



## SISTEMAS DIGITAIS

---

Enzo Lisboa Peixoto e Pedro Scholz Soares

Dezembro de 2025

- Implementar um algoritmo de multiplicação de duas matrizes 3x3.
- HLS e duas opções de otimização
- Fazer o projeto PC-PO
- Comparar as 3 implementações em dados de área e desempenho

```
#include <stdint.h>

#define N 3

void matrix_mult_3x3(uint8_t A[N][N],
                    uint8_t B[N][N],
                    uint16_t R[N][N]) {

    uint16_t soma;

    linha_loop: for (int i = 0; i < N; i++) {
        // Loop das colunas de B
        coluna_loop: for (int j = 0; j < N; j++) {
            soma = 0;

            produto_loop: for (int k = 0; k < N; k++) {
                soma += A[i][k] * B[k][j];
            }

            R[i][j] = soma;
        }
    }
}
```

- Referência - sequencial
- Pipeline - foco em vazão
- Paralelismo Total - foco em Latência

- Adição de pipeline
- O Initiation Interval foi deixado em automático
- Objetivo: Melhorar a vazão, respeitando os limites da memória

- ARRAYPARTITION nas variáveis e UNROLL nos loops
- Converte a memória RAM em registradores distribuídos (FF), permitindo acesso simultâneo. O hardware de multiplicação é duplicado para realizar o cálculo de uma só vez
- Objetivo: Obter a latência mínima possível, aceitando o aumento considerável na área do chip

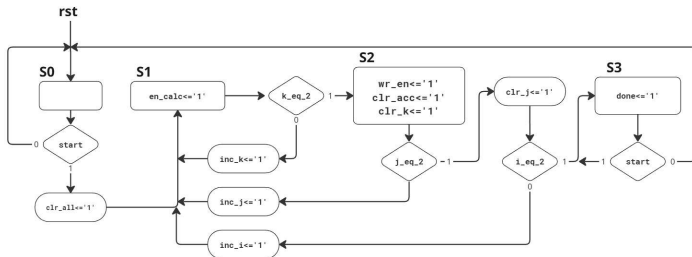
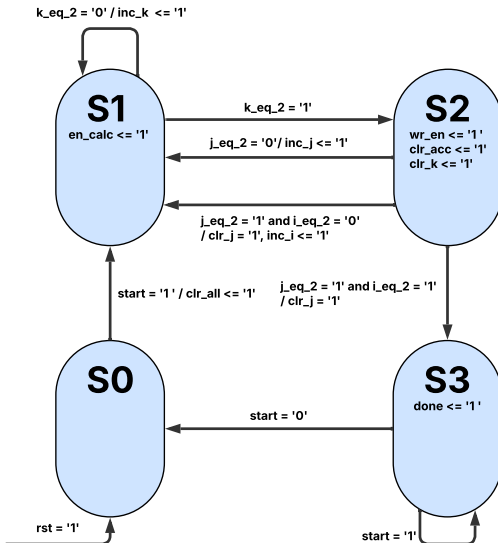


Figura: Fluxograma ASM PCPO





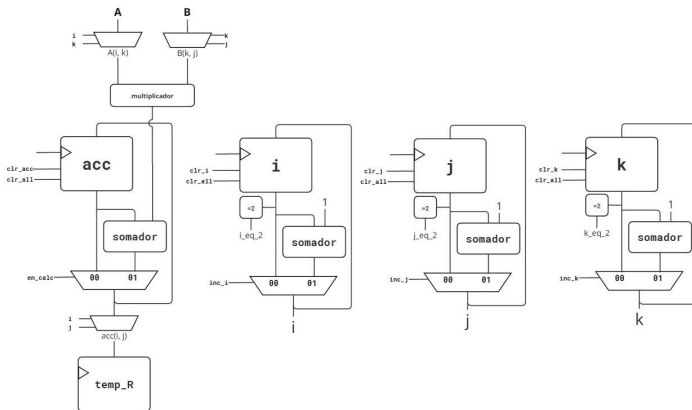


Figura: Parte Operativa

```
process(clk)
begin
    if rising_edge(clk) then
        if clr_all = '1' then
            i <= 0; j <= 0; k <= 0; acc <= (others => '0');
        else
            if inc_k = '1' then k <= k + 1; end if;
            if clr_k = '1' then k <= 0;      end if;

            if inc_j = '1' then j <= j + 1; end if;
            if clr_j = '1' then j <= 0;      end if;

            if inc_i = '1' then i <= i + 1; end if;
        end if;

        if clr_acc = '1' then
            acc <= (others => '0');
        elsif en_calc = '1' then
            acc <= acc + (matrizA(i, k) * matrizB(k, j));
        end if;

        if wr_en = '1' then
            matriz_R_interna(i, j) <= acc;
        end if;
    end if;
end process;
k_eq_2 <= '1' when k = 2 else '0';
j_eq_2 <= '1' when j = 2 else '0';
i_eq_2 <= '1' when i = 2 else '0';
resul <= matriz_R_interna;
```

```
process(clk, rst)
begin
    if rst = '1' then
        estado_atual <= s0;
    elsif rising_edge(clk) then
        case estado_atual is
            when s0 =>
                if start = '1' then
                    estado_atual <= s1;
                end if;

            when s1 =>
                if k_eq_2 = '1' then
                    estado_atual <= s2;
                else
                    estado_atual <= s1;
                end if;

            when s2 =>
                if j_eq_2 = '0' then
                    estado_atual <= s1;
                else
                    if i_eq_2 = '0' then
                        estado_atual <= s1;
                    else
                        estado_atual <= s3;
                    end if;
                end if;
            end if;
        end case;
    end if;
end process;
```

```
        when s3 =>
            if start = '0' then
                estado_atual <= s0;
            end if;
        end case;
    end if;
end process;

--COMBINACIONAL PC
process(estado_atual, k_eq_2, j_eq_2, i_eq_2, start)
begin
    clr_all <= '0'; en_calc <= '0'; inc_k <= '0'; clr_k <= '0';
    inc_j <= '0'; clr_j <= '0'; inc_i <= '0';
    wr_en <= '0'; clr_acc <= '0'; done <= '0';

    case estado_atual is
        when s0 =>
            if start = '1' then clr_all <= '1'; end if;

        when s1 =>
            en_calc <= '1';
            if k_eq_2 = '0' then inc_k <= '1'; end if;

        when s2 =>
            wr_en <= '1';
            clr_acc <= '1';
            clr_k <= '1';
```

```
        if j_eq_2 = '0' then
            inc_j <= '1';
        else
            clr_j <= '1';
            if i_eq_2 = '0' then
                inc_i <= '1';
            end if;
        end if;

        when s3 =>
            done <= '1';
        end case;
    end process;
```

**Enzo Lisbôa Peixoto e Pedro Scholz Soares**

Instituto de Informática — UFRGS

`elpeixoto@inf.ufrgs.br`

`pedro.soares@inf.ufrgs.br`

