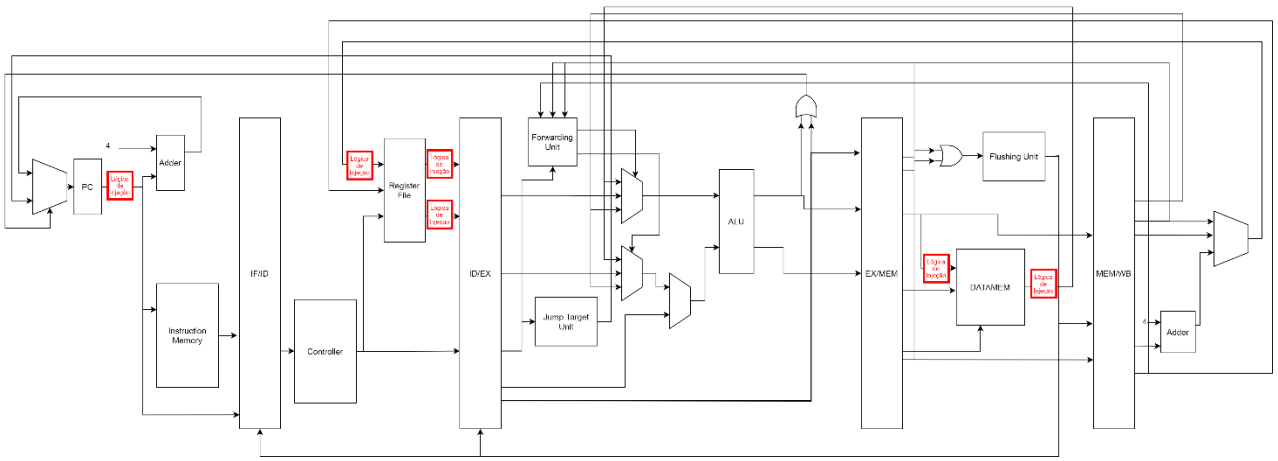

Figura 8. Ilustração da localização das lógicas para injeção de falhas.

Referências

CHRISÓTOMO, João Vitor Rafael. Maestro. Disponível em: https://github.com/Artoriuz/maestro. Acesso em: agosto de 2021.

PATTERSON, D. A.; HENNESSY, J. L.. Computer Organization and Design RISC-V Edition: The Hardware Software Interface. Morgan Kaufmann, 2017.

SIM, Mong Tee; ZHUANG, Yanyan. A Dual Lockstep Processor System-on-a-Chip for Fast Error Recovery in Safety-Critical Applications. In: IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society. IEEE, 2020. p. 2231-2238.