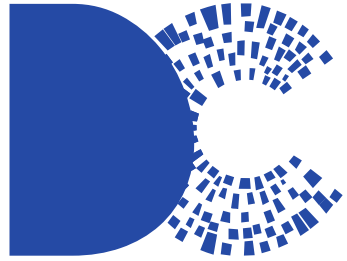
UNIVERSIDADE FEDERAL DE SERGIPE



DEPARTAMENTO DE COMPUTAÇÃO

Fluxo de desenvolvimento

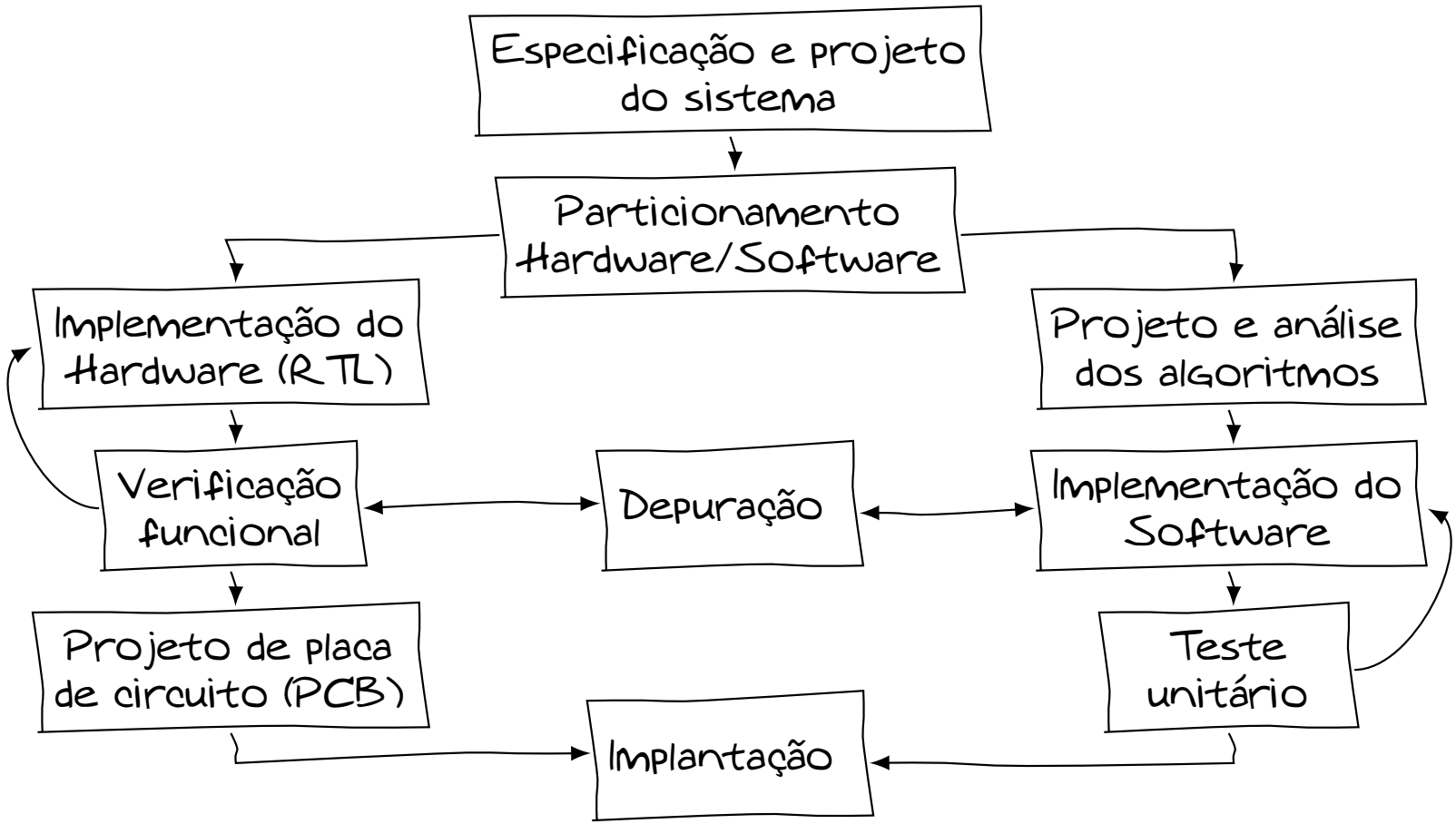
Sistemas Embarcados

Bruno Prado

Departamento de Computação / UFS

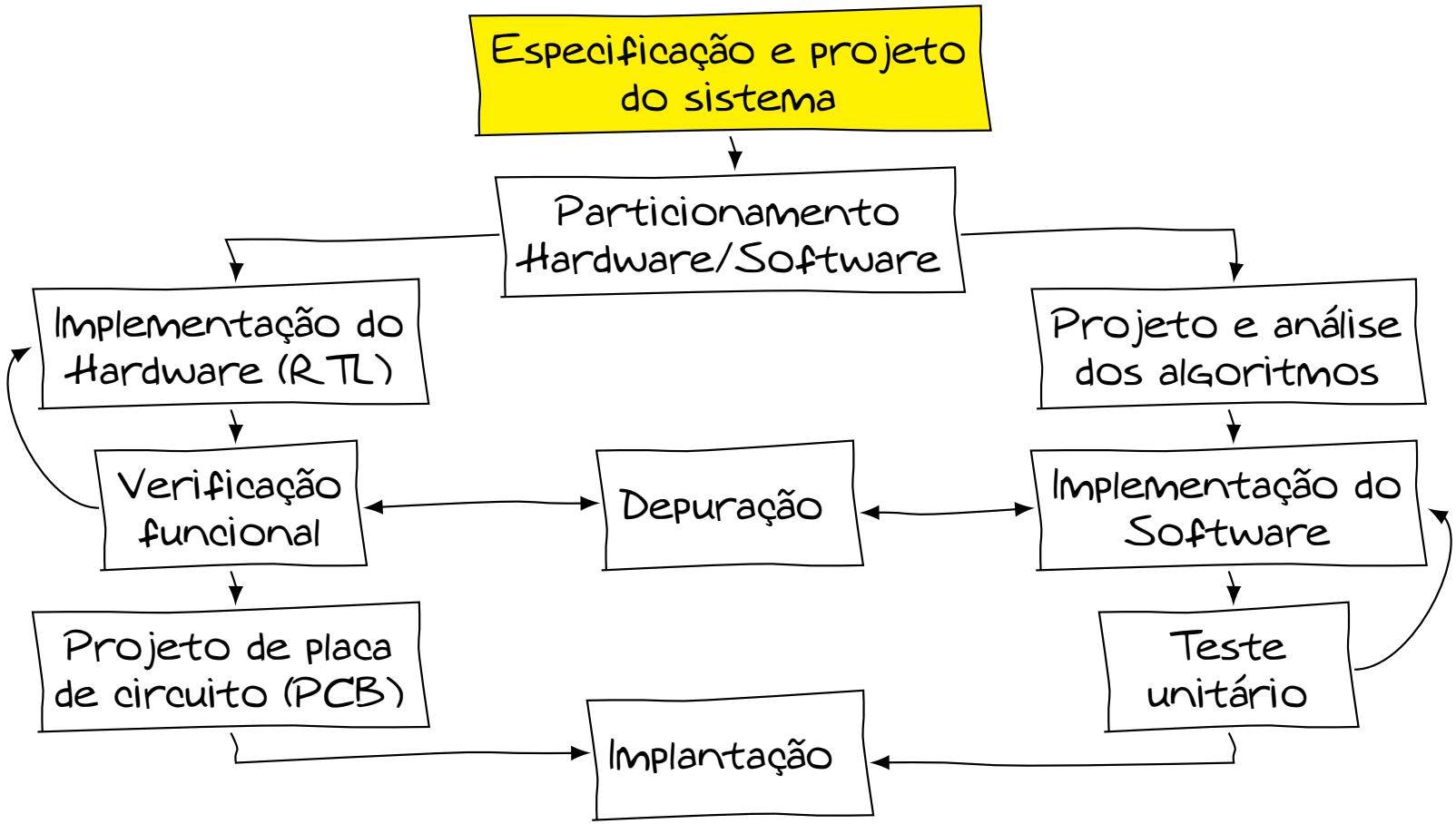
Introdução

- Fluxo de desenvolvimento



Introdução

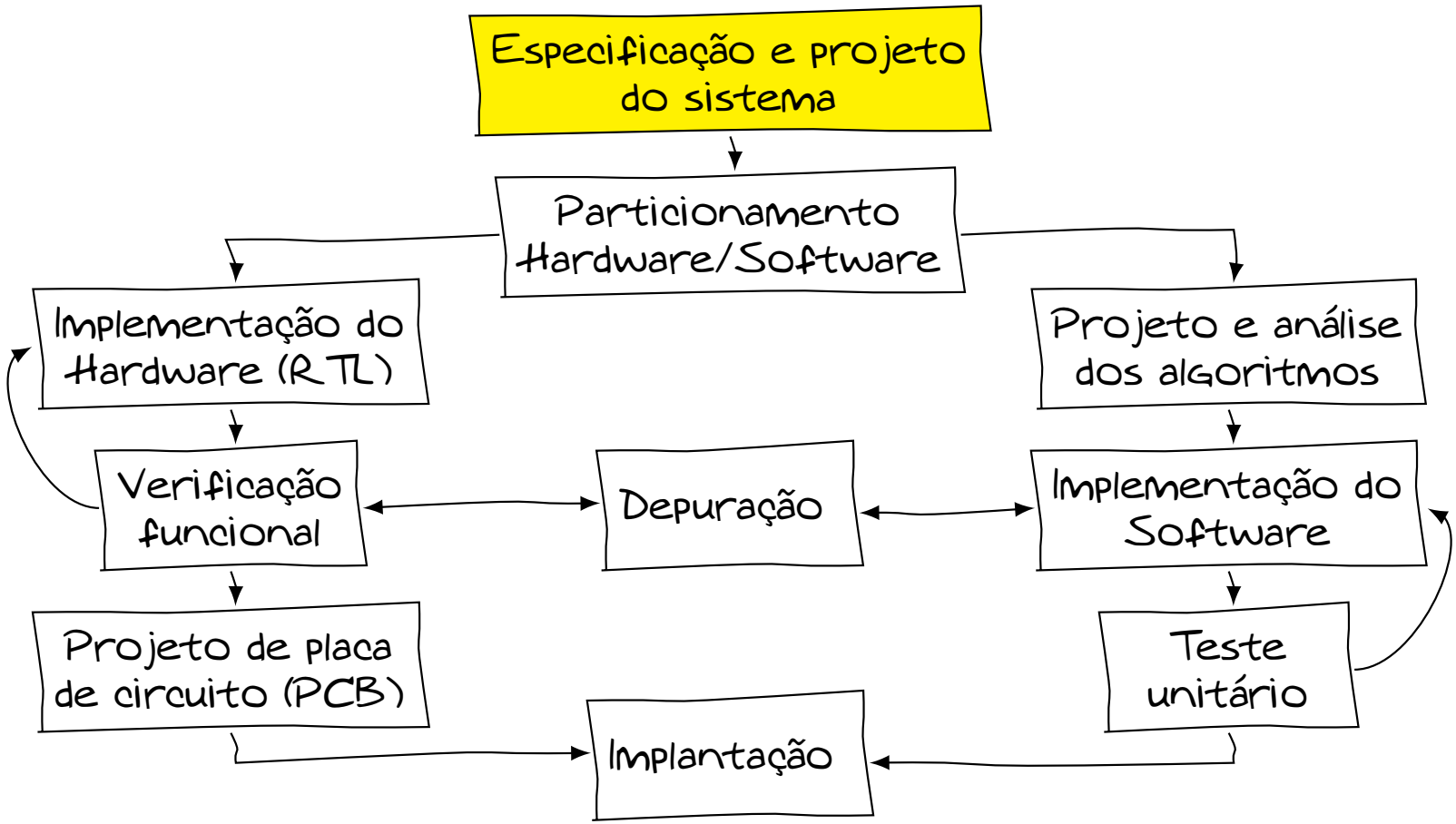
- Fluxo de desenvolvimento



Transformação do conceito em produto

Introdução

- Fluxo de desenvolvimento



Elicitação de requisitos desejáveis e obrigatórios (funcionais) + restrições (não funcionais)

Introdução

- Fluxo de desenvolvimento

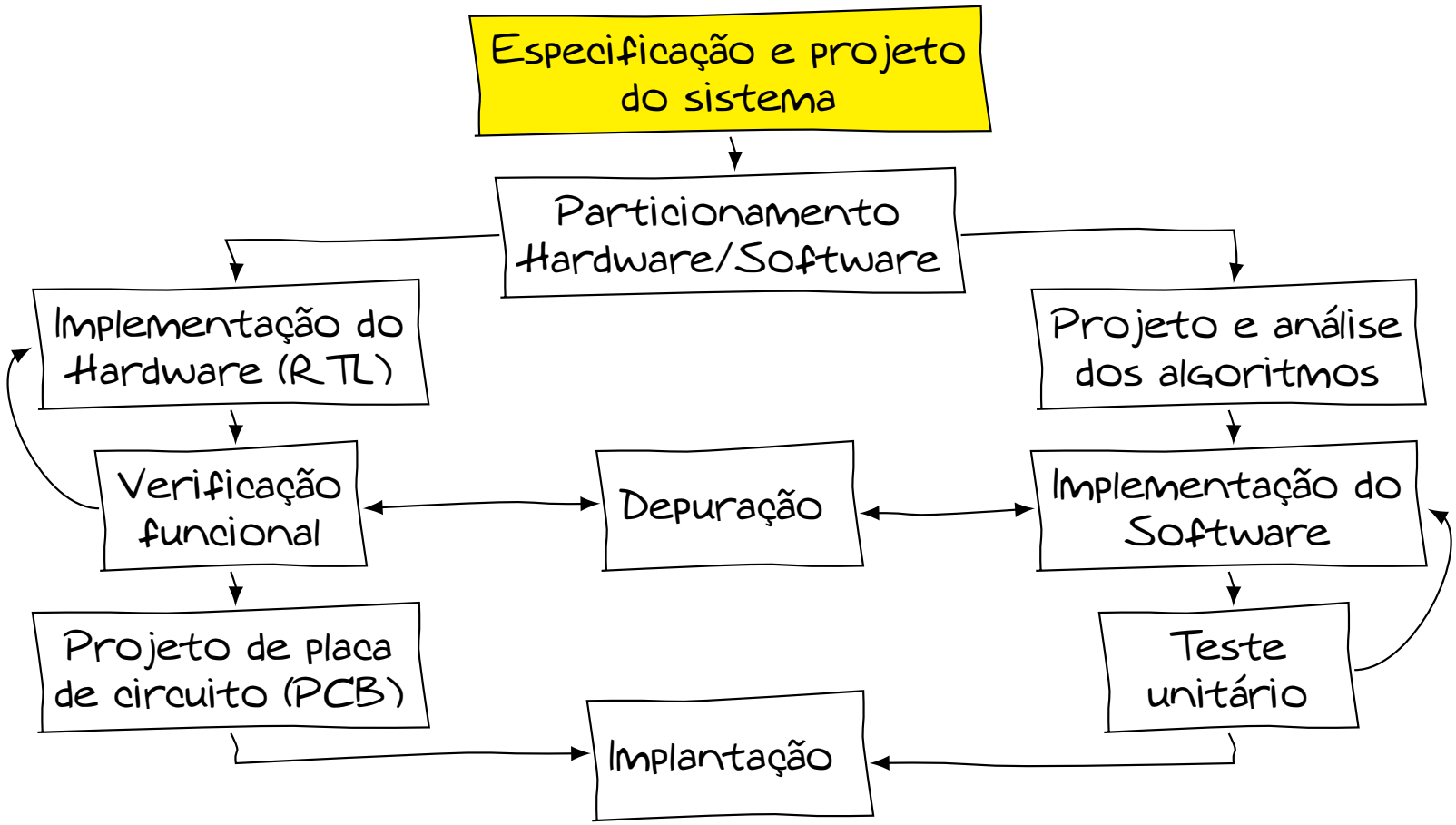
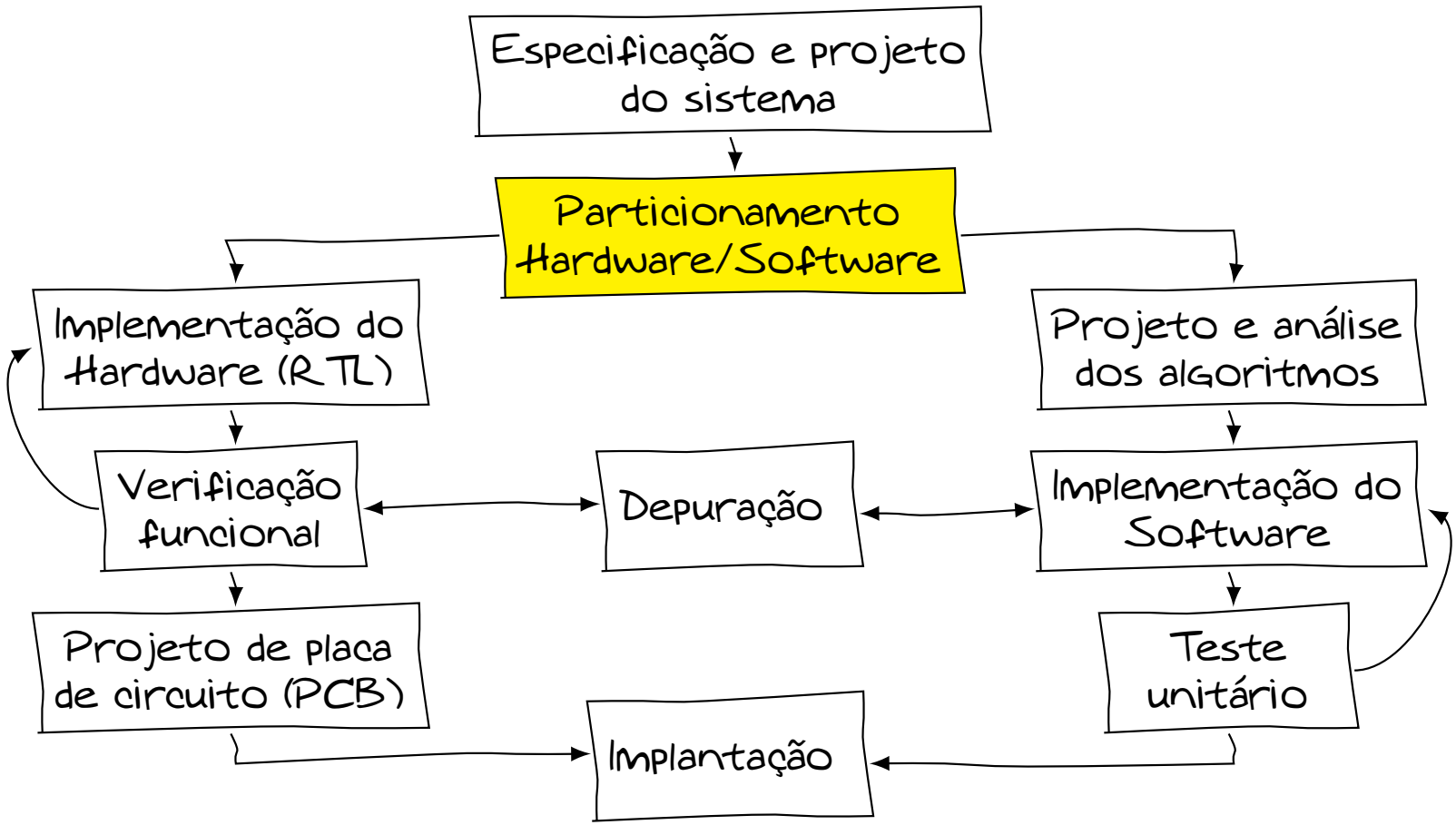


Diagrama de blocos (componentes)
e arquitetura do sistema (interfaces)

Introdução

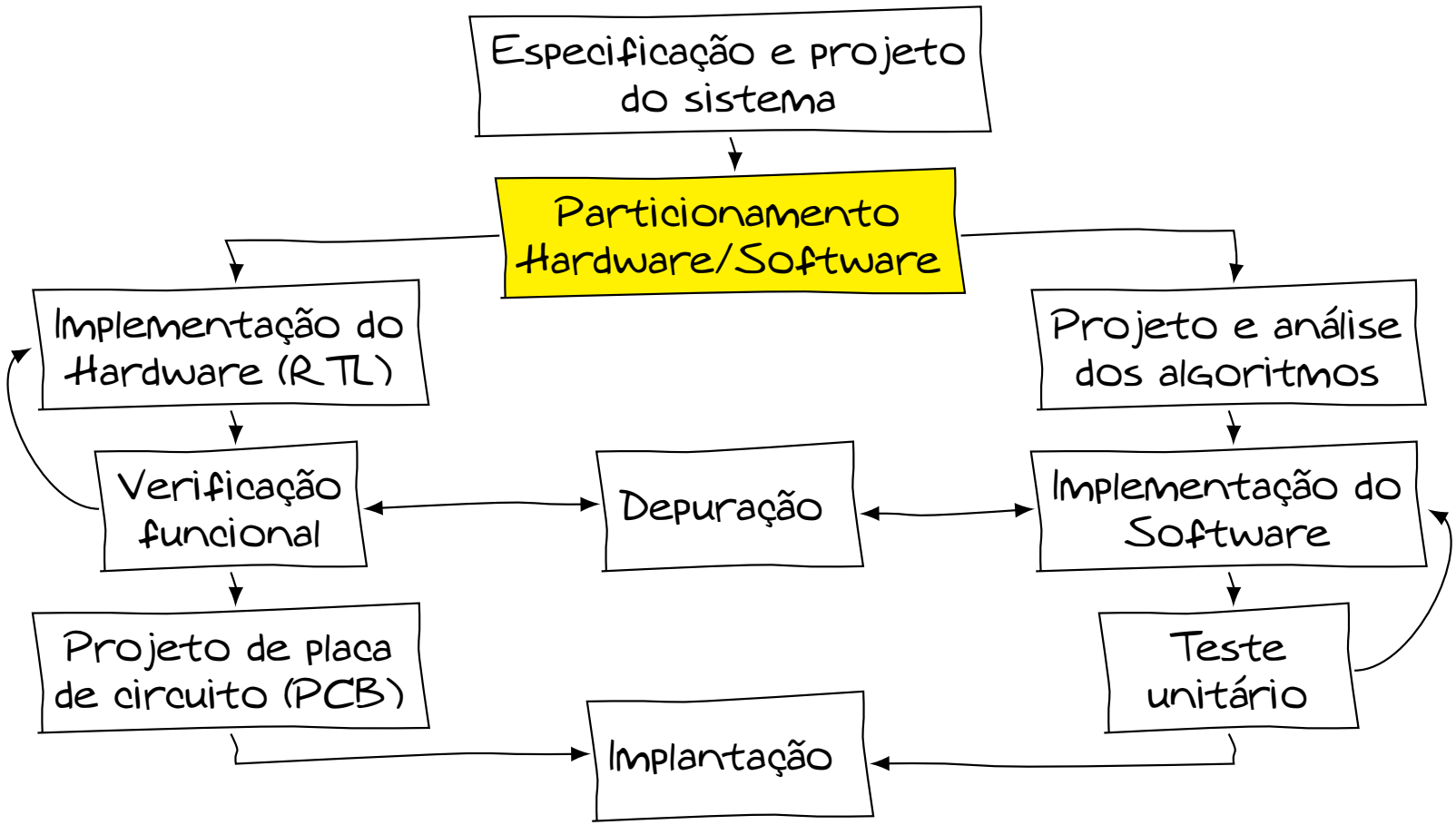
- Fluxo de desenvolvimento



O que será implementado em Hardware e em Software?

Introdução

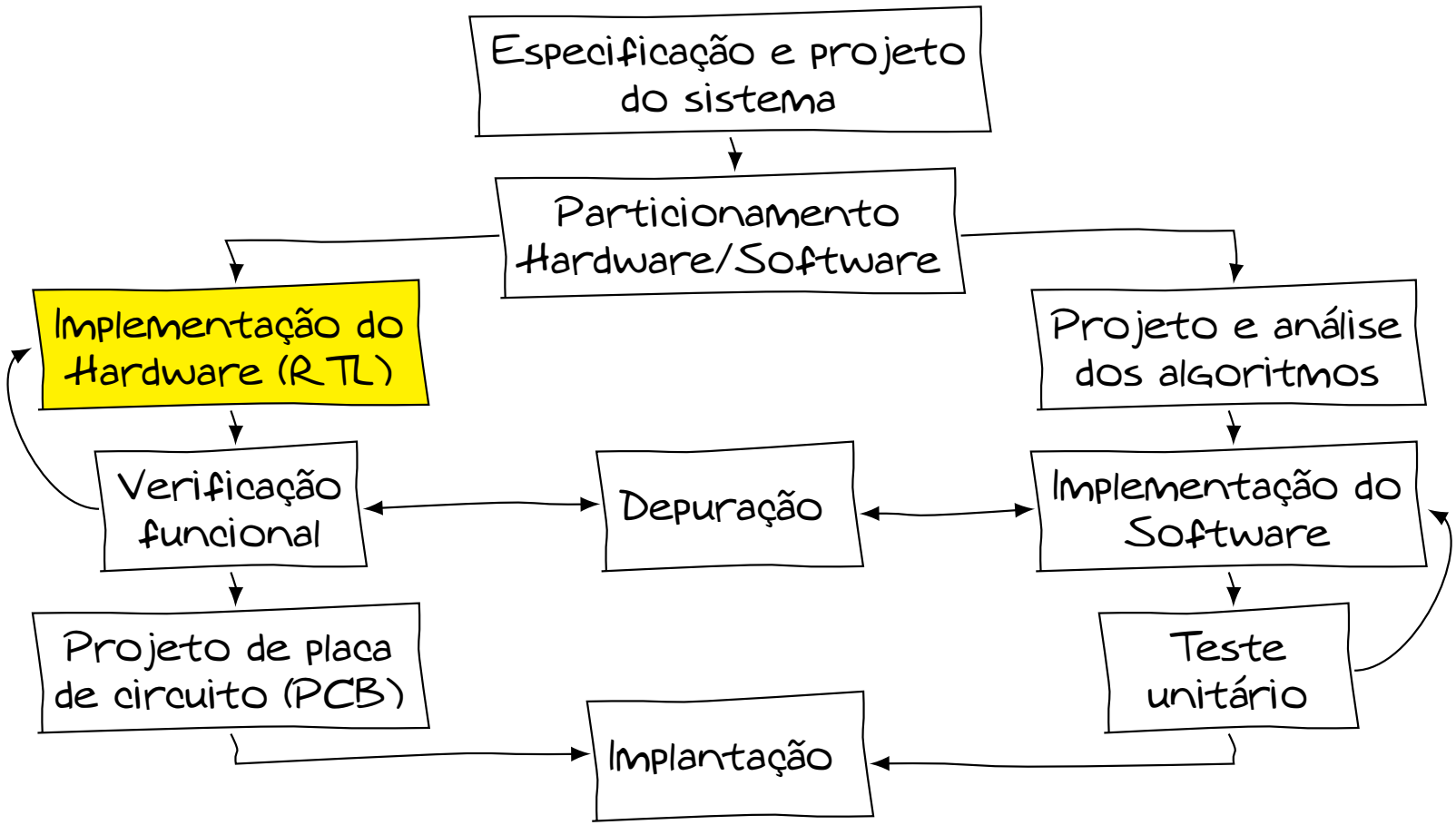
- Fluxo de desenvolvimento



Custo $\times$ Desempenho $\times$ Padronização

Introdução

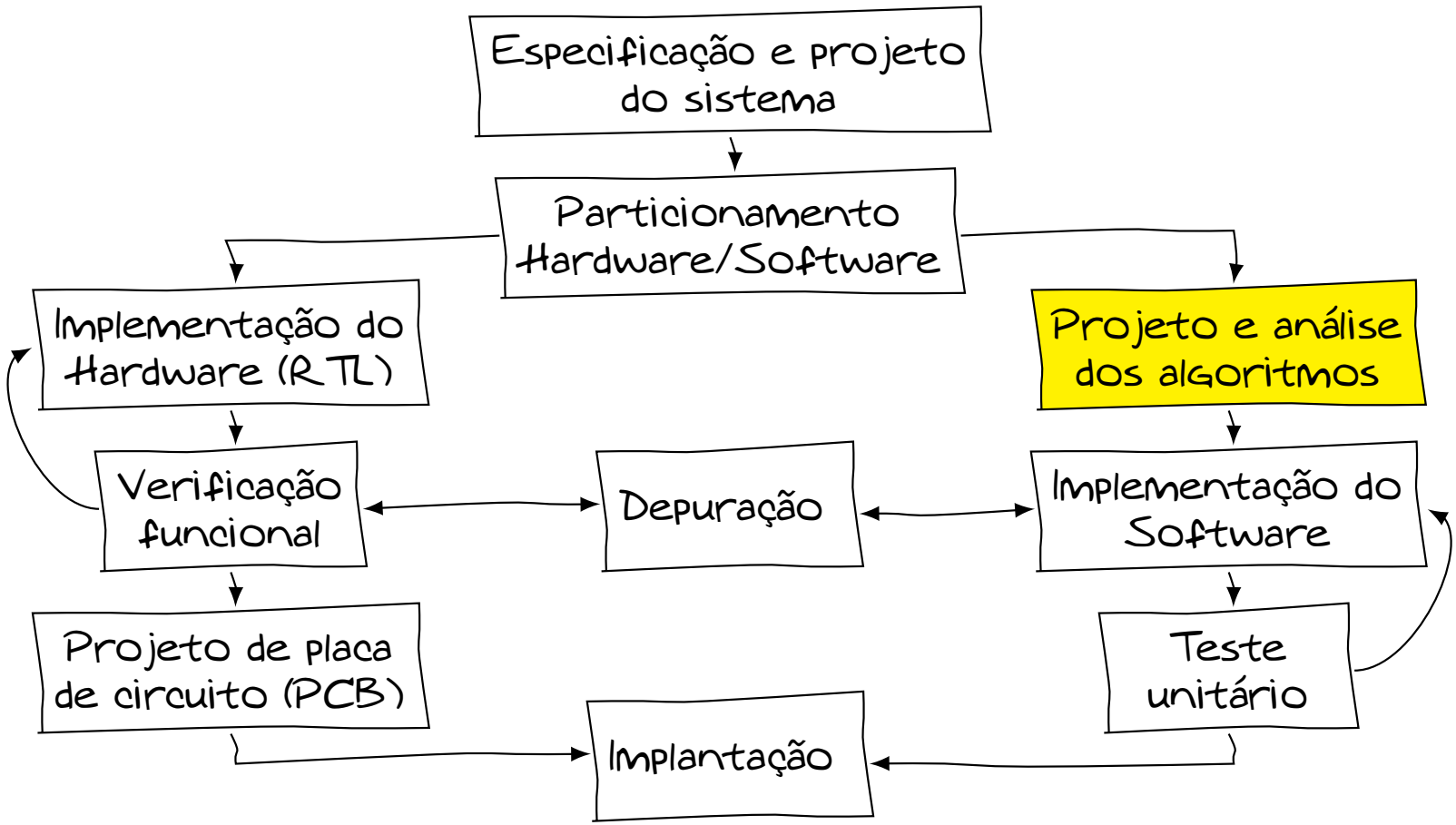
- Fluxo de desenvolvimento



Prototipação em ASIC ou FPGA

Introdução

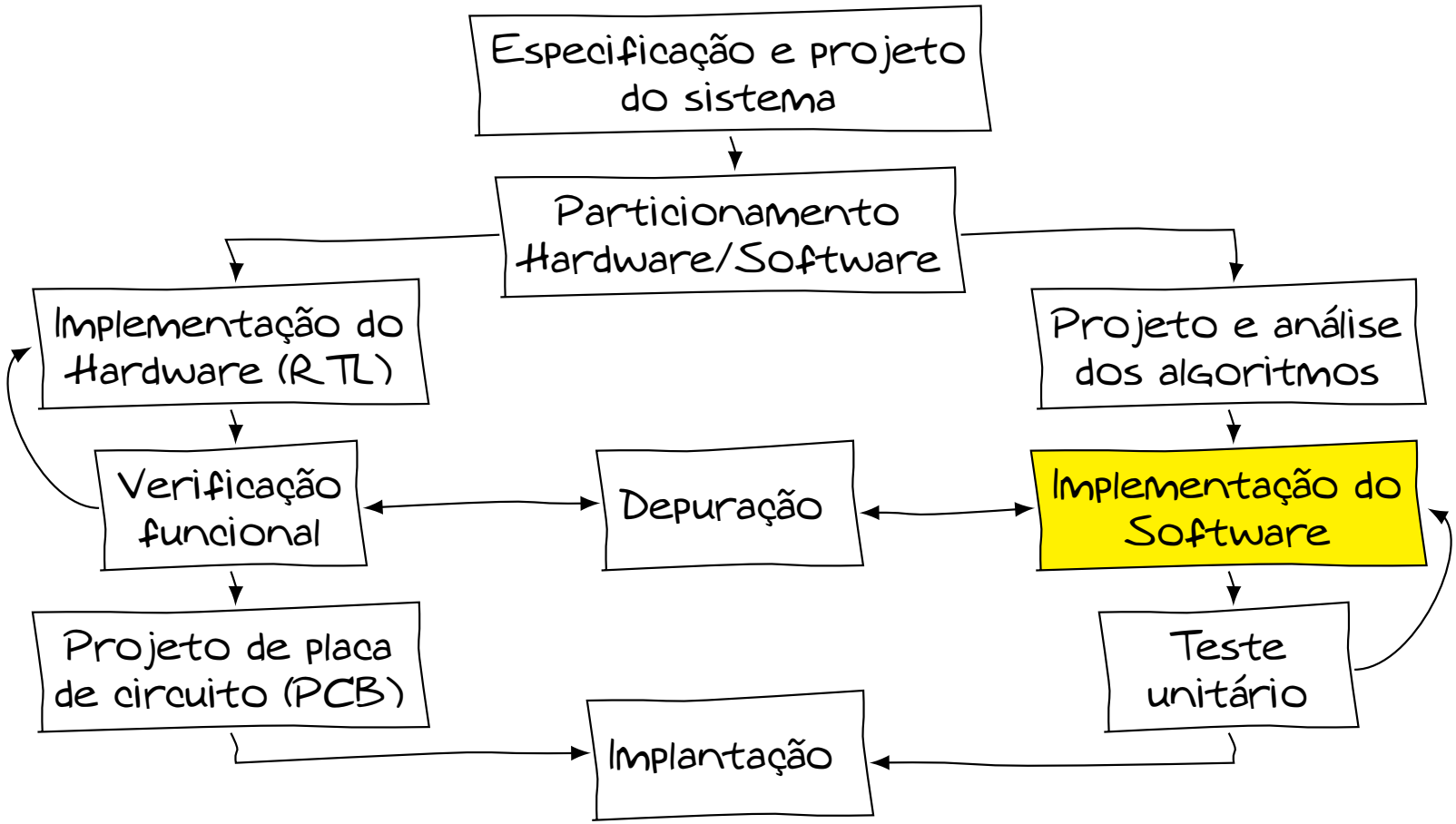
- Fluxo de desenvolvimento



Definição dos algoritmos e das estruturas de dados

Introdução

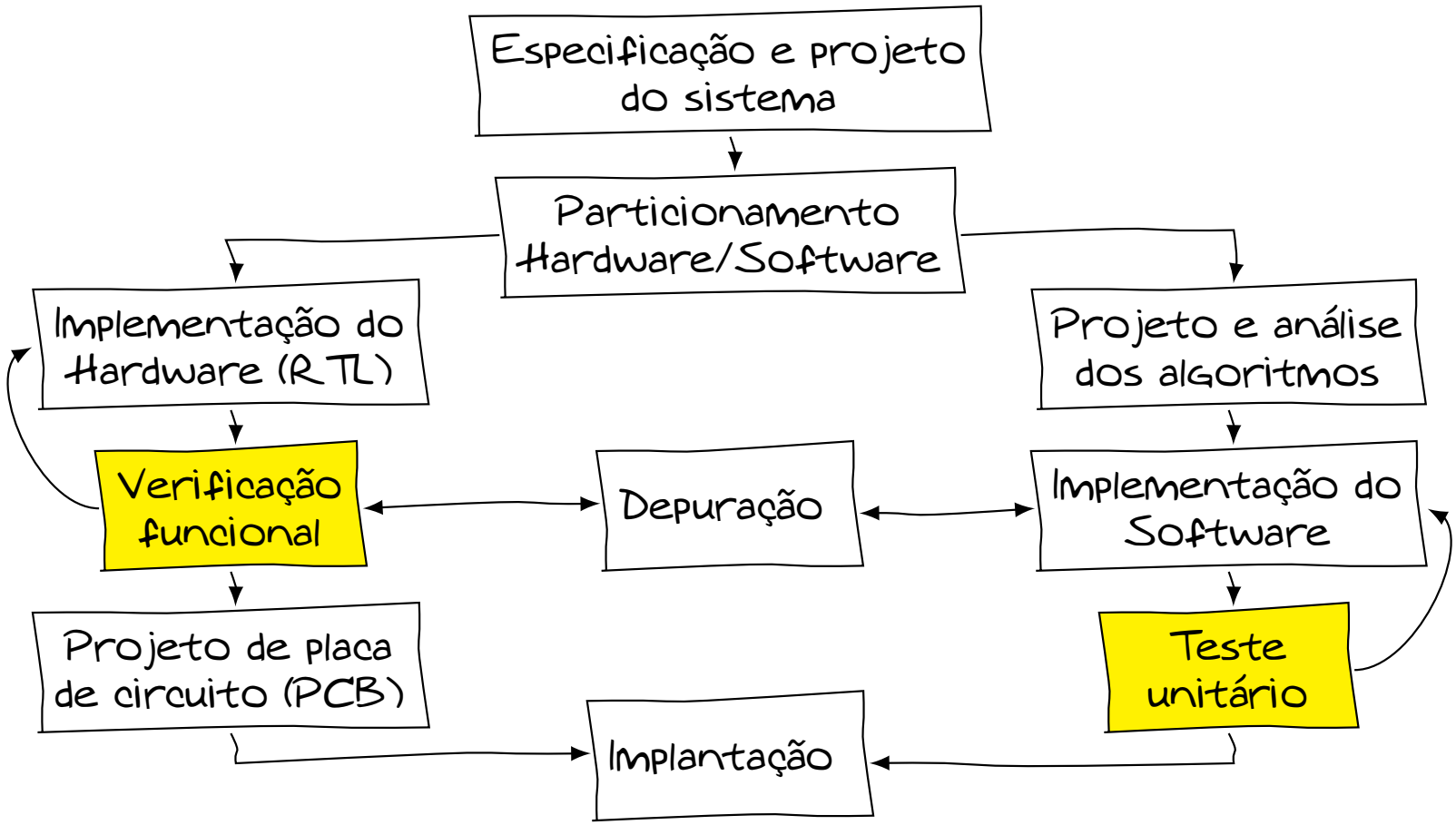
- Fluxo de desenvolvimento



Ambiente de desenvolvimento
(compilação cruzada e programação do dispositivo)

Introdução

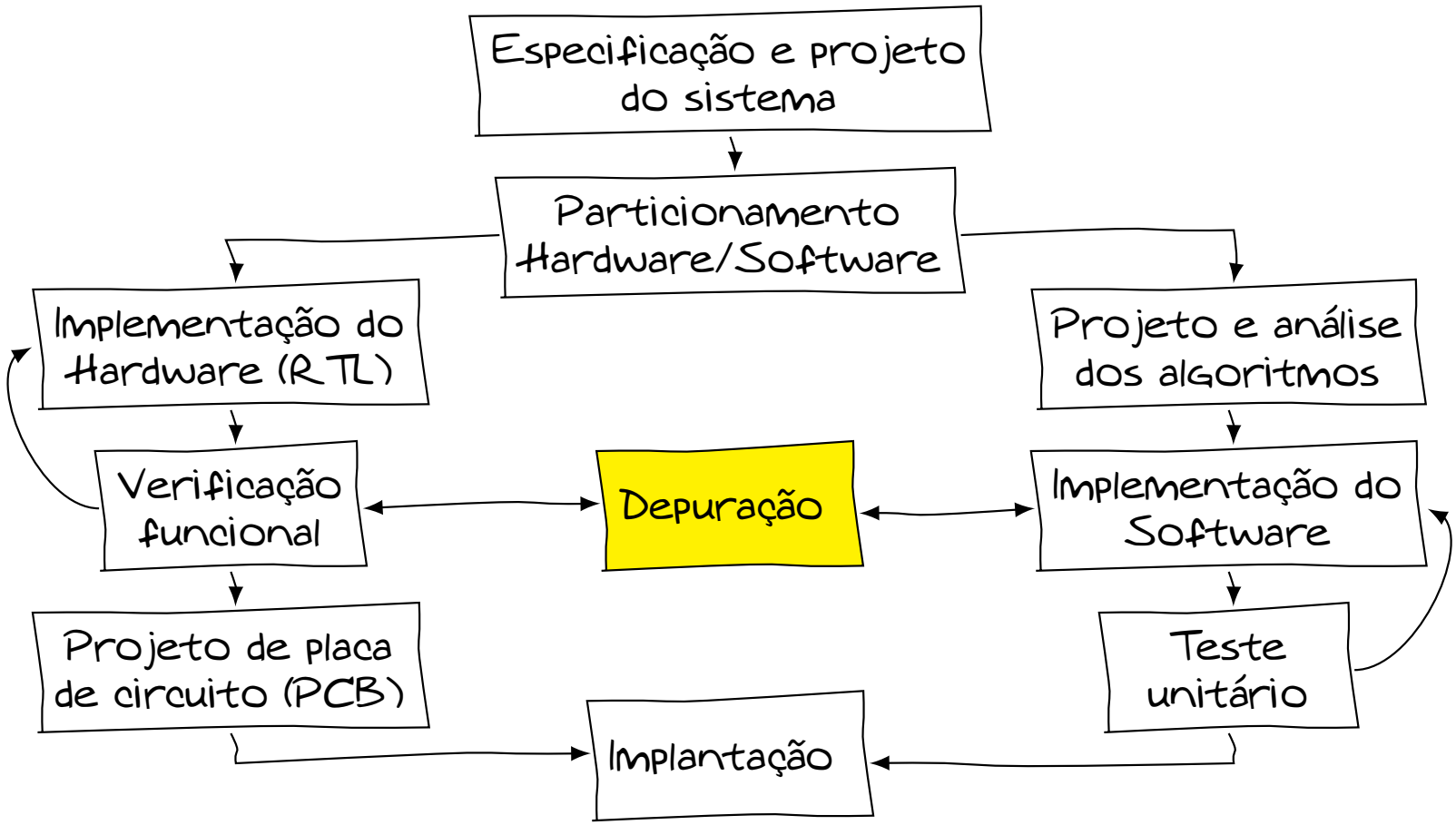
- Fluxo de desenvolvimento



Verificação de conformidade com a especificação (processo iterativo e incremental)

Introdução

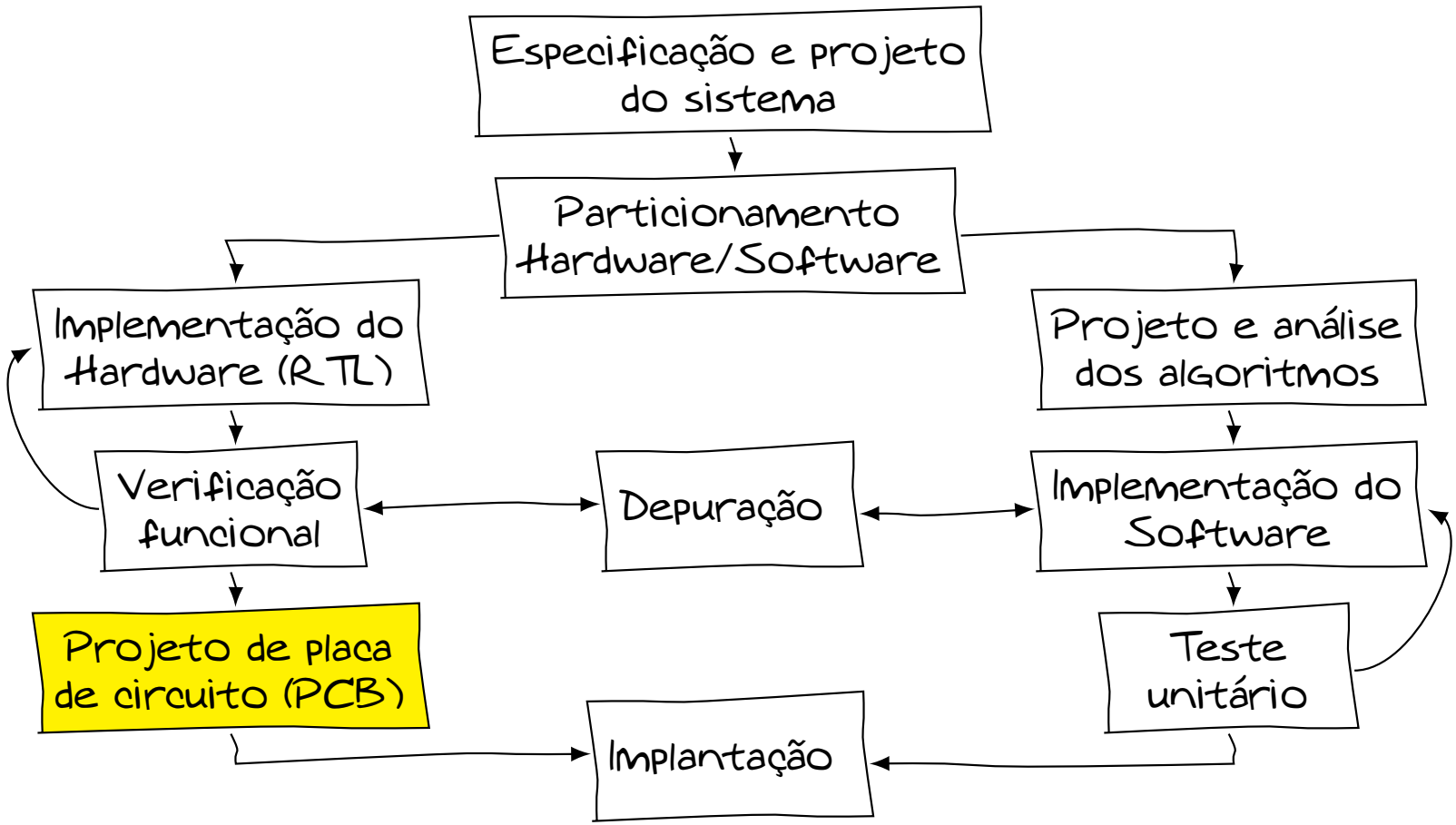
- Fluxo de desenvolvimento



Depuração por Hardware (JTAG + OCD)
e por Software (GDB + simulador)

Introdução

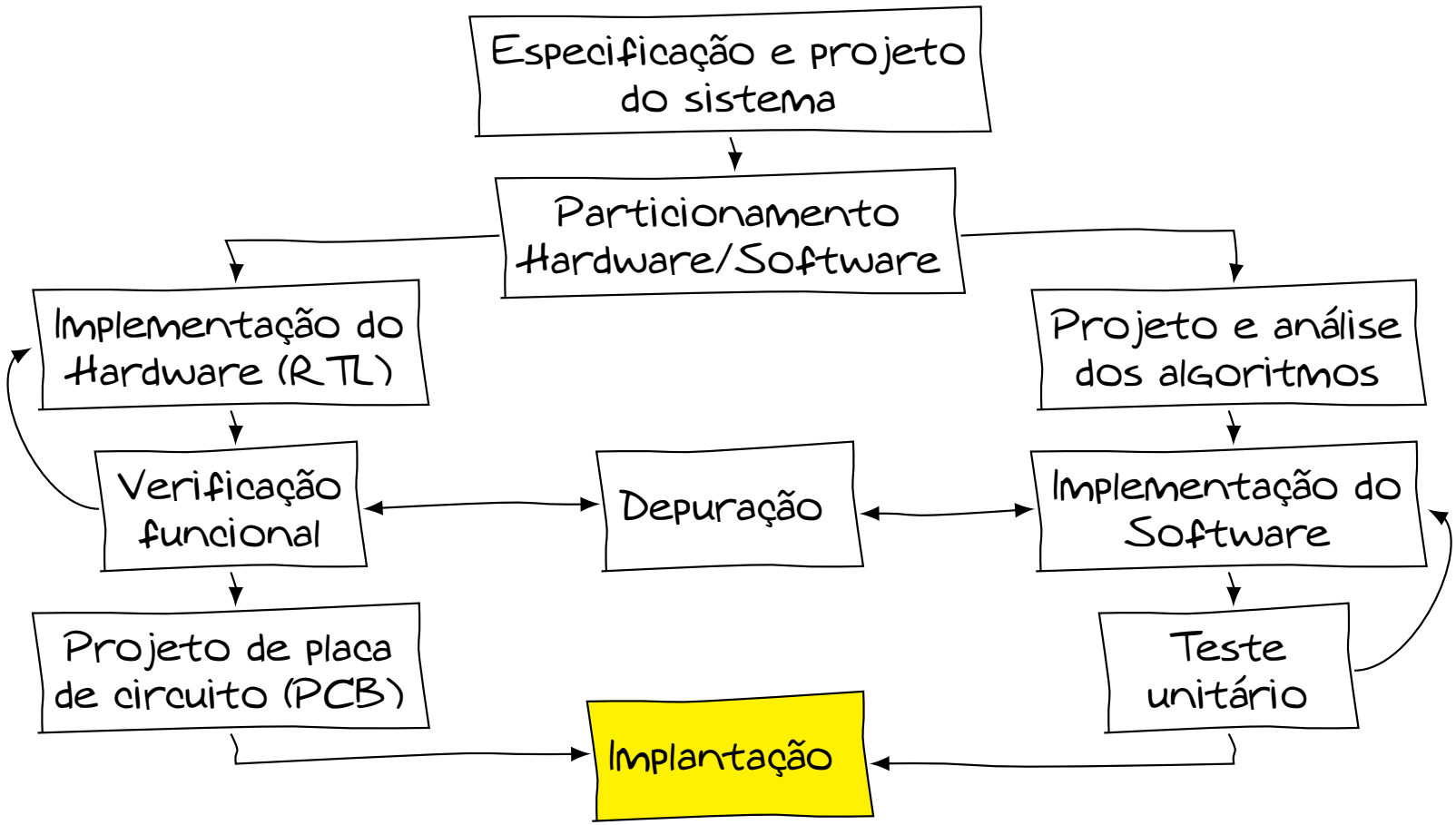
- Fluxo de desenvolvimento



Integração dos componentes de Hardware

Introdução

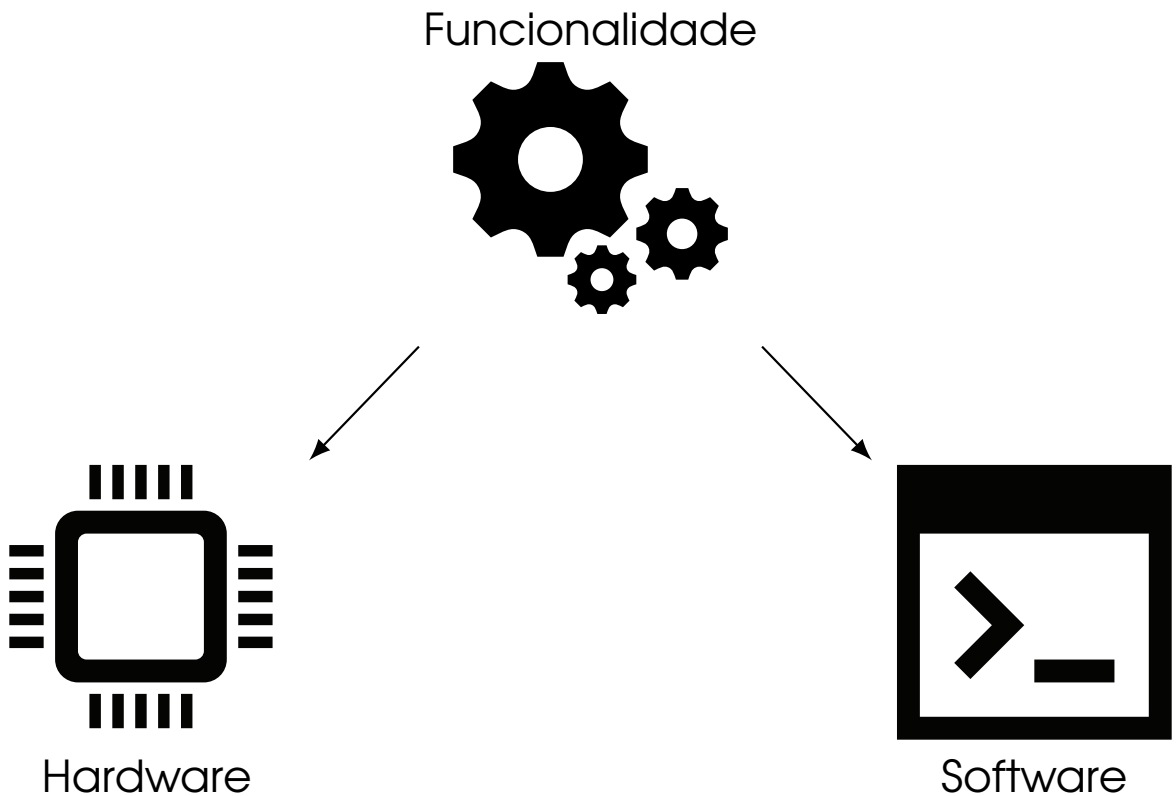
- Fluxo de desenvolvimento



Entrega do produto com atualizações e manutenção

Particionamento

- Hardware ou Software?
  - A equipe de desenvolvimento precisa decidir como as funcionalidades do sistema embarcado serão implementadas (particionamento)



Particionamento

- Hardware ou Software?
  - Operações de ponto flutuante (IEEE 754)

Particionamento

- Hardware ou Software?
  - Operações de ponto flutuante (IEEE 754)
    - A maioria absoluta dos sistemas embarcados não tem suporte em hardware para realizar estas operações por serem complexas e demoradas

Particionamento

- Hardware ou Software?
  - Operações de ponto flutuante (IEEE 754)
    - A maioria absoluta dos sistemas embarcados não tem suporte em hardware para realizar estas operações por serem complexas e demoradas
    - Na aritmética de ponto flutuante, o programador não precisa se preocupar com questões de arredondamento e de precisão das operações

Particionamento

- Hardware ou Software?
  - Operações de ponto flutuante (IEEE 754)
    - A maioria absoluta dos sistemas embarcados não tem suporte em hardware para realizar estas operações por serem complexas e demoradas
    - Na aritmética de ponto flutuante, o programador não precisa se preocupar com questões de arredondamento e de precisão das operações
    - Pode ser implementado por um coprocessador aritmético dedicado (hardware) ou por uma biblioteca de emulação (software)

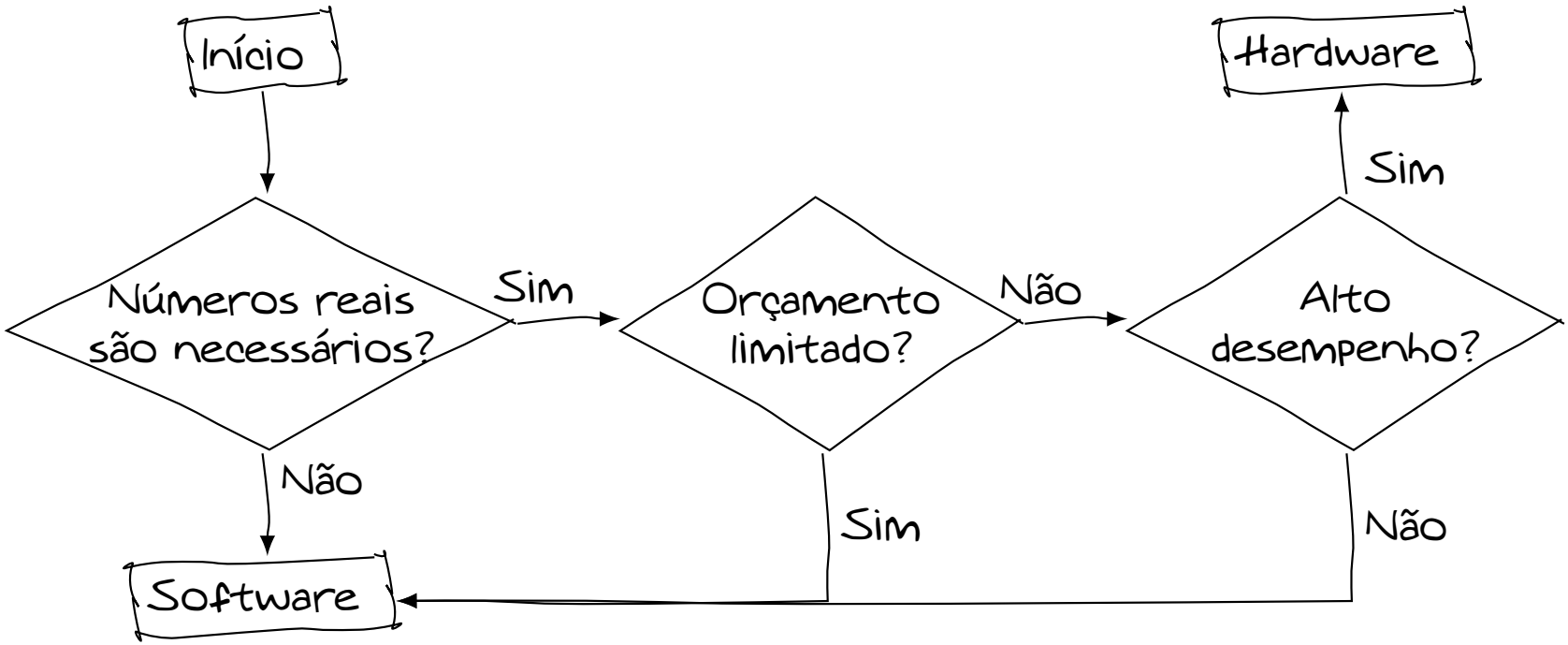
Particionamento

- Hardware ou Software?
  - Operações de ponto flutuante (IEEE 754)
    - A maioria absoluta dos sistemas embarcados não tem suporte em hardware para realizar estas operações por serem complexas e demoradas
    - Na aritmética de ponto flutuante, o programador não precisa se preocupar com questões de arredondamento e de precisão das operações
    - Pode ser implementado por um coprocessador aritmético dedicado (hardware) ou por uma biblioteca de emulação (software)

Como tomar esta decisão de projeto?

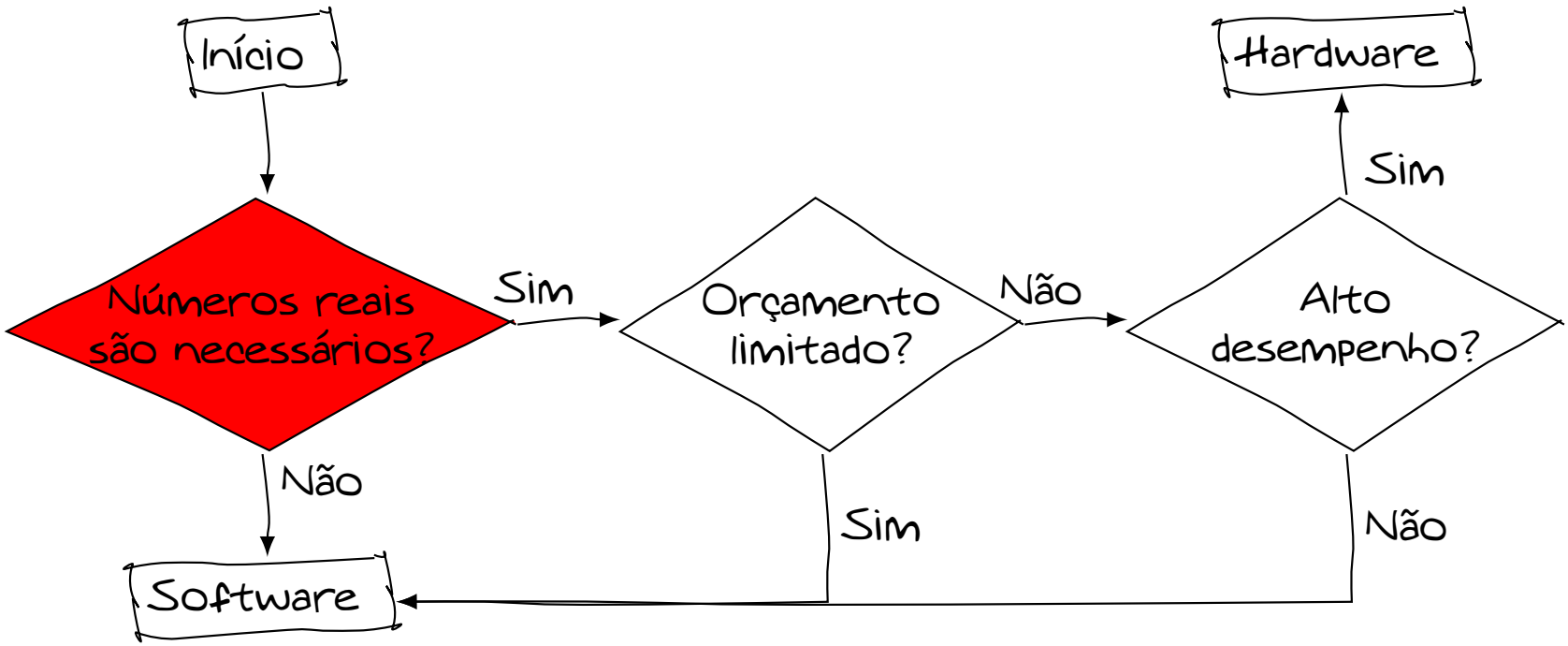
Particionamento

- Hardware ou Software?
  - Operações de ponto flutuante (IEEE 754)



Particionamento

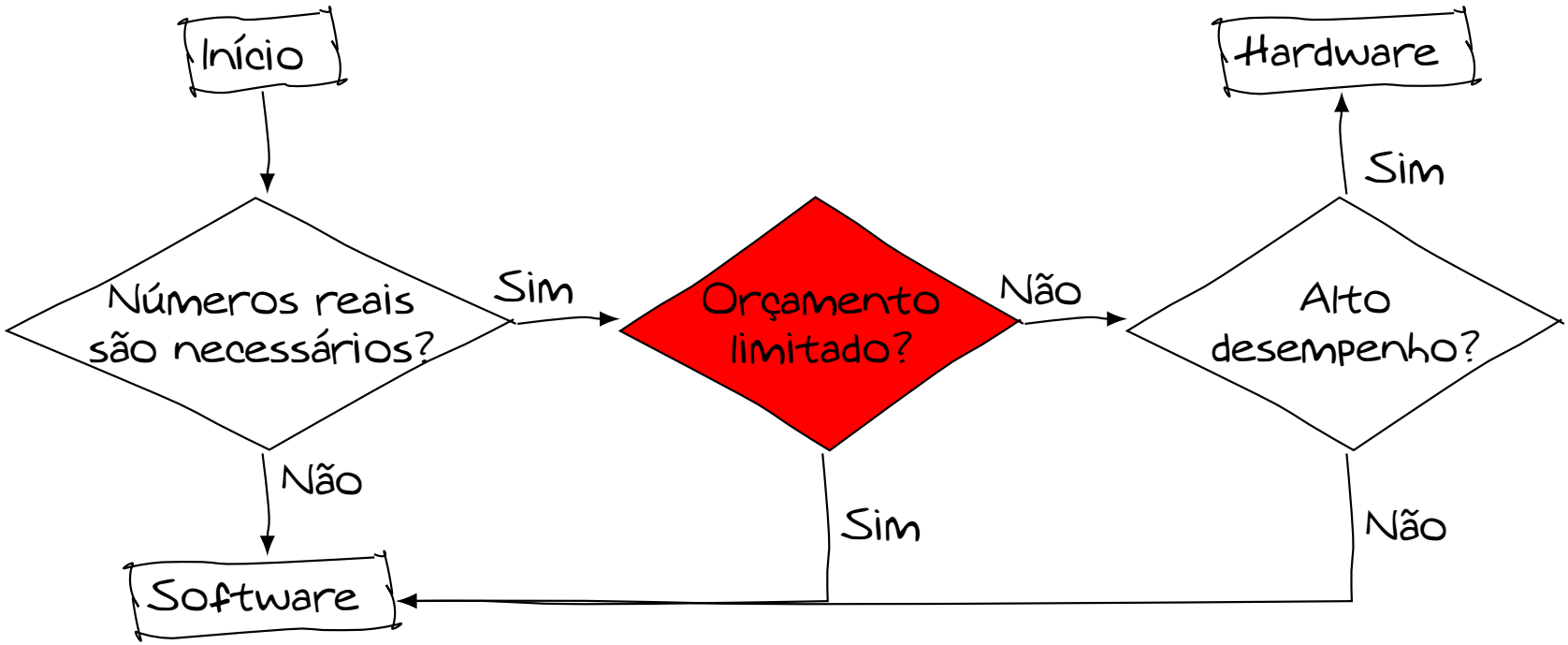
- Hardware ou Software?
  - Operações de ponto flutuante (IEEE 754)



Verificar alternativas
para implementação de números reais

Particionamento

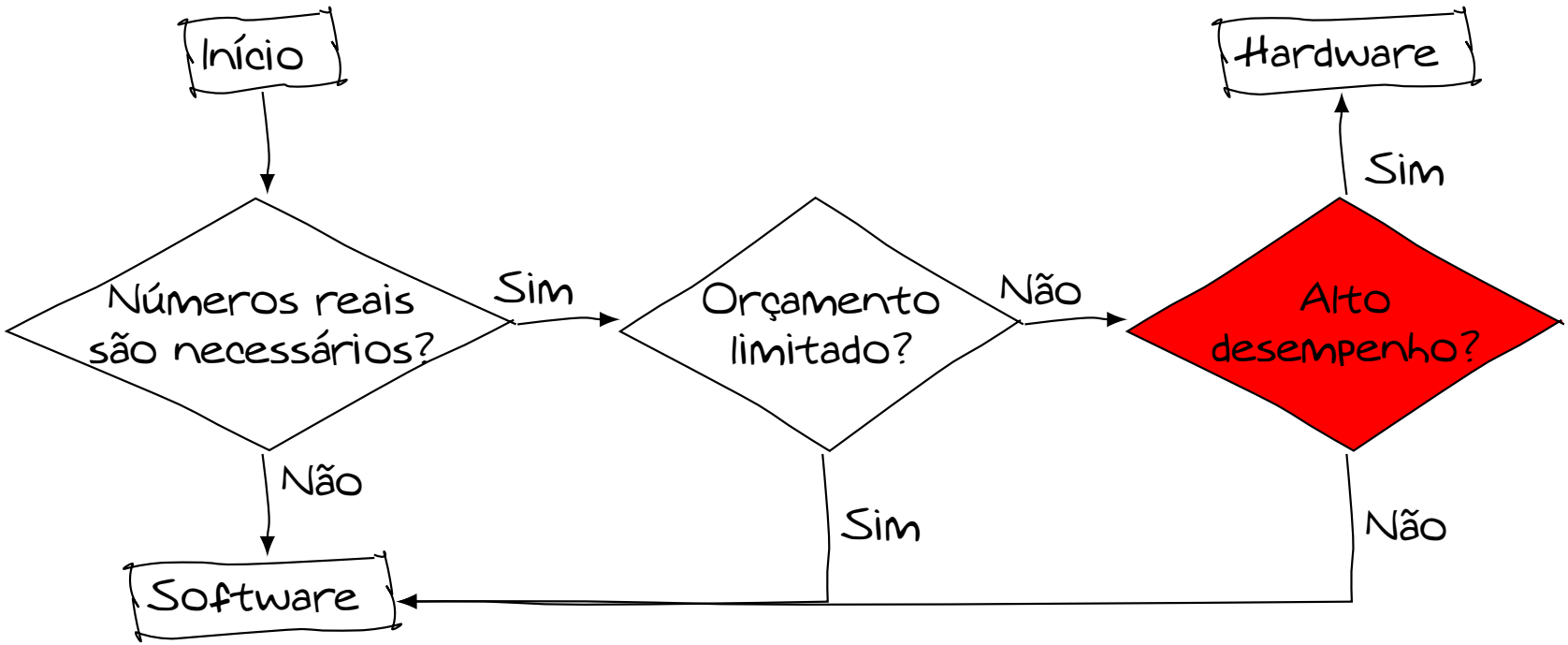
- Hardware ou Software?
  - Operações de ponto flutuante (IEEE 754)



Aumentar o tempo de desenvolvimento gera custos e impacta no cronograma

Particionamento

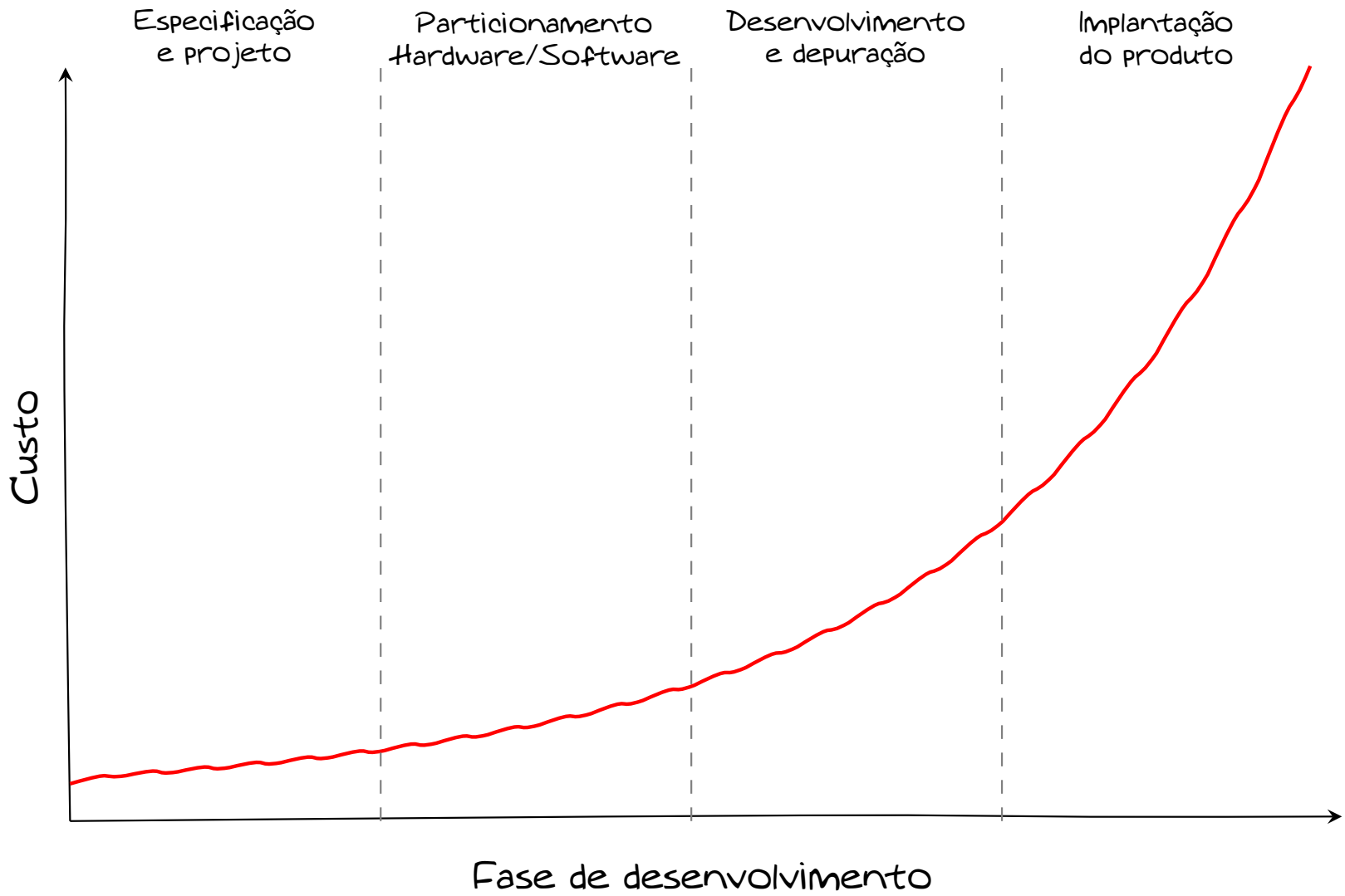
- Hardware ou Software?
  - Operações de ponto flutuante (IEEE 754)



Escolher entre componente personalizado ou de prateleira

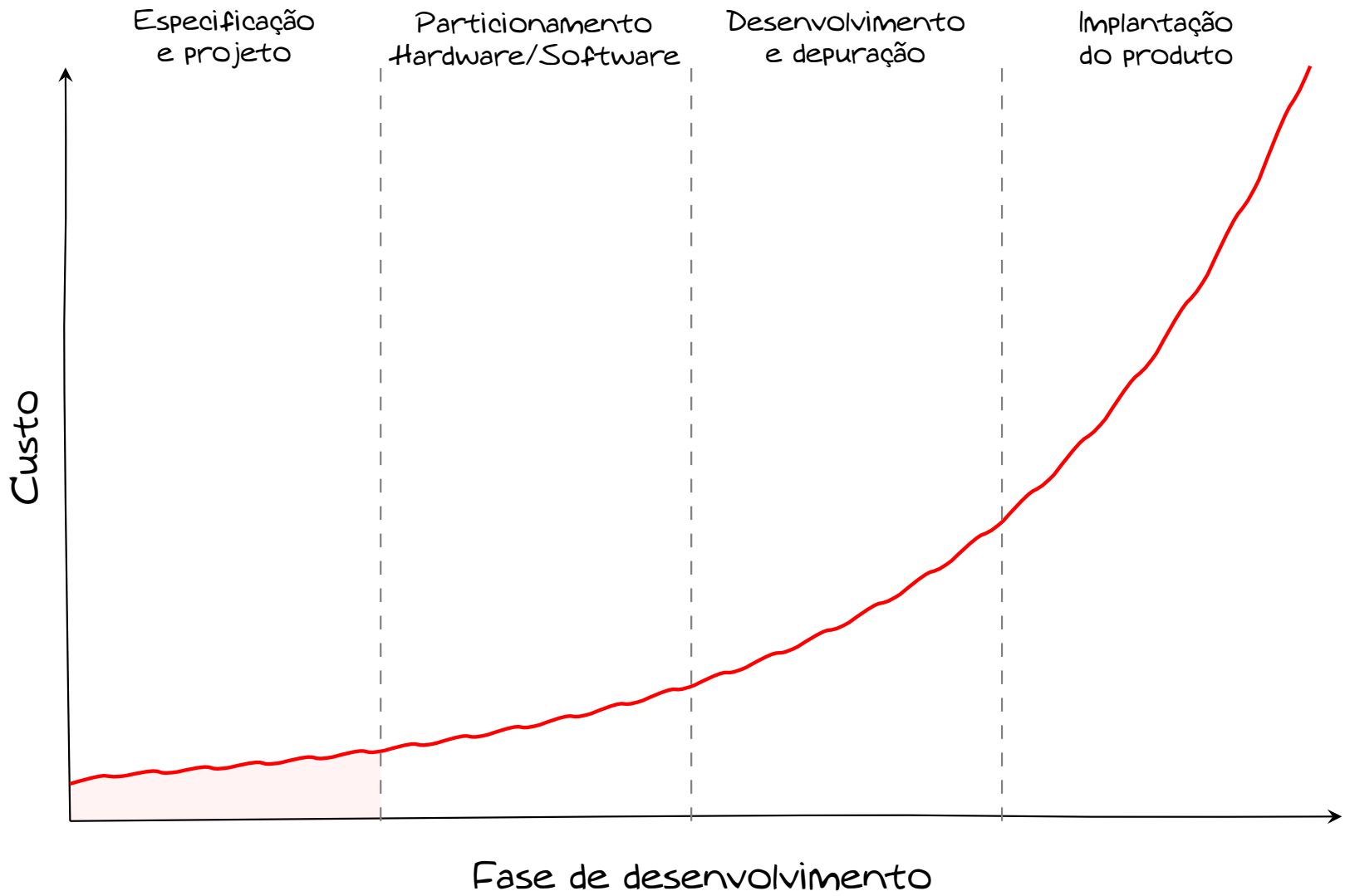
Particionamento

- Hardware ou Software?
  - Custo de correções e mudanças no desenvolvimento



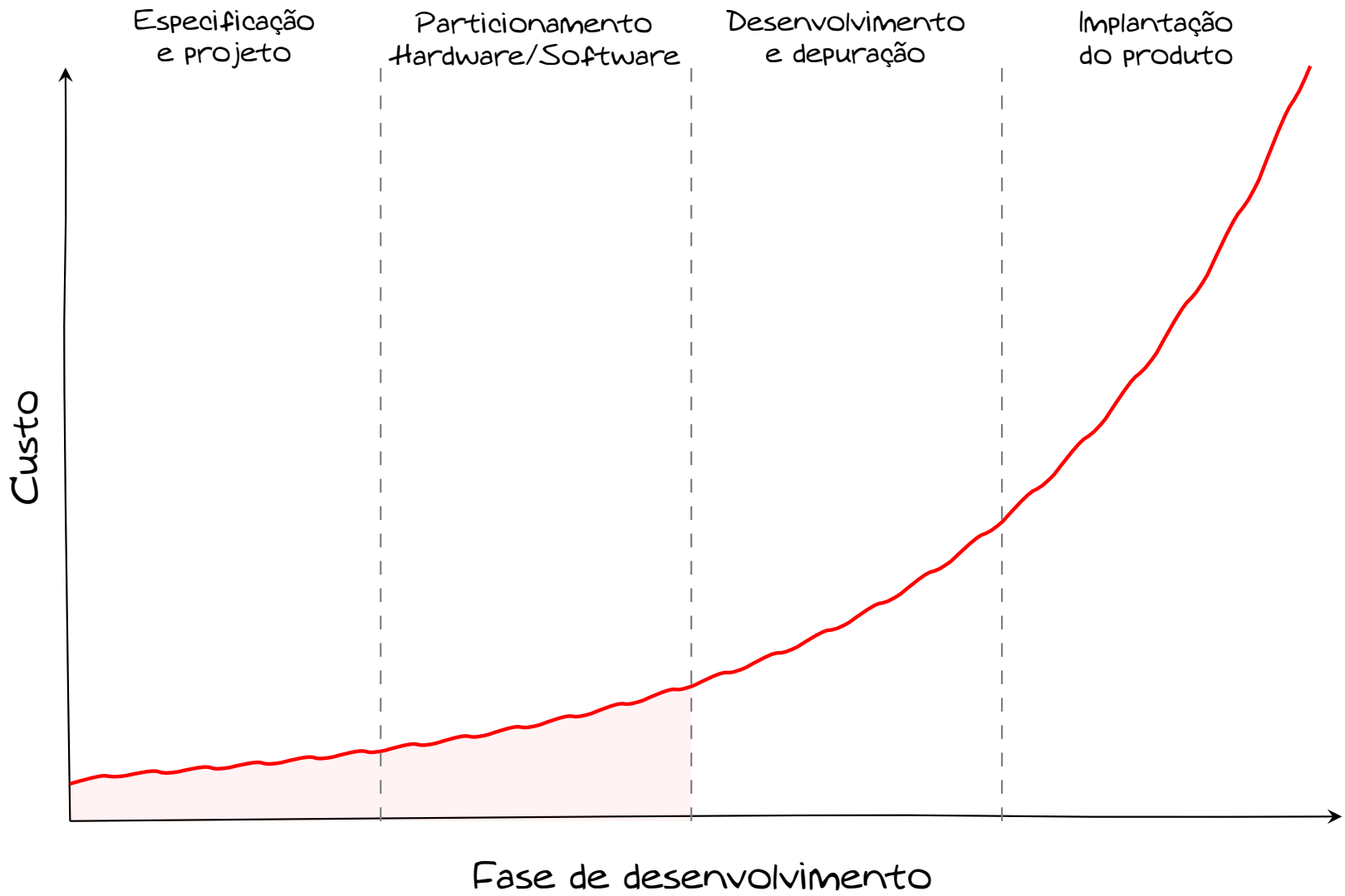
Particionamento

- Hardware ou Software?
  - Custo de correções e mudanças no desenvolvimento



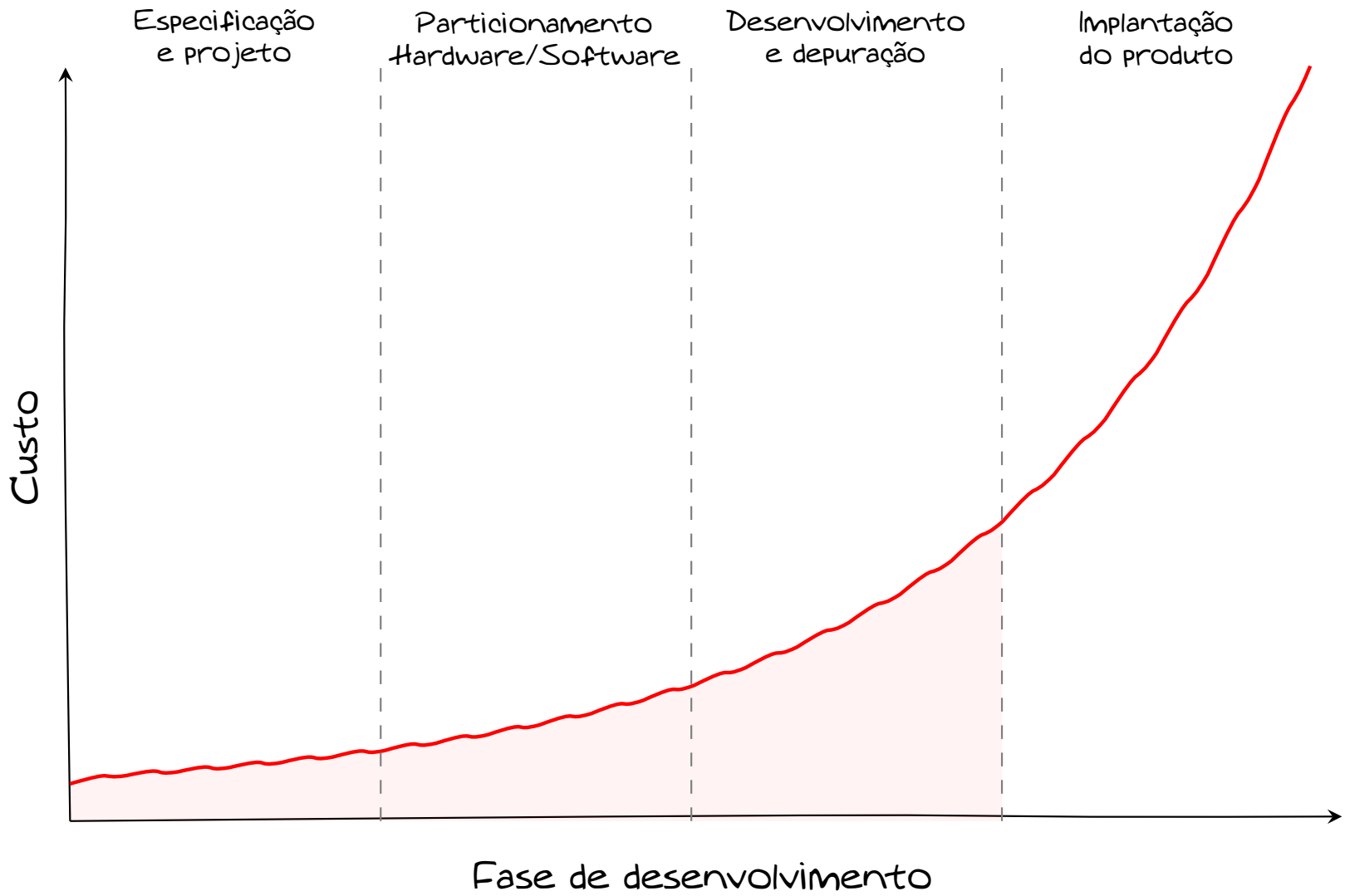
Particionamento

- Hardware ou Software?
  - Custo de correções e mudanças no desenvolvimento



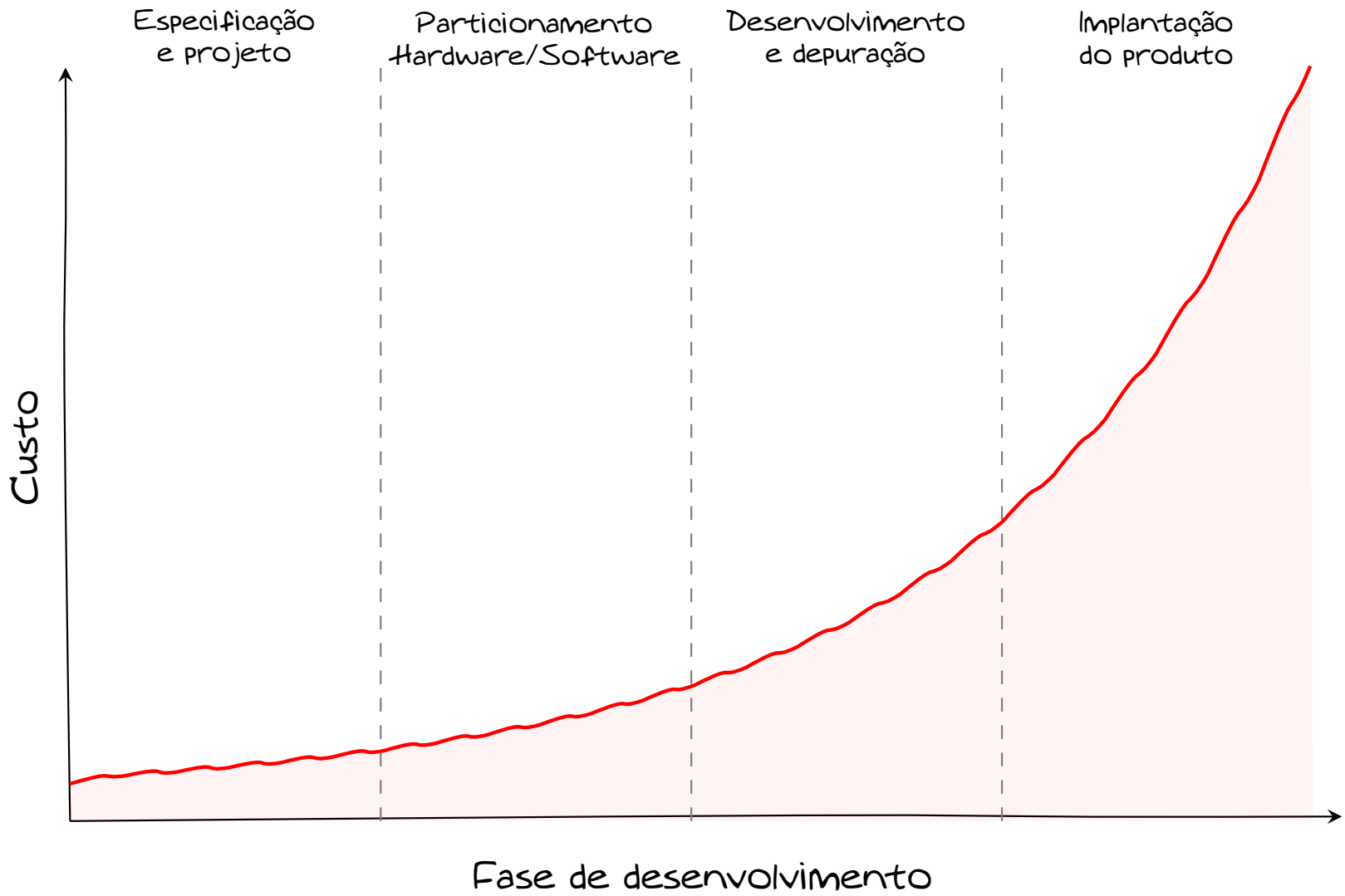
Particionamento

- Hardware ou Software?
  - Custo de correções e mudanças no desenvolvimento



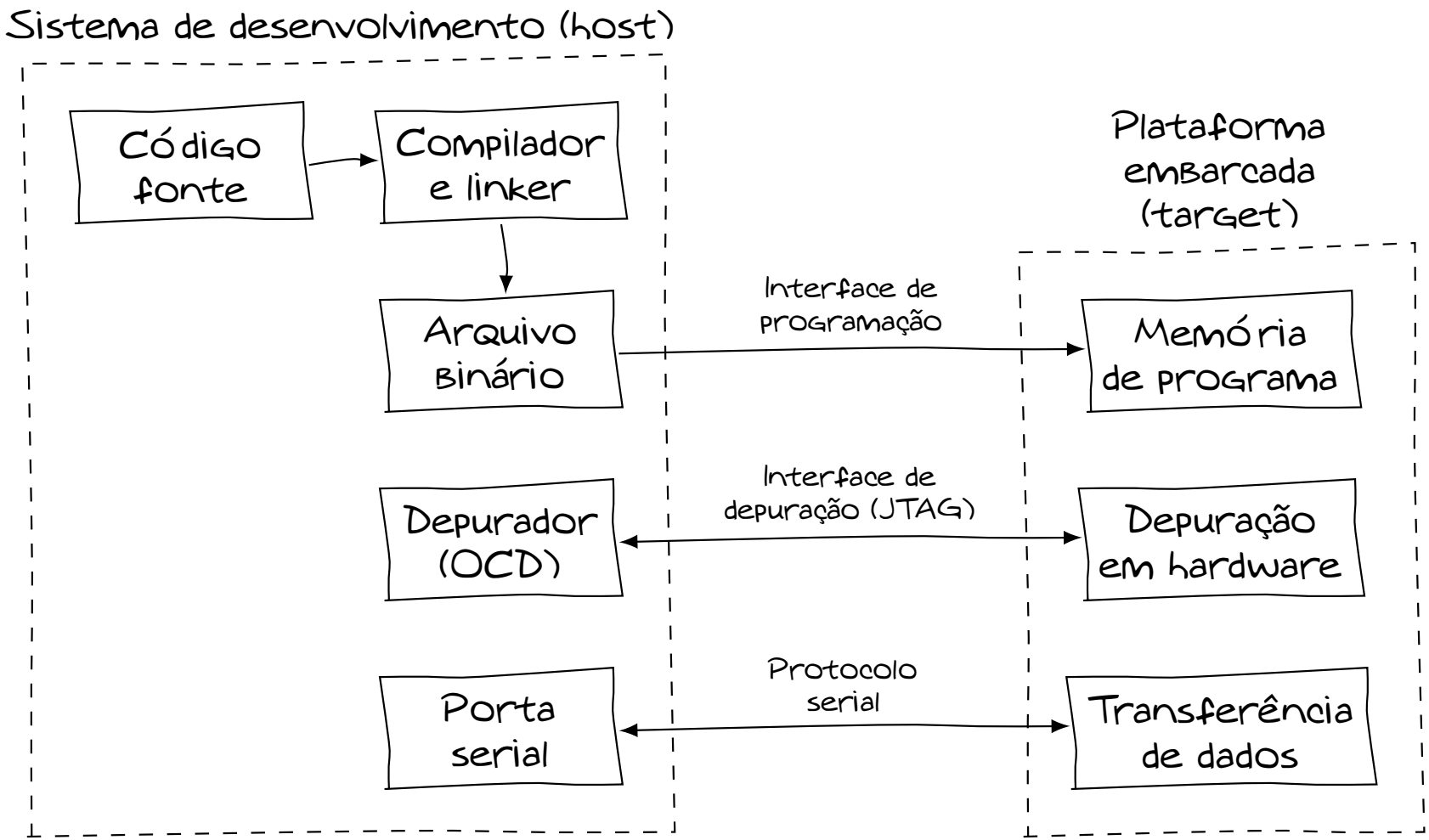
Particionamento

- Hardware ou Software?
  - Custo de correções e mudanças no desenvolvimento



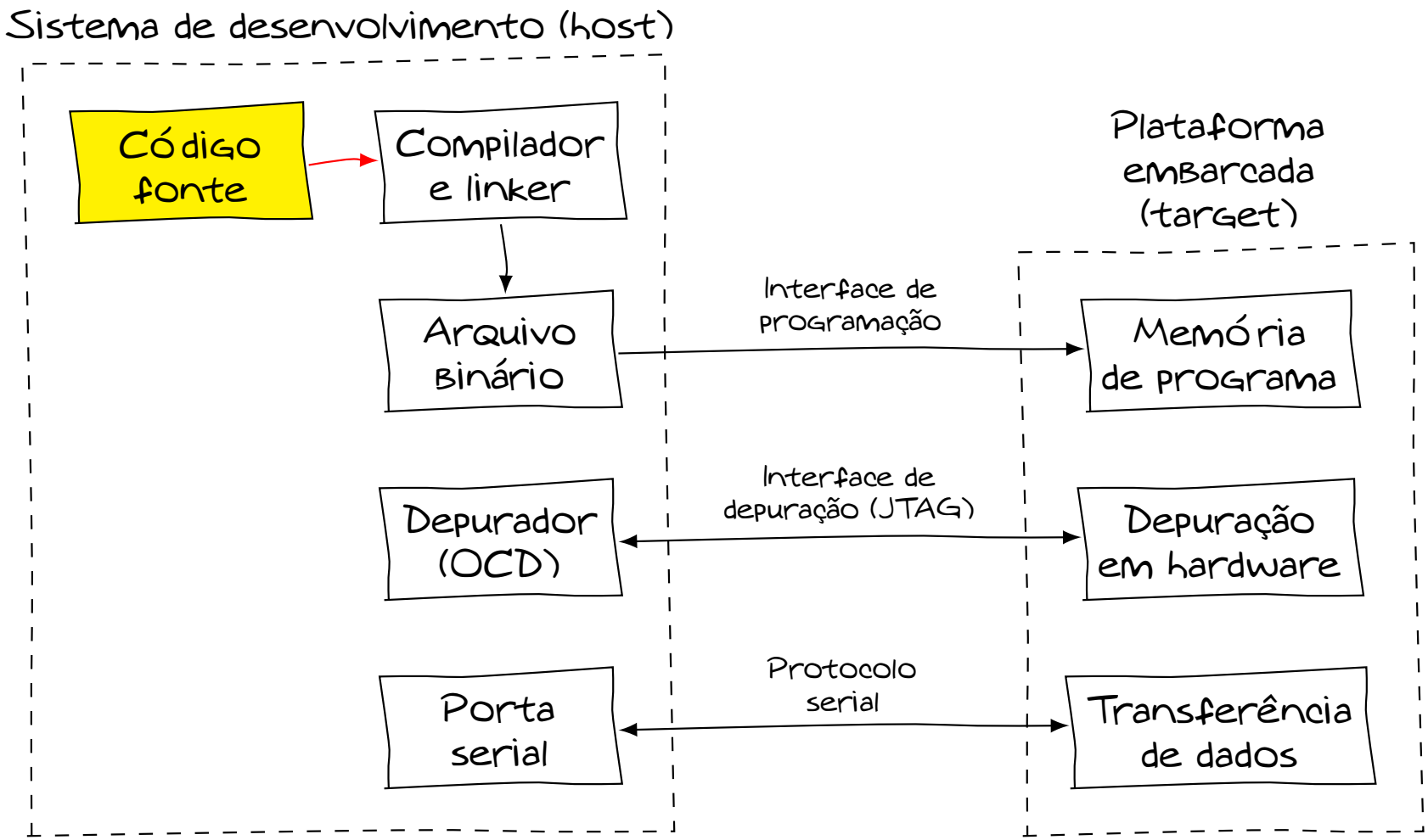
Implementação

- Ambiente de desenvolvimento embarcado



Implementação

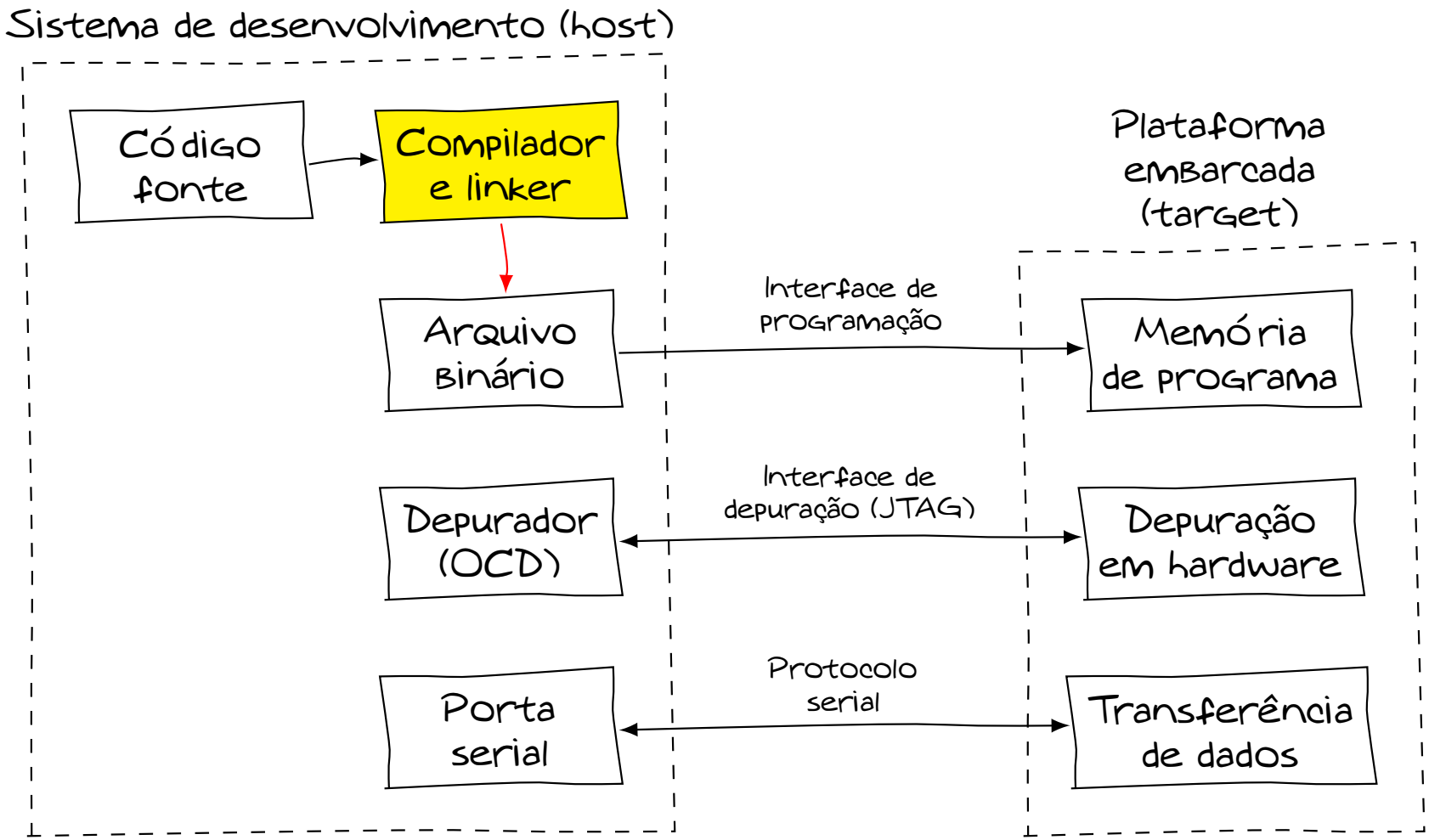
- Ambiente de desenvolvimento embarcado



Geralmente em C e subconjunto de C++

Implementação

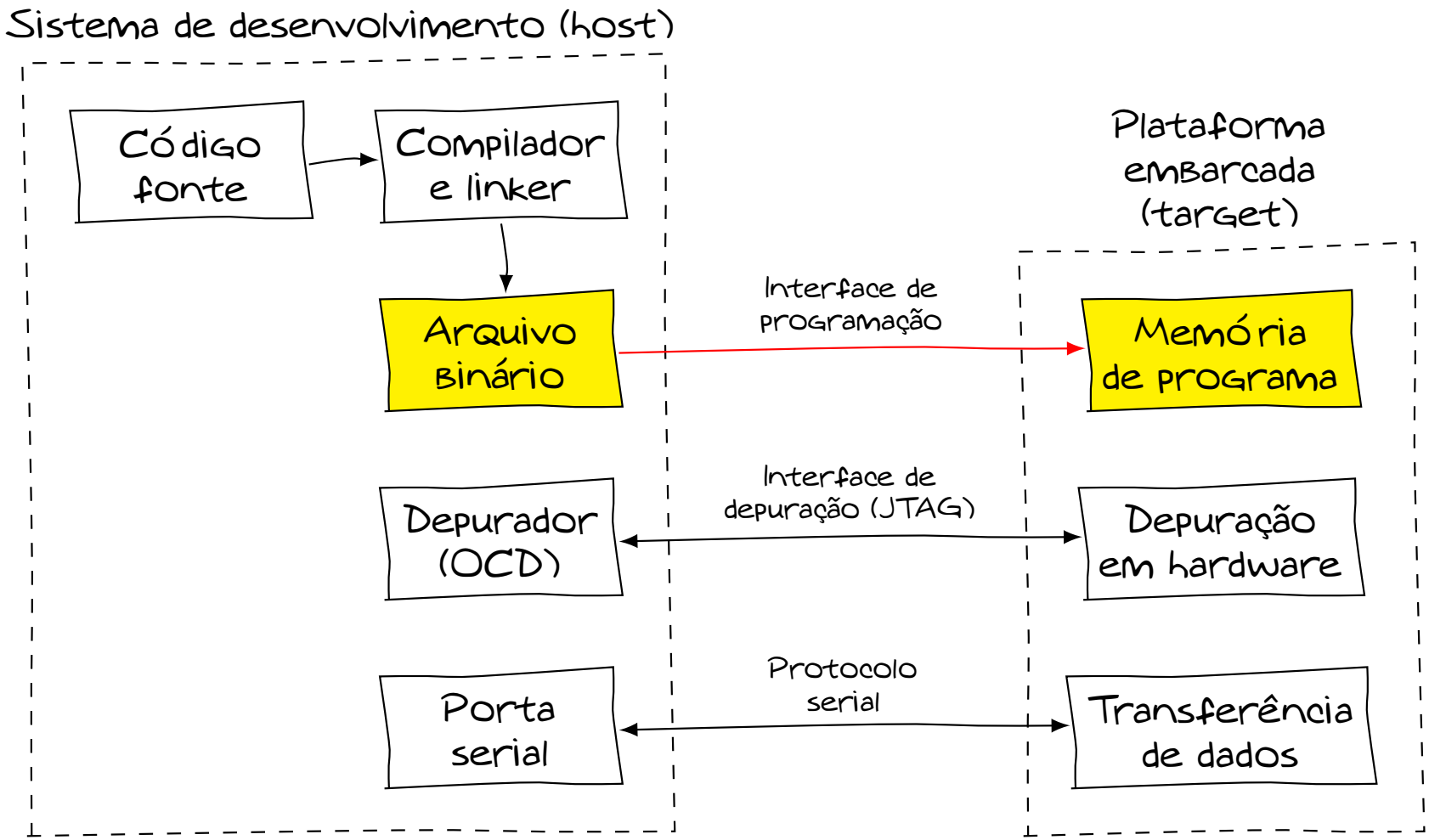
- Ambiente de desenvolvimento embarcado



Compilação cruzada para a arquitetura alvo (*target*)

Implementação

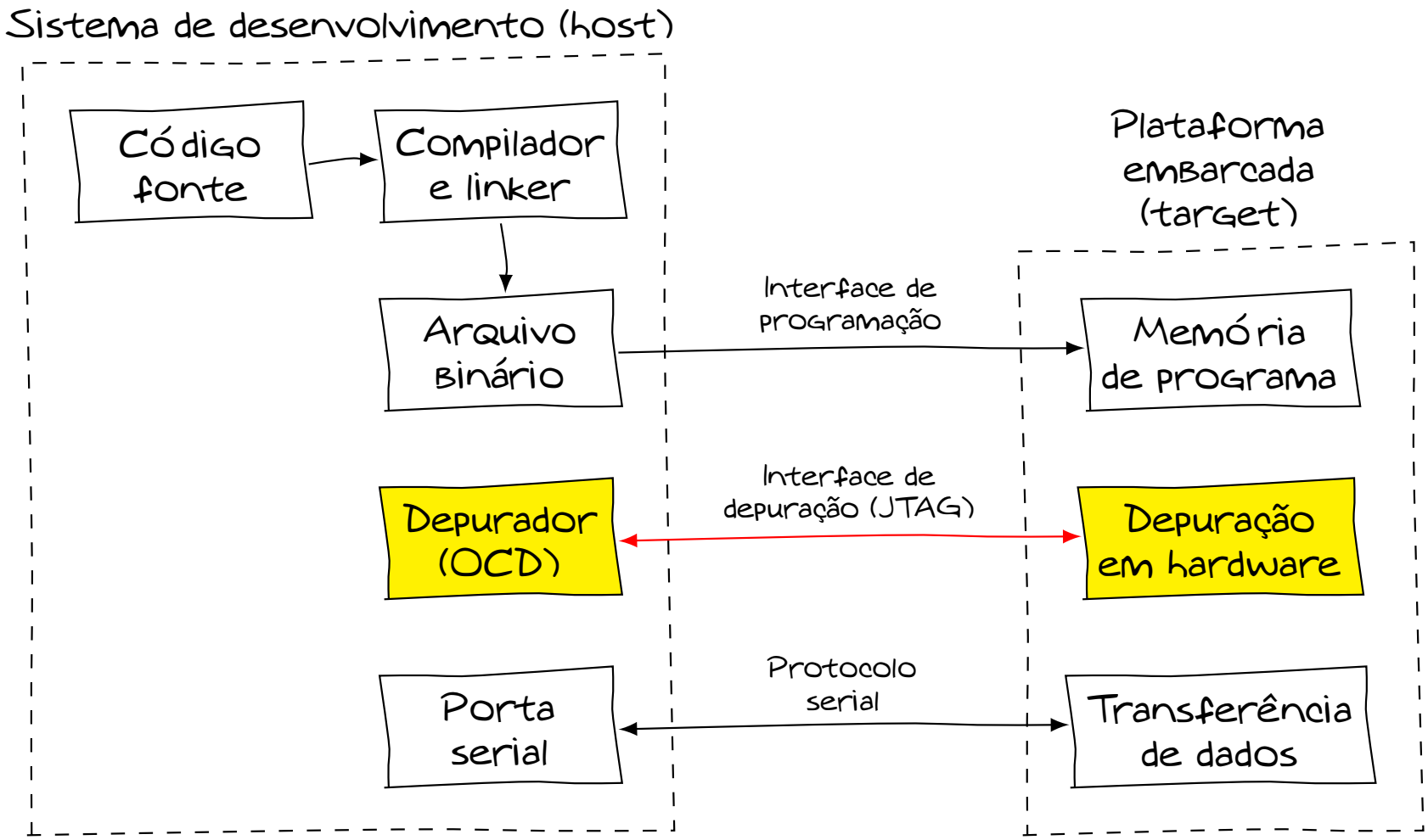
- Ambiente de desenvolvimento embarcado



Programação da memória do processador (Intel HEX)

Implementação

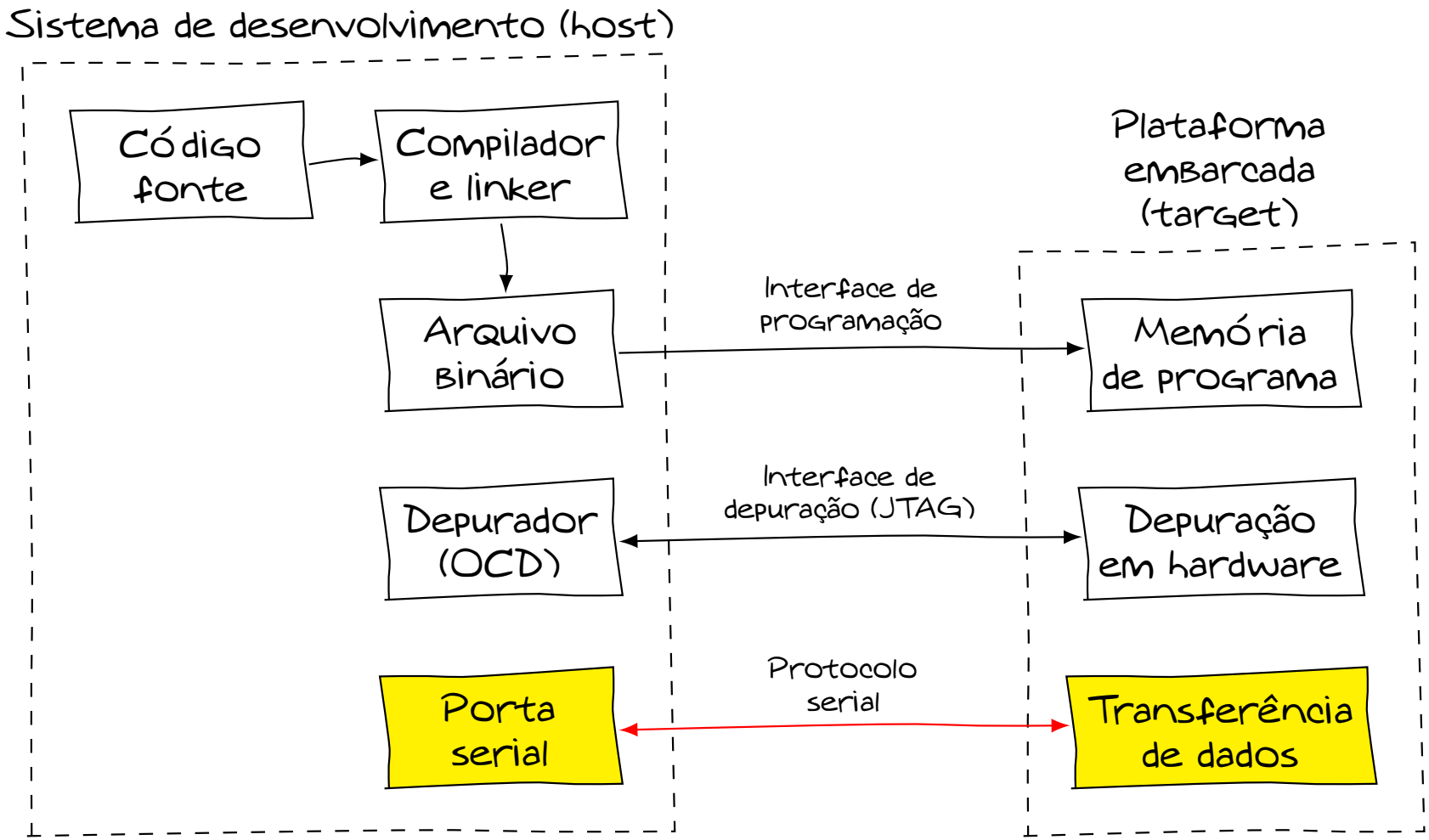
- Ambiente de desenvolvimento embarcado



Depuração na placa de desenvolvimento (limitada)

Implementação

- Ambiente de desenvolvimento embarcado



Troca de mensagens para comunicação e depuração

Implementação

- Desafios do desenvolvimento

Implementação

- Desafios do desenvolvimento
  - Consumo de potência
    - Colocar o processador para dormir (*sleep mode*)
    - Desligar componentes não utilizados da plataforma e reduzir frequência de operação do processador
    - Utilizar interrupção ao invés de *polling*

Implementação

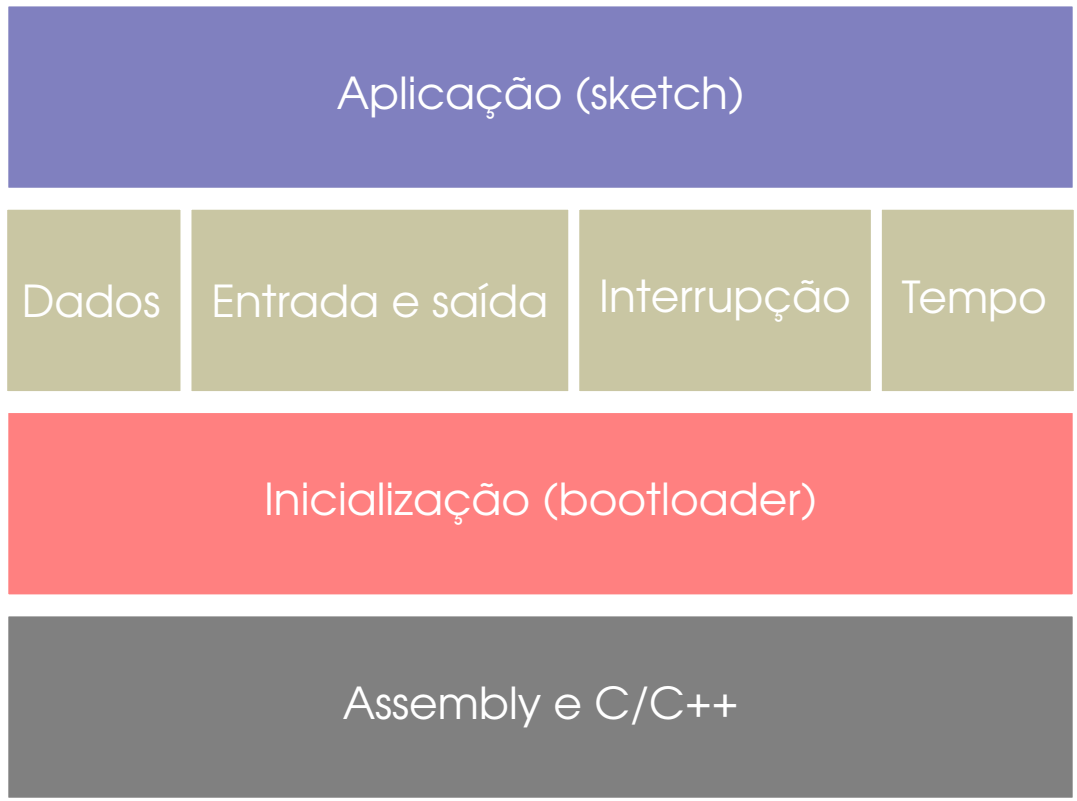
- Desafios do desenvolvimento
  - Consumo de potência
    - Colocar o processador para dormir (*sleep mode*)
    - Desligar componentes não utilizados da plataforma e reduzir frequência de operação do processador
    - Utilizar interrupção ao invés de *polling*
  - Quantidade de memória disponível
    - Alocação eficiente do espaço (~KiB)
    - RAM (dados, *heap* e pilha)
    - ROM (programa e constantes)

Implementação

- Desafios do desenvolvimento
  - Consumo de potência
    - Colocar o processador para dormir (*sleep mode*)
    - Desligar componentes não utilizados da plataforma e reduzir frequência de operação do processador
    - Utilizar interrupção ao invés de *polling*
  - Quantidade de memória disponível
    - Alocação eficiente do espaço (~KiB)
    - RAM (dados, *heap* e pilha)
    - ROM (programa e constantes)
  - Tempo de execução
    - Otimização de trechos importantes (*hot spots*)
    - Múltiplas tarefas concorrentes
    - Restrições de tempo real

Implementação

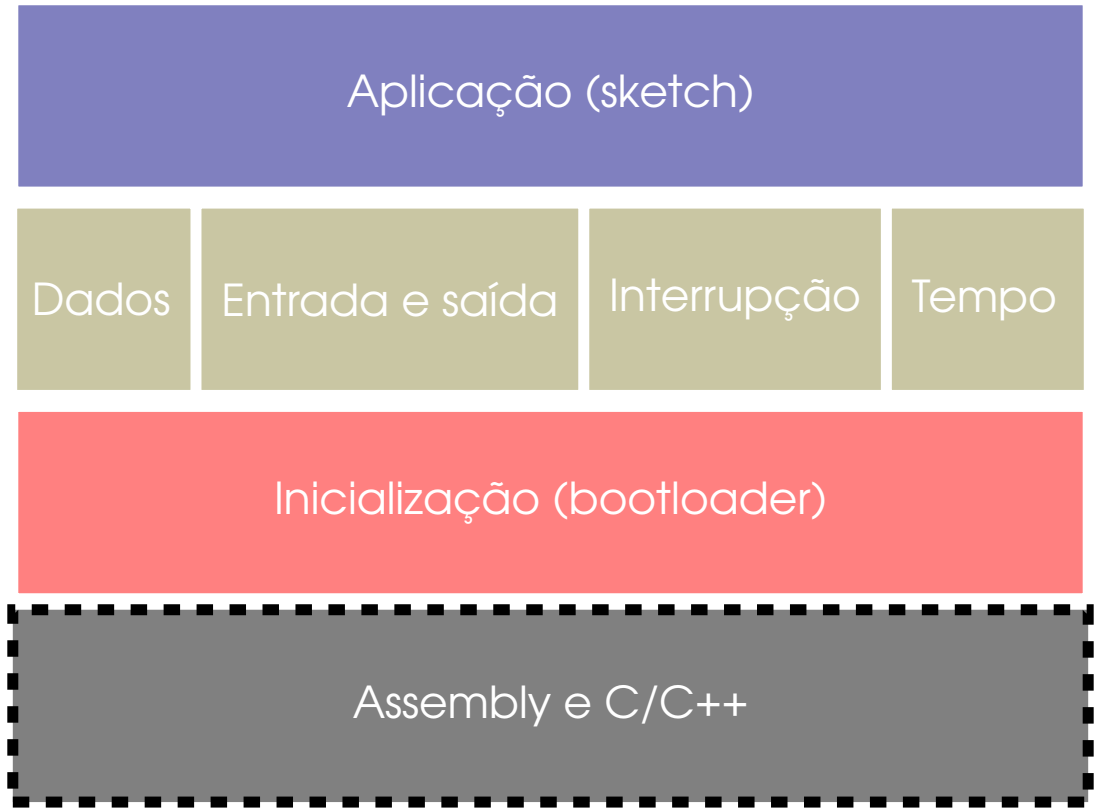
- Infraestrutura de software (*framework*)



Arquitetura de software em camadas (API)

Implementação

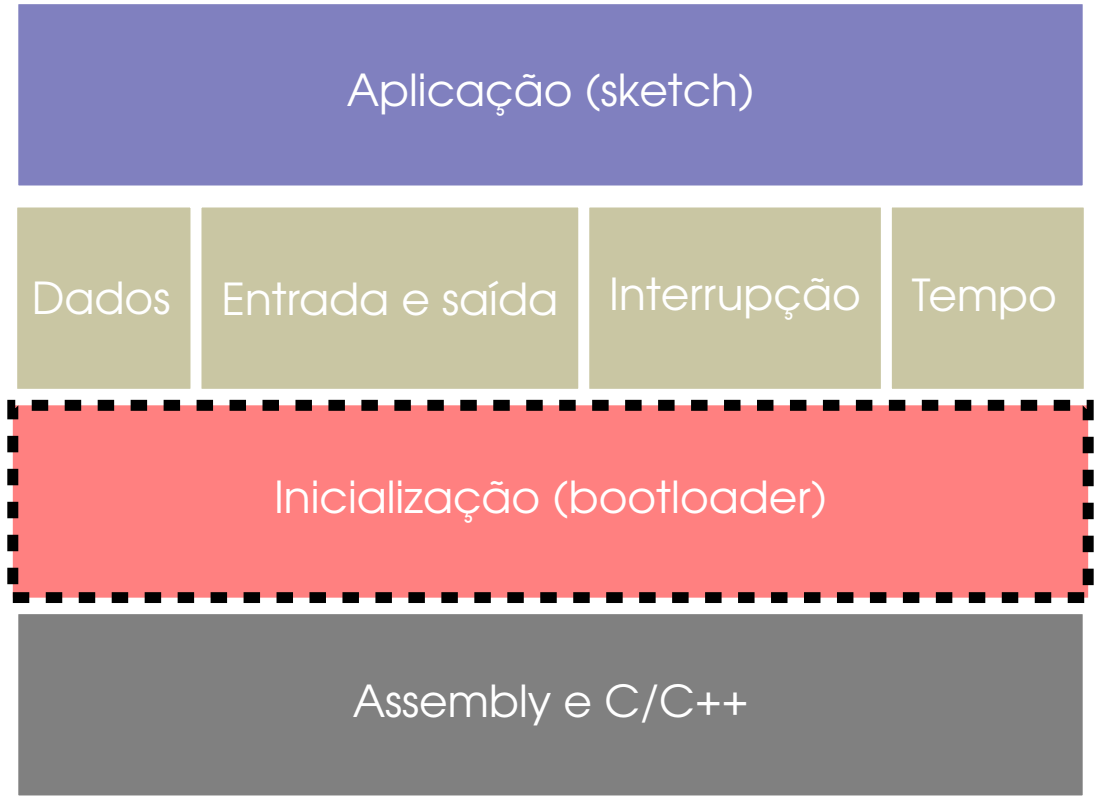
- Infraestrutura de software (*framework*)



Linguagens de programação suportadas pelas ferramentas de desenvolvimento

Implementação

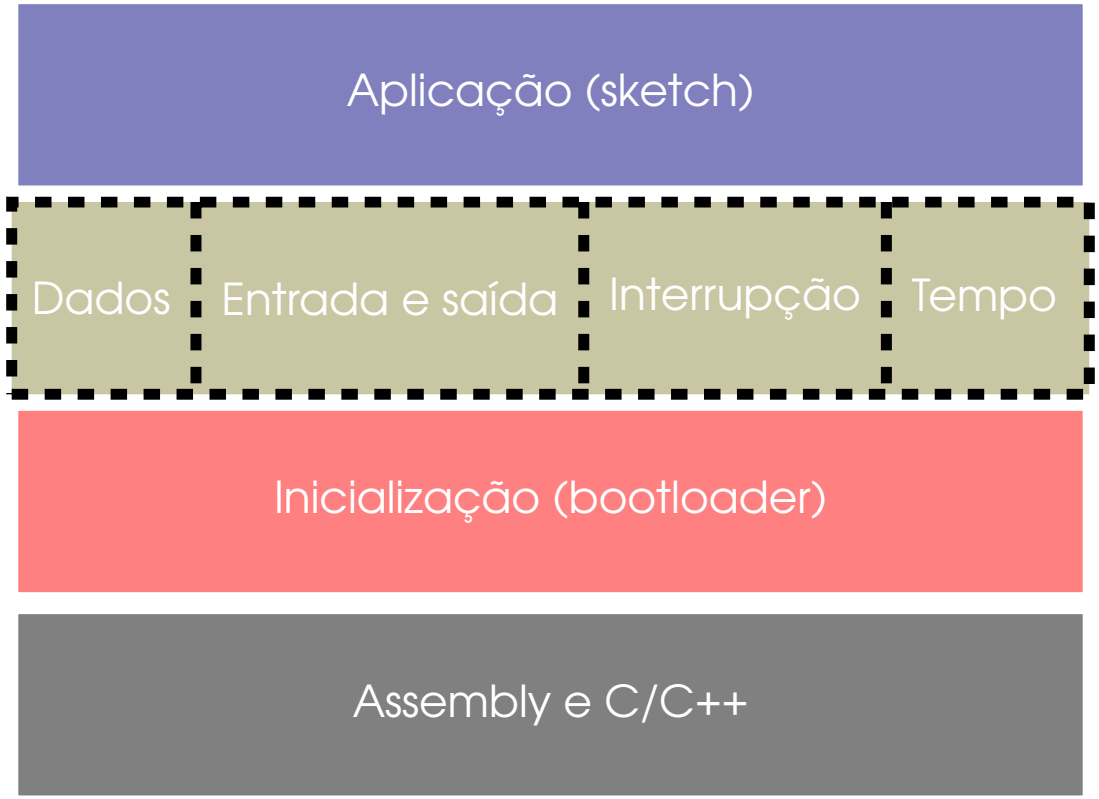
- Infraestrutura de software (*framework*)



Inicialização da plataforma
e chamada da função principal

Implementação

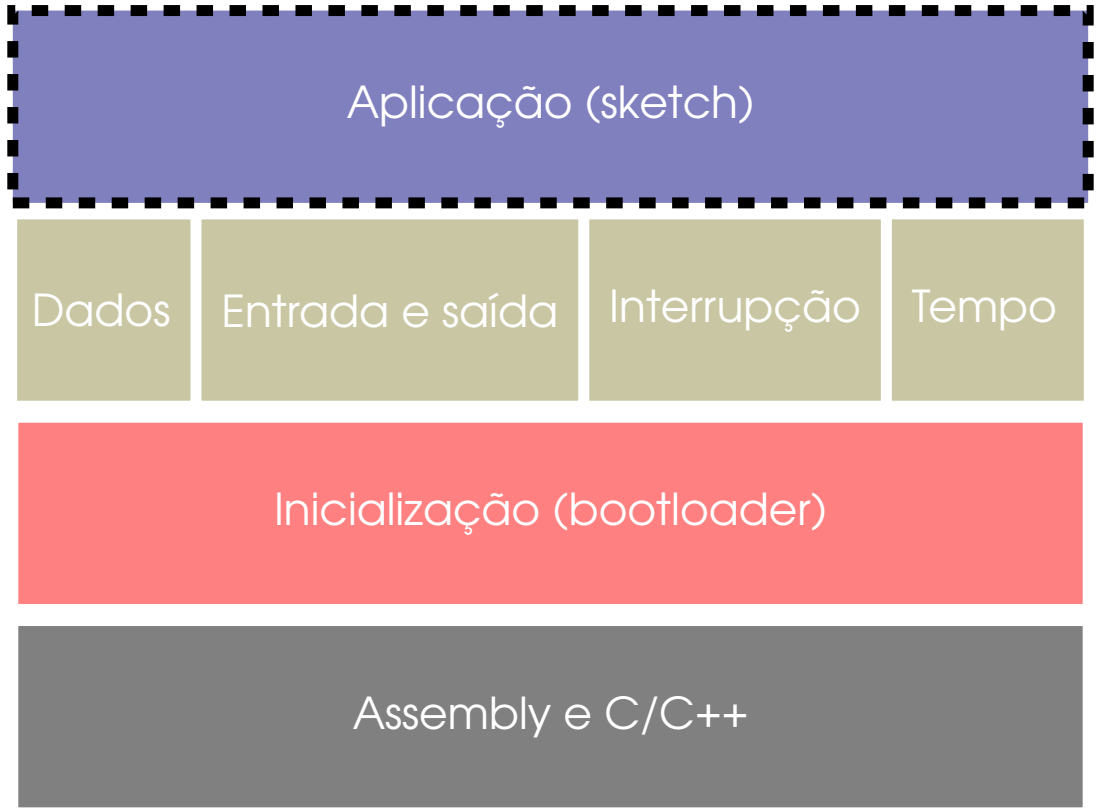
- Infraestrutura de software (*framework*)



Bibliotecas para tipos de dados, interface de E/S, gerenciamento de interrupção e temporização

Implementação

- Infraestrutura de software (*framework*)



Código fonte com as funcionalidades implementadas pelo sistema

Implementação

- Organização do código fonte da aplicação

```
1  // Função principal
2  int main() {
3      // Configuração da plataforma
4      setup();
5      // Laço infinito
6      for(;;) {
7          // Execução das operações
8          loop();
9      }
10     // Nunca será executado
11     return 0;
12 }
```

Implementação

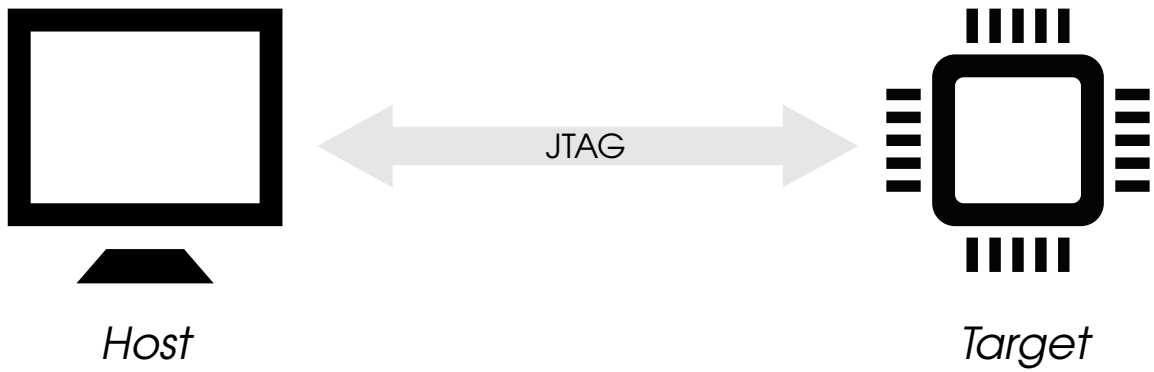
- Organização do código fonte da aplicação

```
1  // Função principal
2  int main() {
3      // Configuração da plataforma
4      setup();
5      // Laço infinito
6      for(;;) {
7          // Execução das operações
8          loop();
9      }
10     // Nunca será executado
11     return 0;
12 }
```

Por que o código precisa ficar em laço infinito?

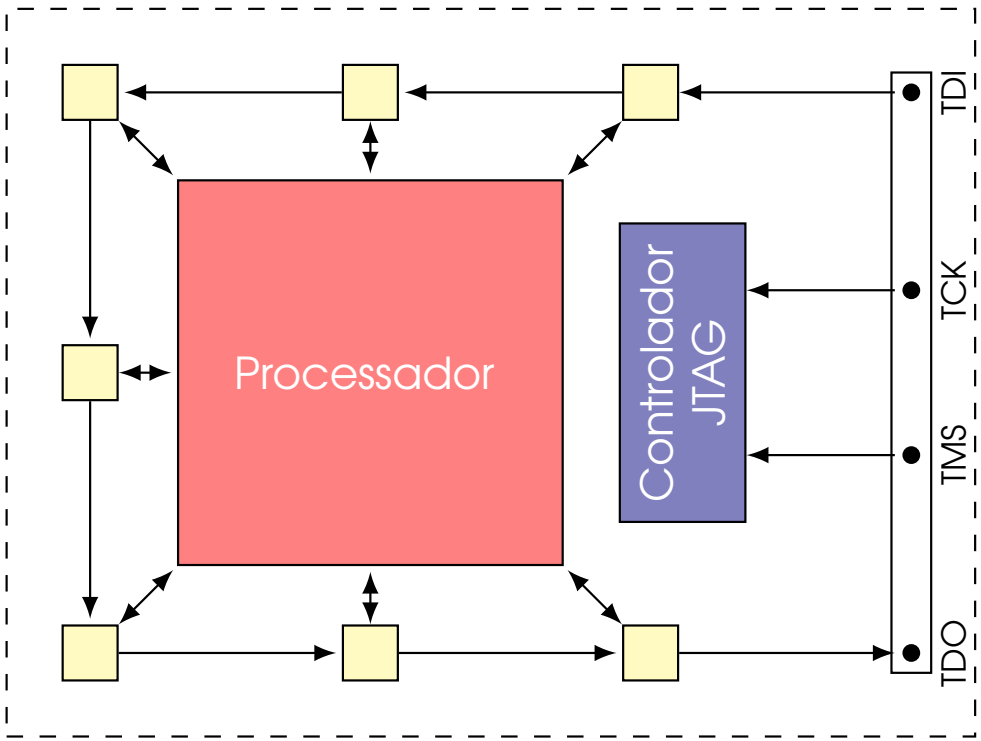
Depuração

- *On-Chip Debugging* (OCD)
  - Controle de fluxo em tempo de execução diretamente na plataforma de hardware
    - Conteúdo de registradores e de memória
    - Execução iterativa controlada pelo usuário
    - Pontos de parada do software (*breakpoints*)



Depuração

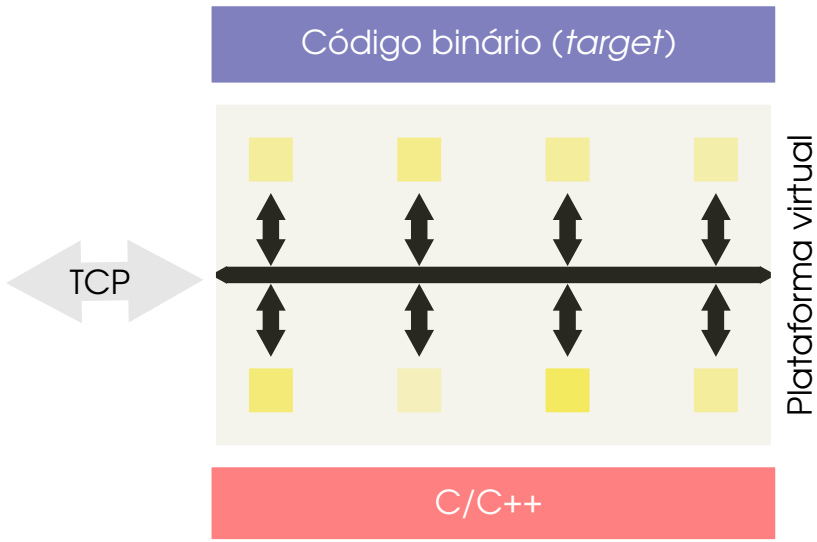
- *Joint Test Action Group* (JTAG)
  - Foi criado pela indústria de teste de placas de computadores (IEEE 1149.1) e permite a análise ativa ou passiva dos componentes do sistema
  - Consiste de um laço contínuo e serial de pontos de prova (*probes*) para geração de fluxo de bits (representação do estado interno dos circuitos)



Depuração

- Plataforma virtual
  - É um ambiente de simulação para prototipação virtual de sistemas (hardware + software integrados)
  - Possibilita explorar rapidamente o espaço de projeto, mesmo quando o hardware não está disponível

```
$ gdb
GNU gdb (GDB) 8.2.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later
<http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and
redistribute it.
...
For help, type "help".
Type "apropos word"to search for commands related
to "word".
(gdb)
```



Sistema de desenvolvimento (*host*)

Depuração

- Simulação nativa
  - O código fonte da aplicação é compilado, depurado e executado nativamente (*host*)
  - Para suportar a execução no sistema de desenvolvimento (*host*), o Software dependente do Hardware (HdS) precisa ser portado (*target* $\rightarrow$ *host*)