
Universidade Federal da Bahia
Escola Politécnica
2022.2

# Sistema supervisório com Node-RED via Modbus

**Discentes:**
André Paiva
Caroline Alves
Jeisiane Macedo
Pedro Augusto Correia

Salvador, 01 de dezembro de 2022

# Introdução

Nos últimos anos, presenciamos uma crescente procura das empresas pela adoção da automatização de processos.

Tal automatização promove redução de custos e aumento na produtividade.

Neste contexto, destaca-se a importância dos sistemas supervisórios.

# Importância de Sistemas Supervisórios

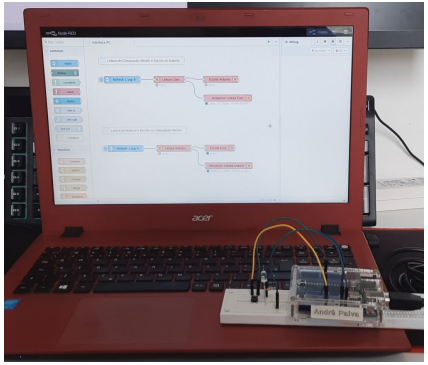
- Monitoramento de dados de um processo
- Controle de dados de um processo
- Aumento da produtividade
- Redução de custos

# Projeto desenvolvido

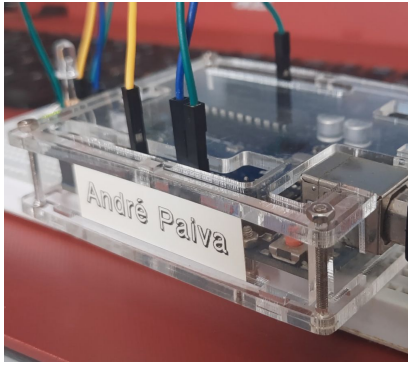
- Prova de conceito de um sistema supervisório
- Demonstração de funcionamento e configuração

# Componentes de Hardware

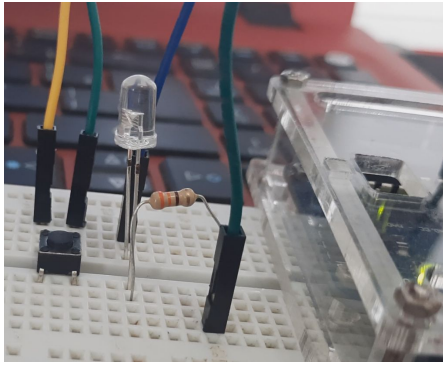
Core PC



Interface PC

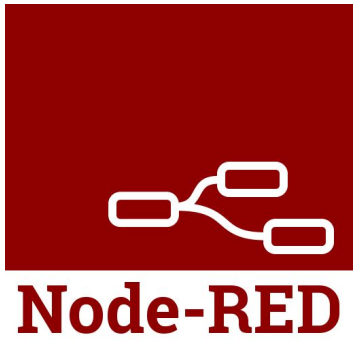



Arduino



Circuito

# Componentes de Software




Supervisório/IHM
e CLP


Comunicação entre
Supervisório e CLP


Comunicação CLP e
arduino

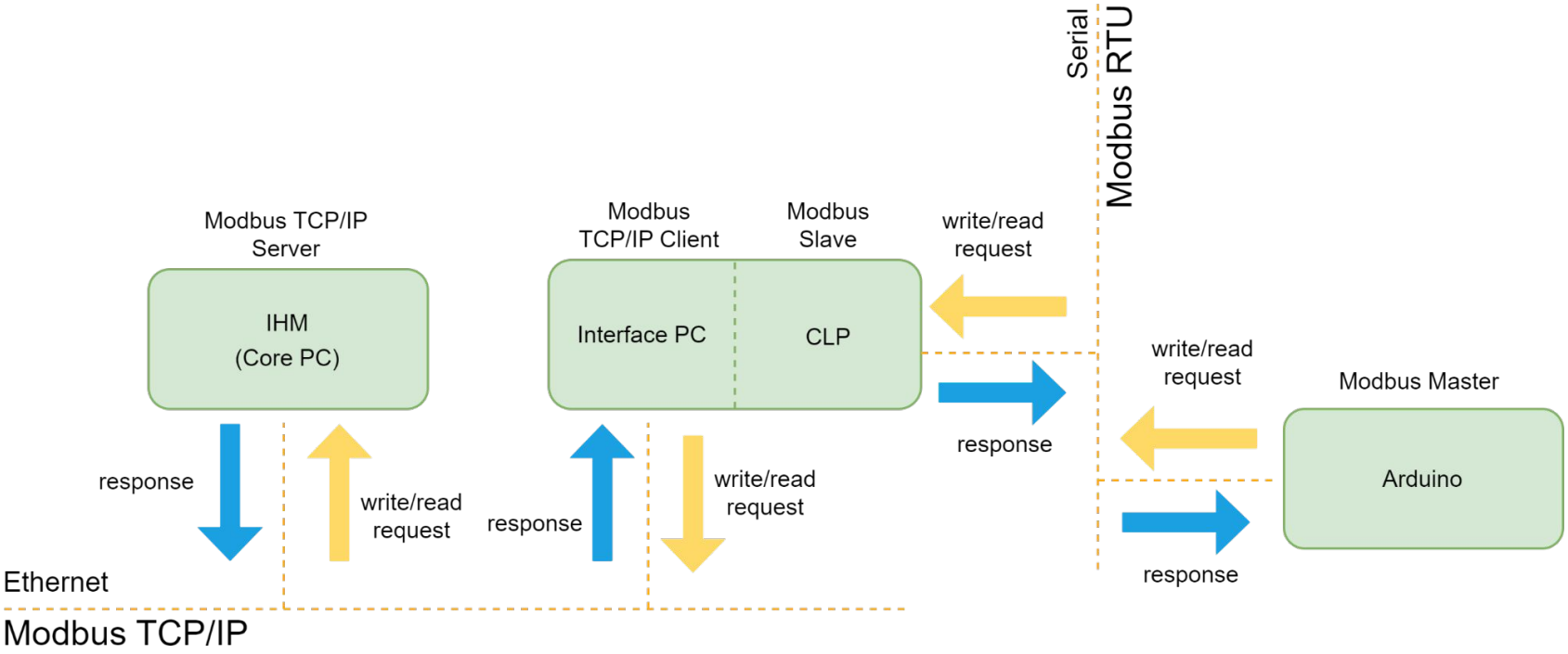
# Node-RED

- Ferramenta versátil que facilita a criação de aplicações IoT desenvolvida pela IBM.
- Possui uma abordagem gráfica para programação baseada em fluxos através do navegador.
- As conexões são realizadas por nós com propósitos definidos de execução. Cada nó recebe o dado, o processa de acordo com sua função e passa adiante.
- O projeto de fluxo pode ser exportado via JSON.

# Modbus

- É um protocolo aberto para transmissão de informações entre dispositivos, via portas serial ou Ethernet, desenvolvido pela Modicon e pertencente atualmente à Schneider.
- É um dos protocolos mais utilizados na automação industrial devido à sua facilidade de implementação.
- Arquitetura Mestre/Escravo (ou Cliente/Servidor).
- A transmissão pode ser feita em modo RTU ou ASCII. Esses modos não se comunicam entre si. A versão TCP/IP utiliza o modo RTU.

# Arquitetura da aplicação



# Desenvolvimento do projeto

# Core PC

O Core PC é um PC que fica em uma das extremidades da rede, sendo utilizado como um Modbus Server e como interface humano-máquina por meio de um Dashboard que possibilita controle e monitoramento.

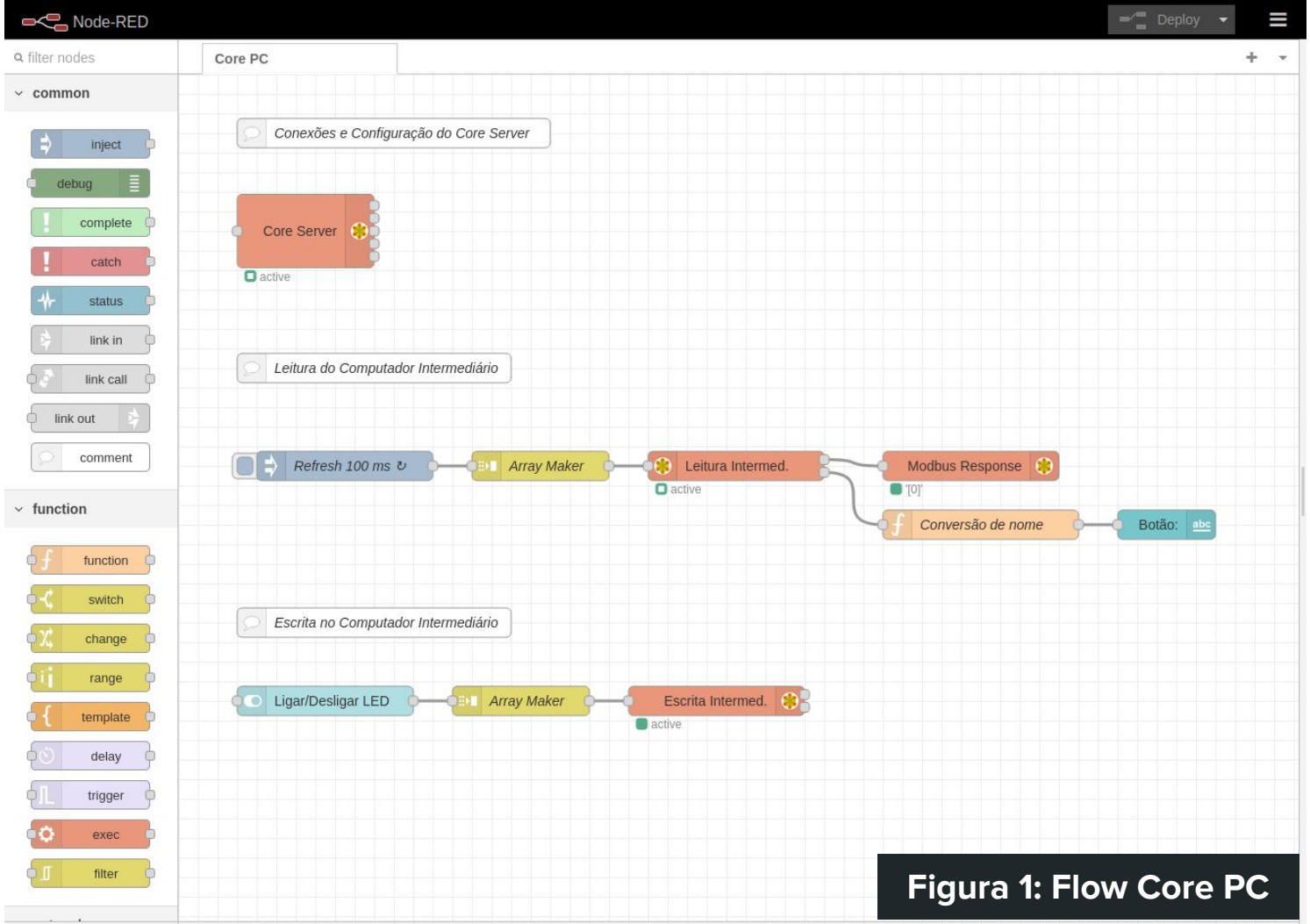



**Figura 1: Flow Core PC**

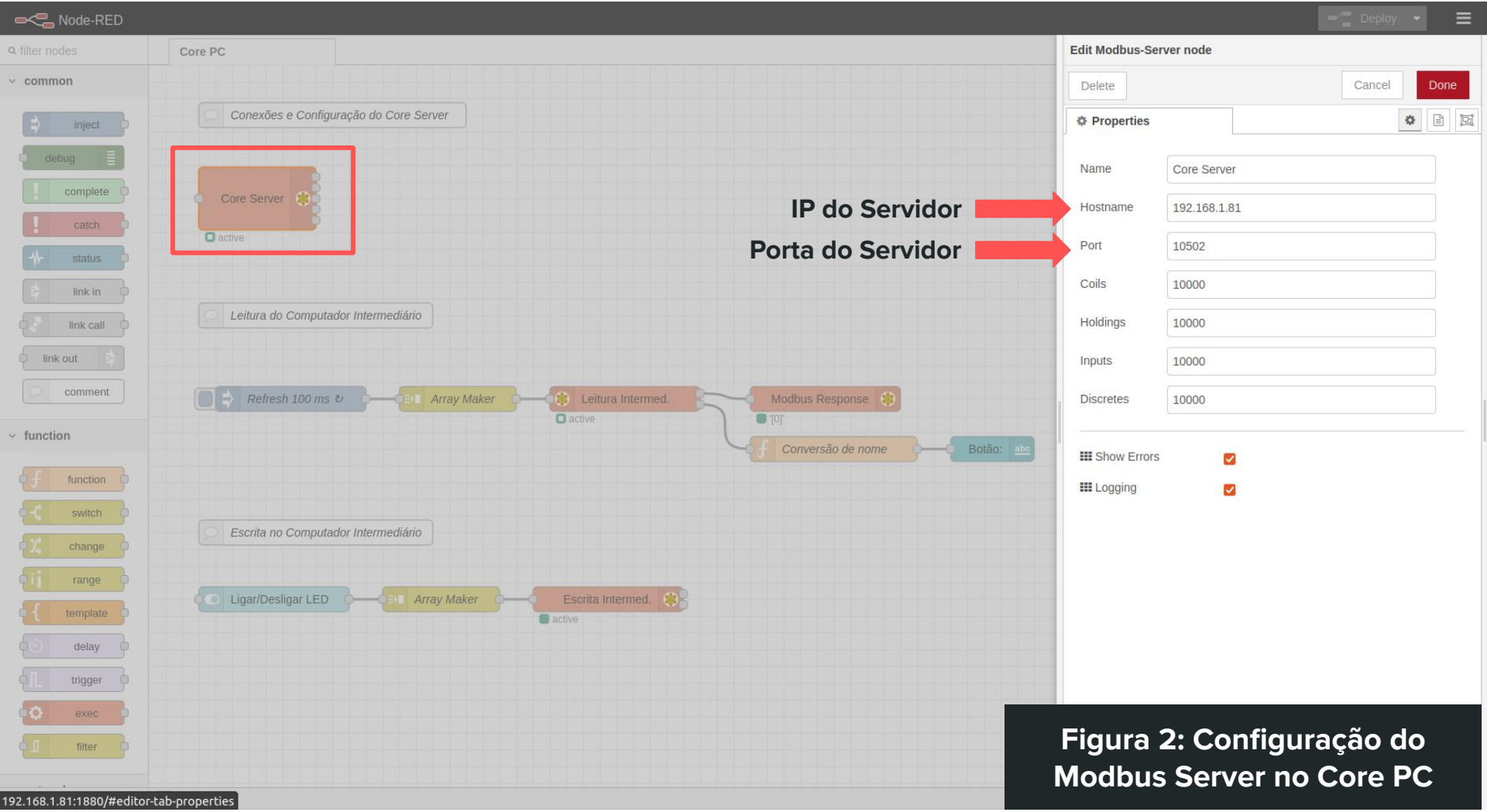



**Figura 2: Configuração do Modbus Server no Core PC**

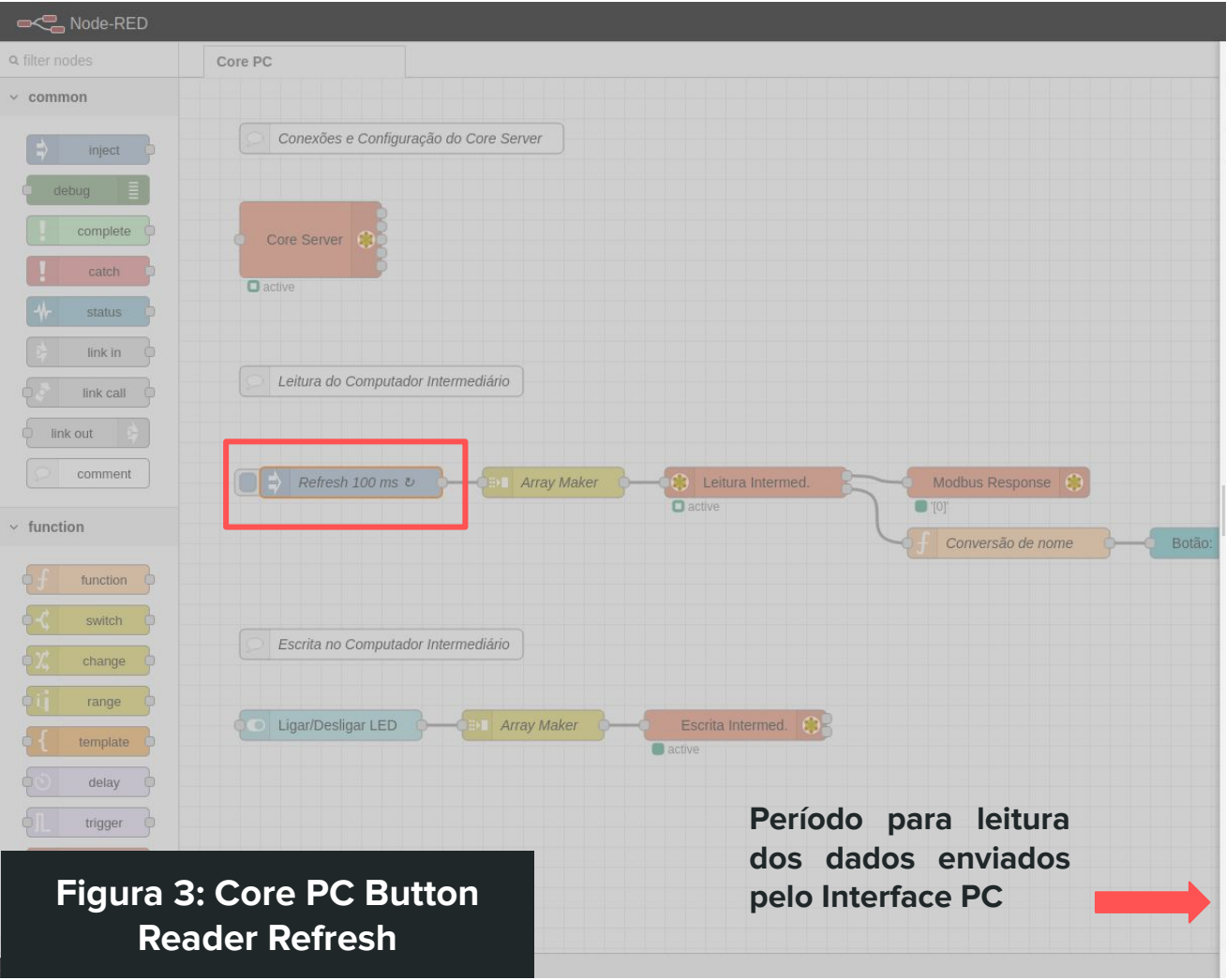



**Figura 3: Core PC Button Reader Refresh**

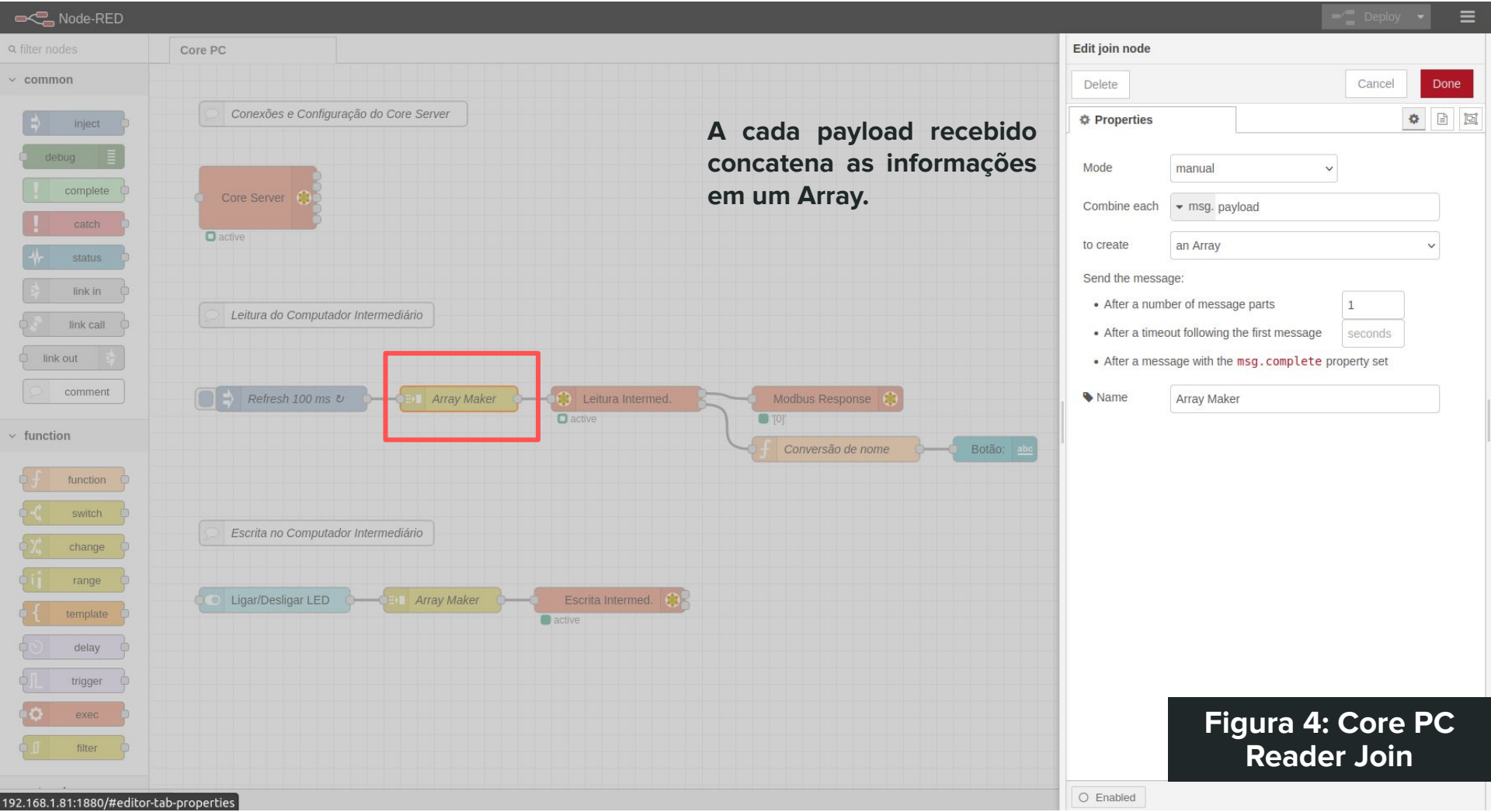



**Figura 4: Core PC Reader Join**

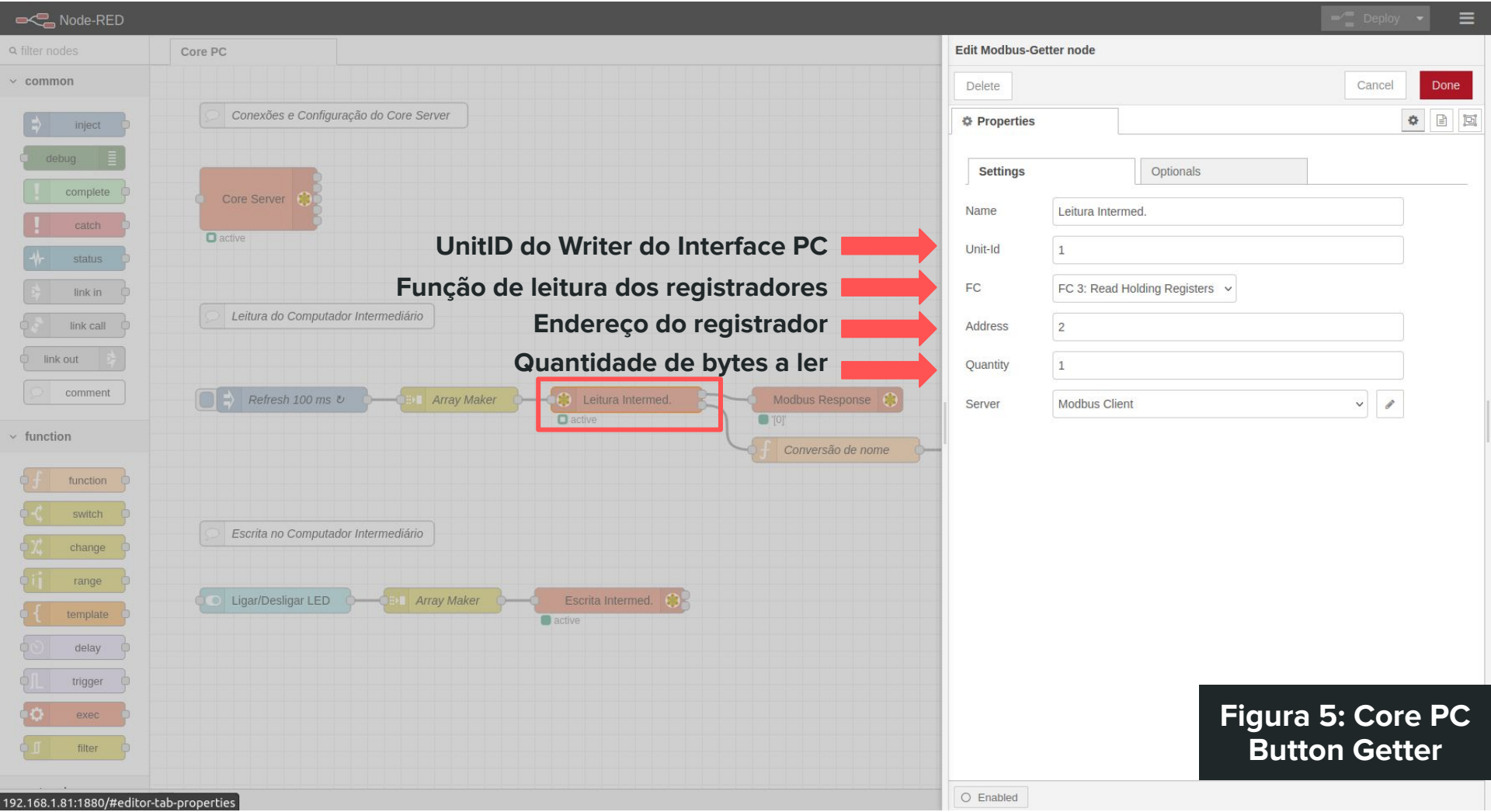



**Figura 5: Core PC Button Getter**

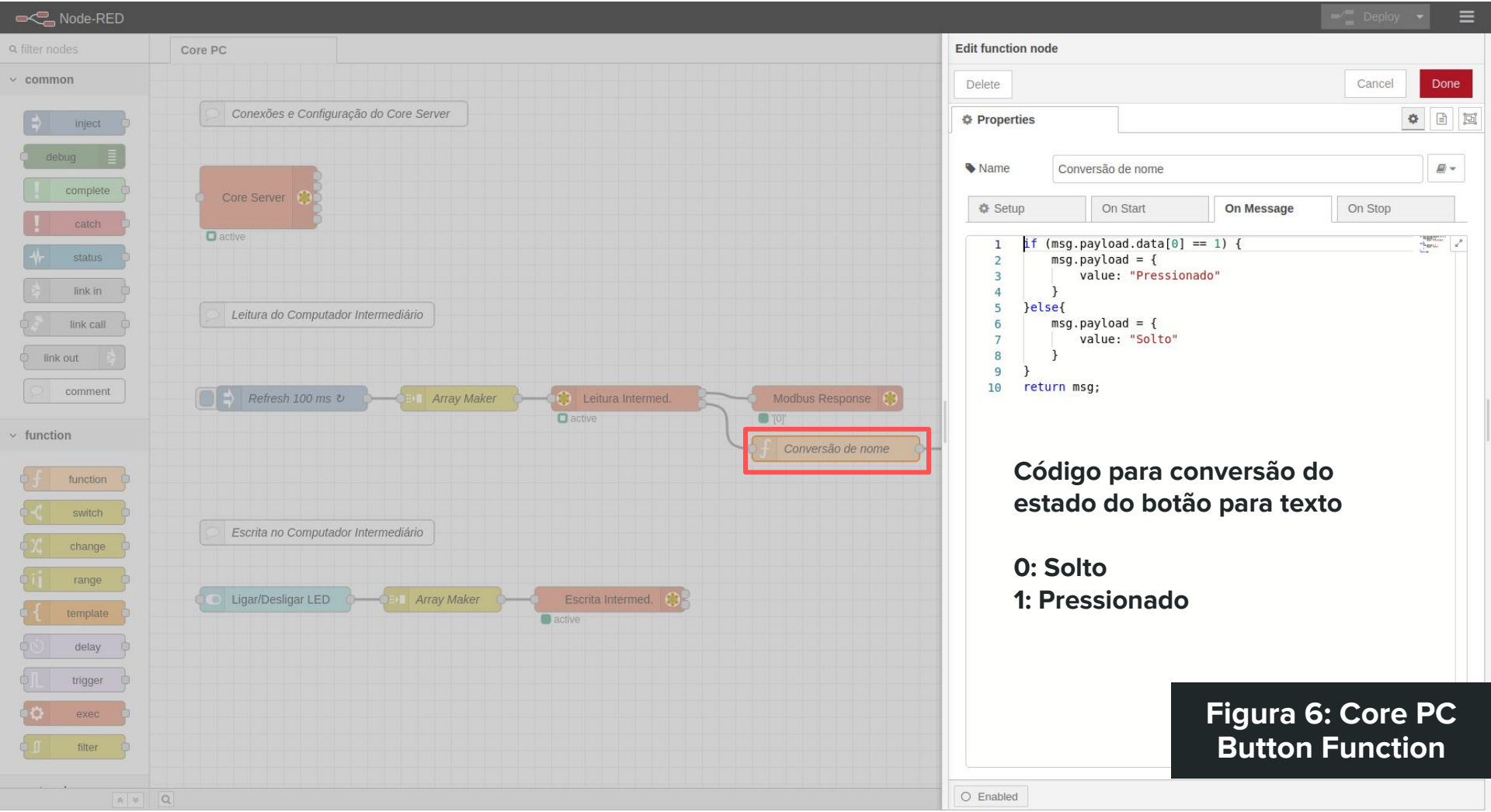



**Figura 6: Core PC Button Function**

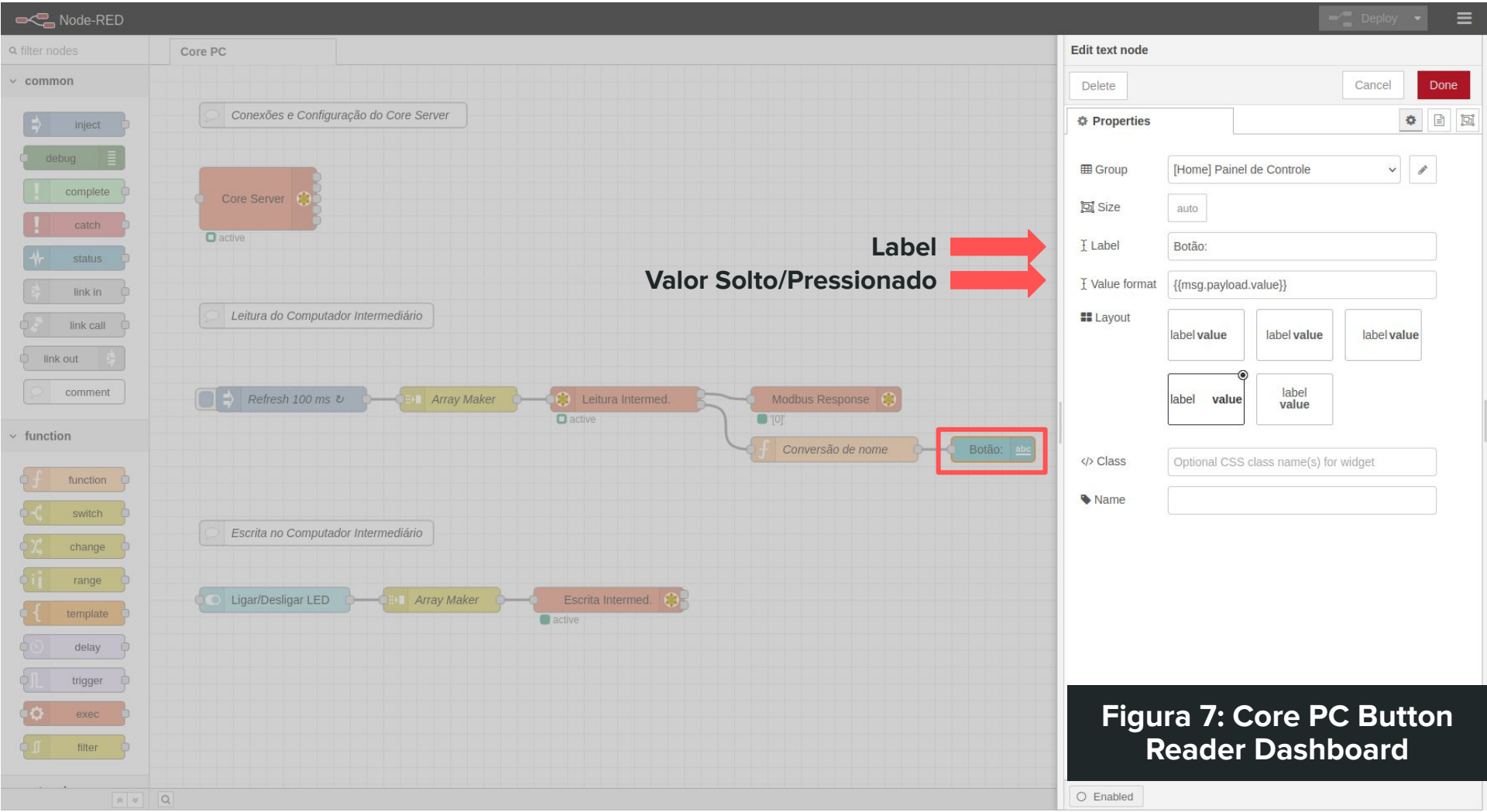



**Figura 7: Core PC Button Reader Dashboard**

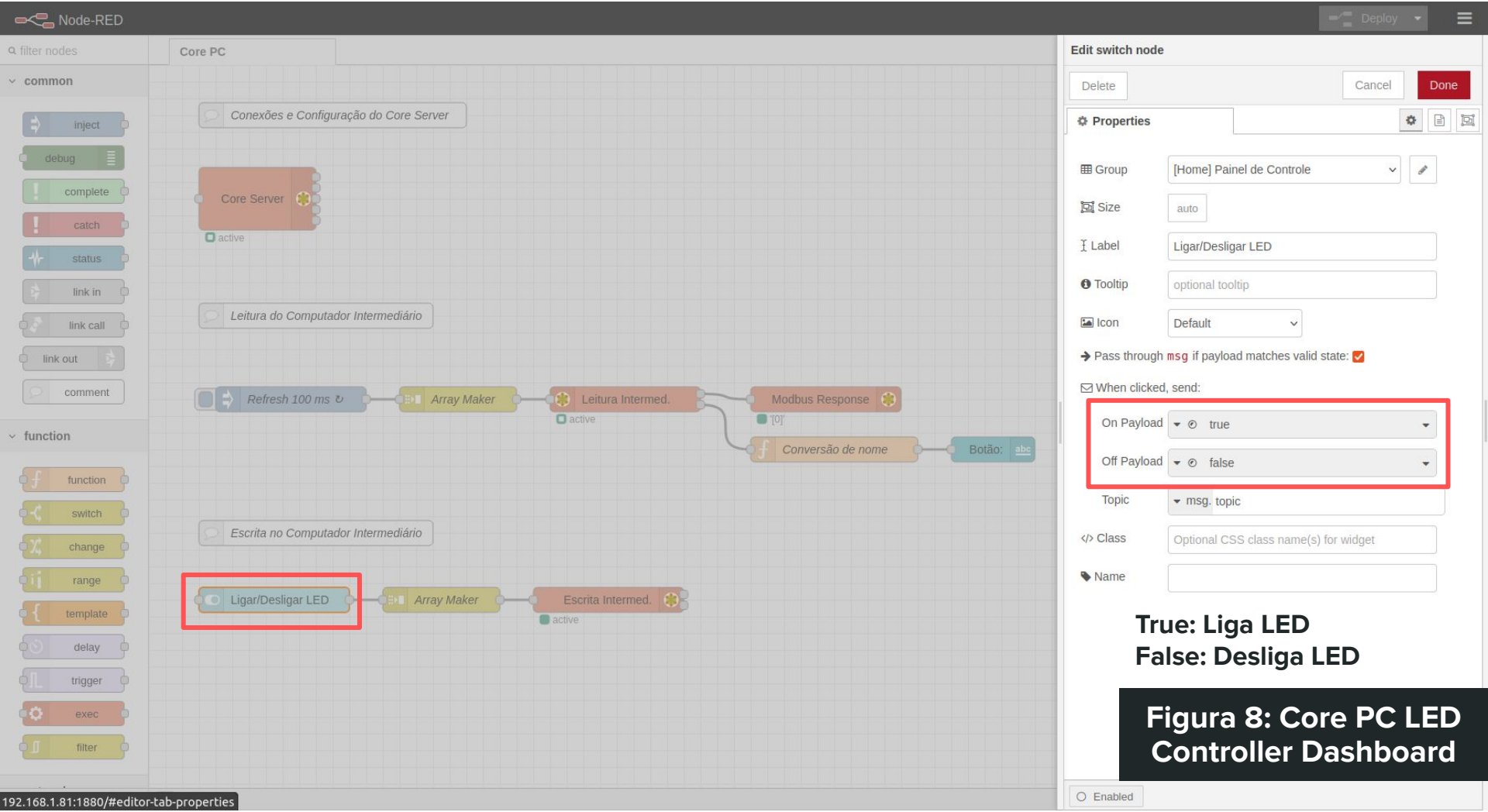



**Figura 8: Core PC LED Controller Dashboard**

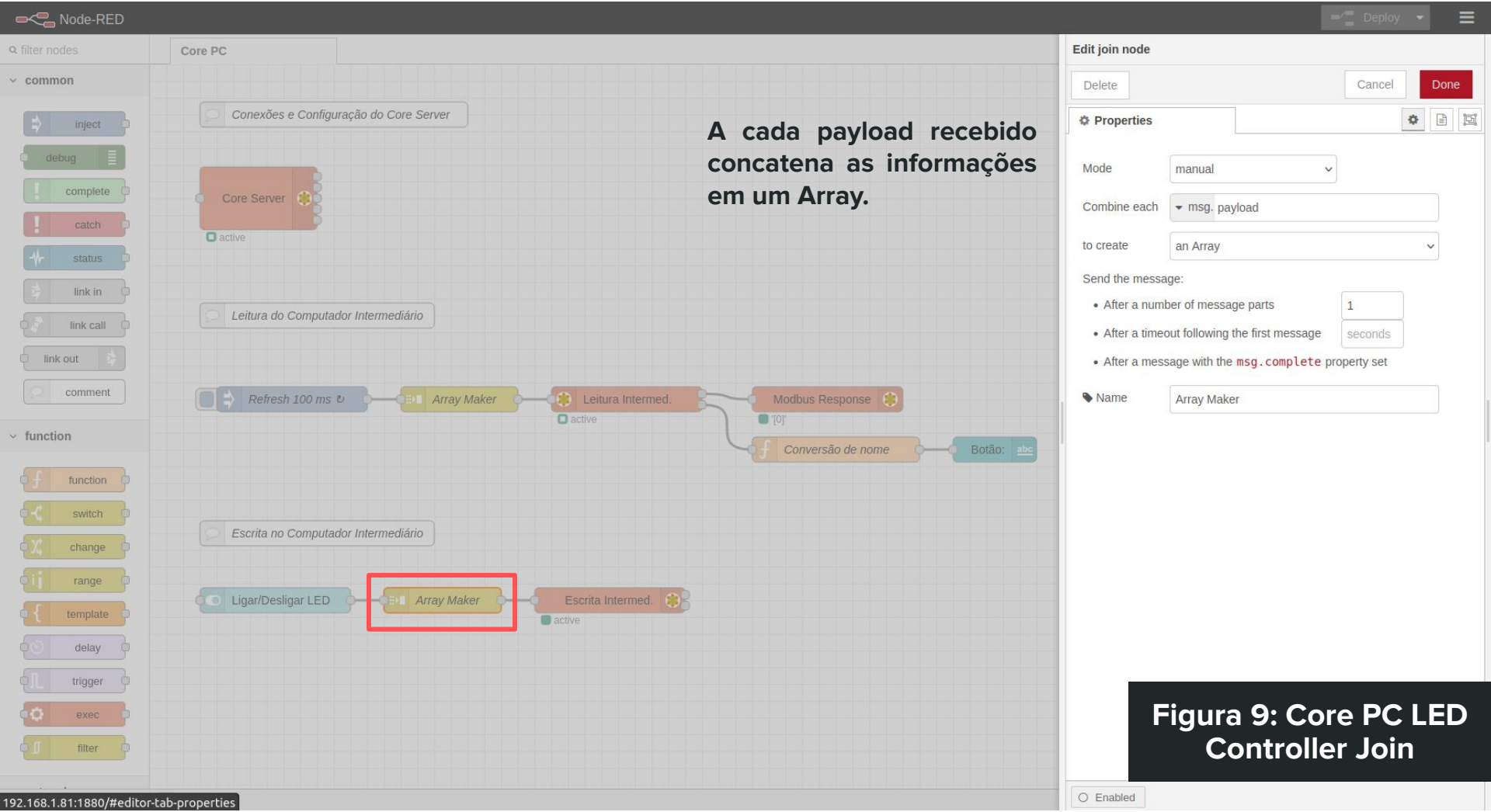



**Figura 9: Core PC LED Controller Join**

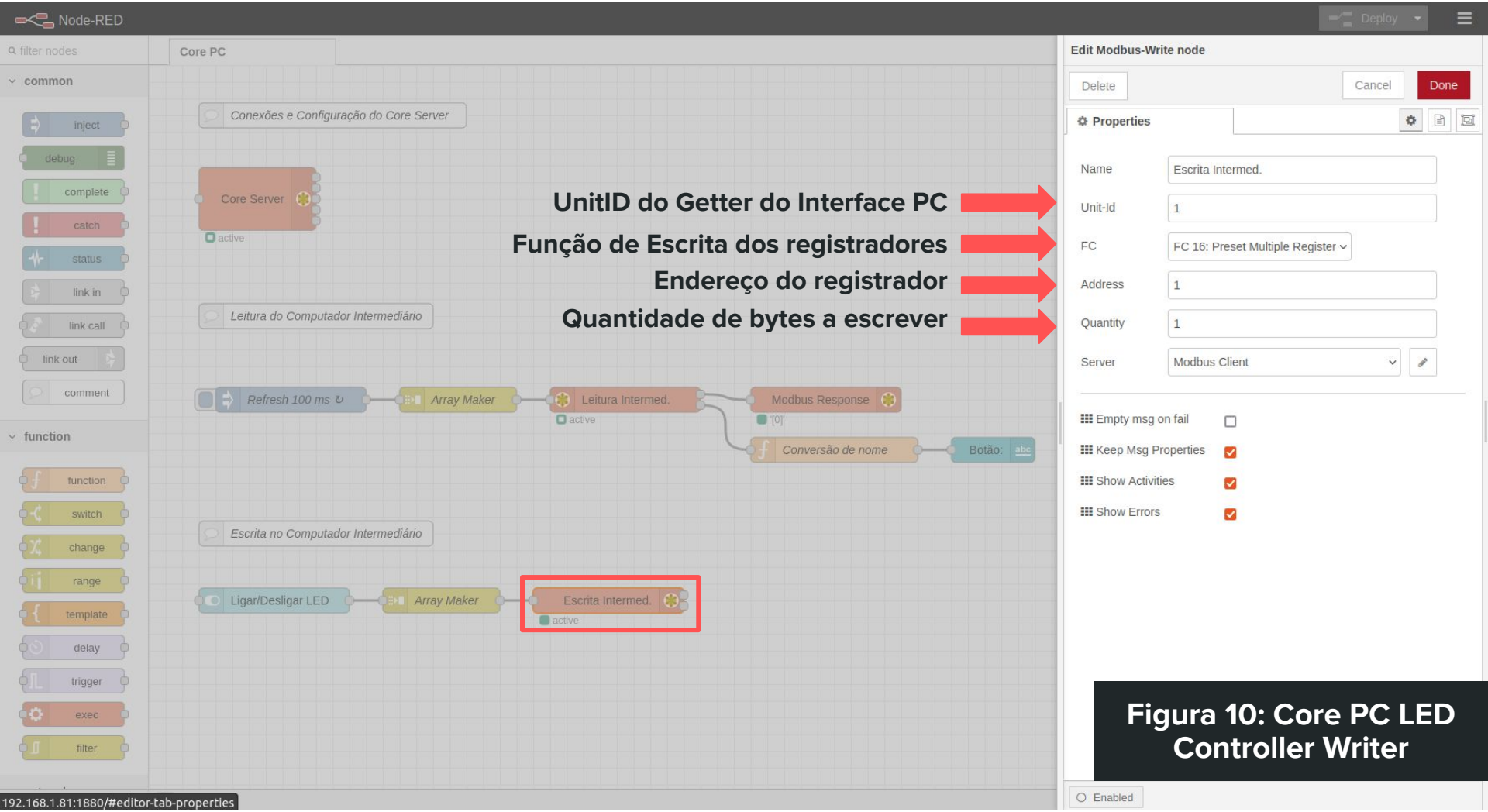



**Figura 10: Core PC LED Controller Writer**

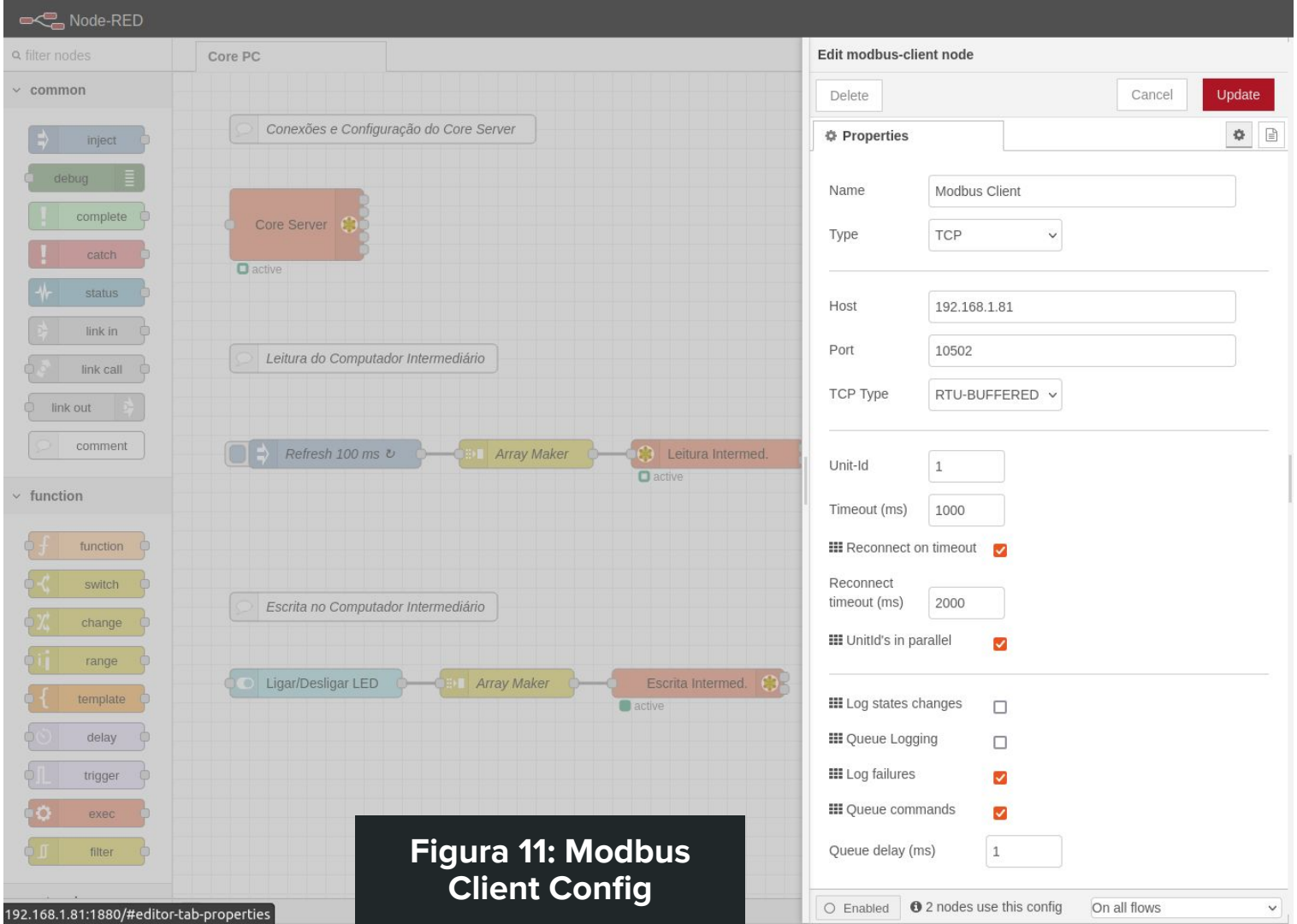



**Figura 11: Modbus Client Config**


**Figura 12: Core PC Dashboard**

# Interface PC

O Interface PC é um PC configurado para simular o comportamento de um CLP, servindo apenas para interfaceamento e transmissão de informações por entre as extremidades da rede configurada (Core PC e Arduino).

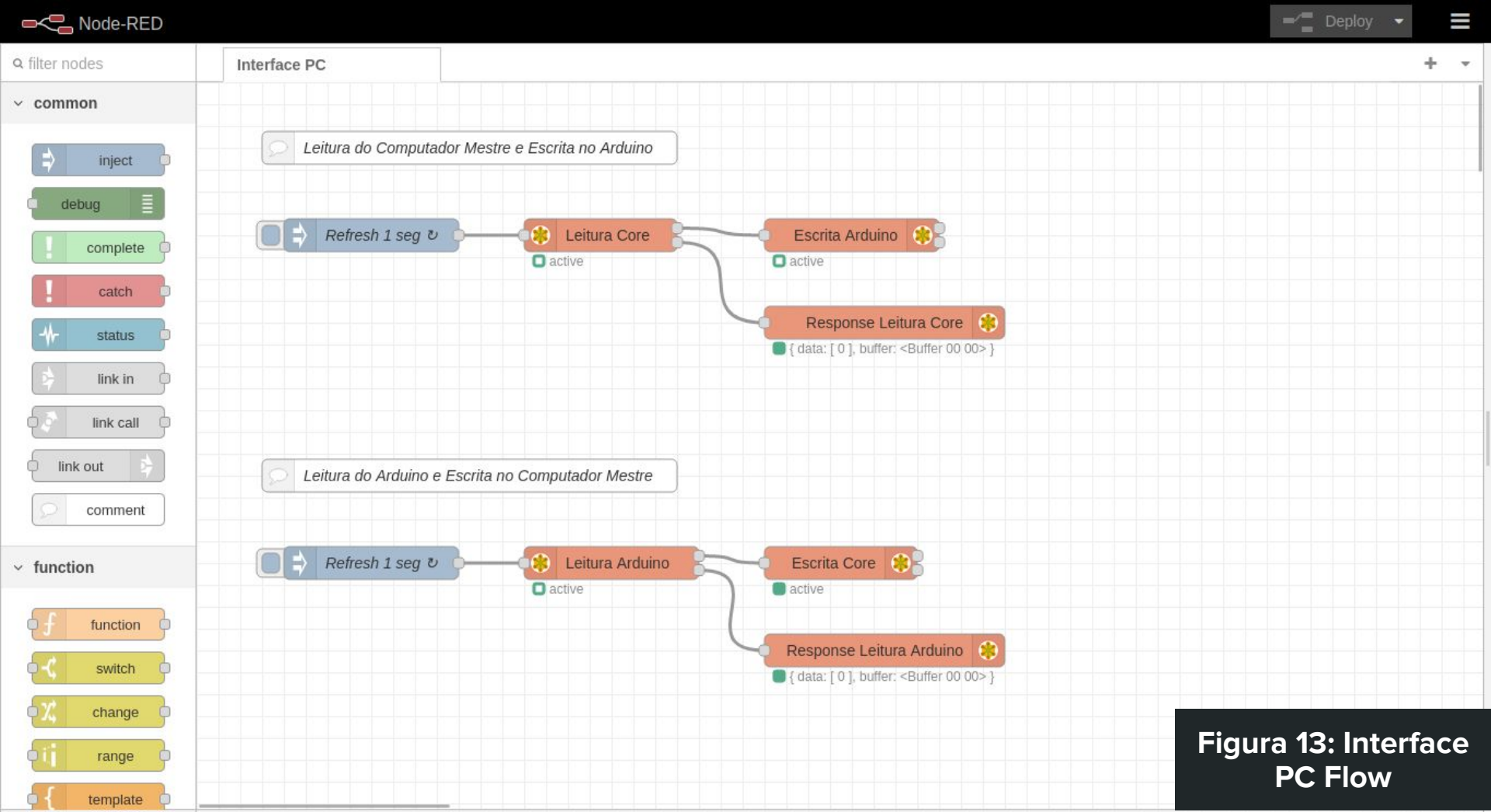



**Figura 13: Interface PC Flow**

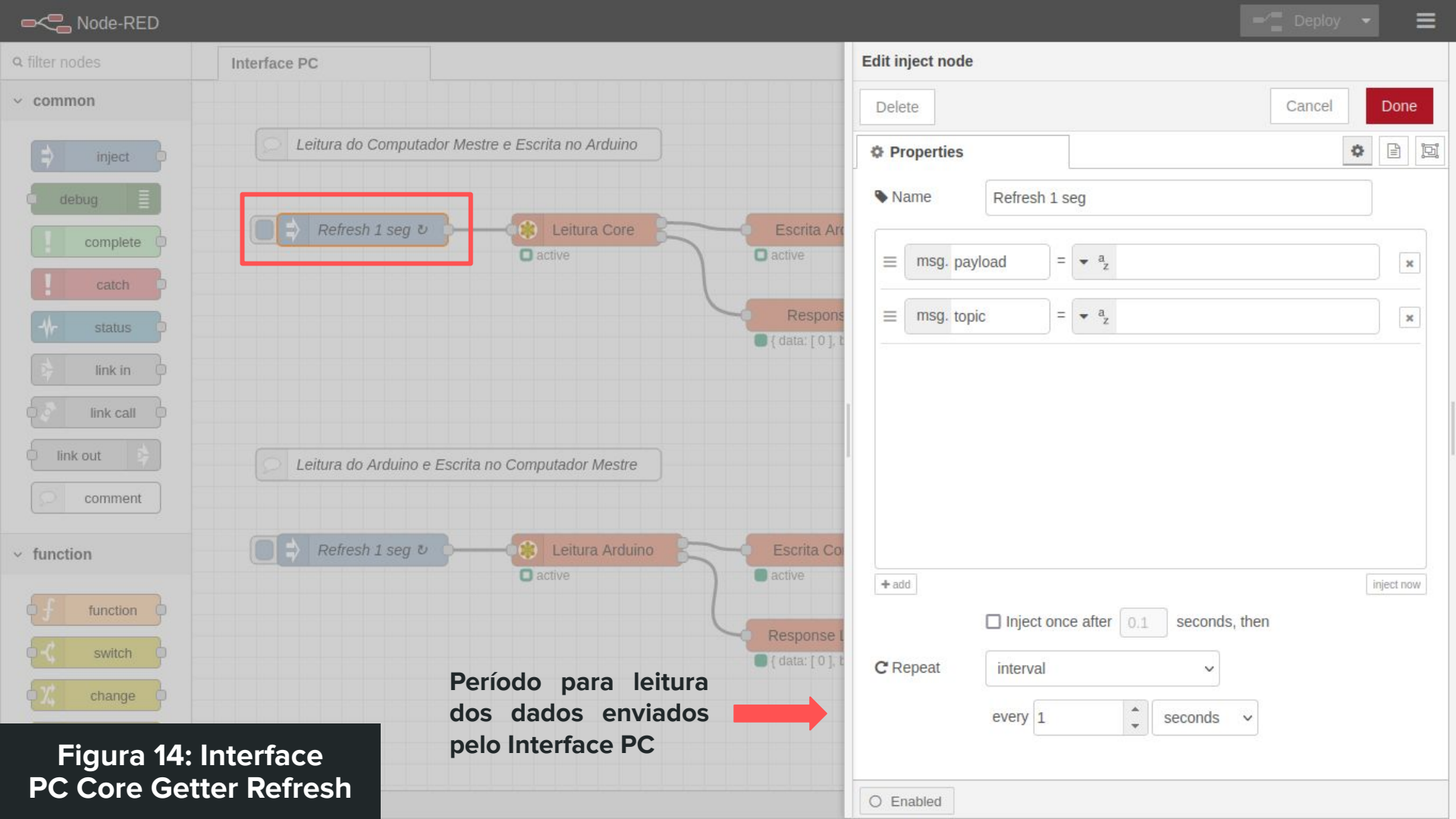



**Figura 14: Interface PC Core Getter Refresh**

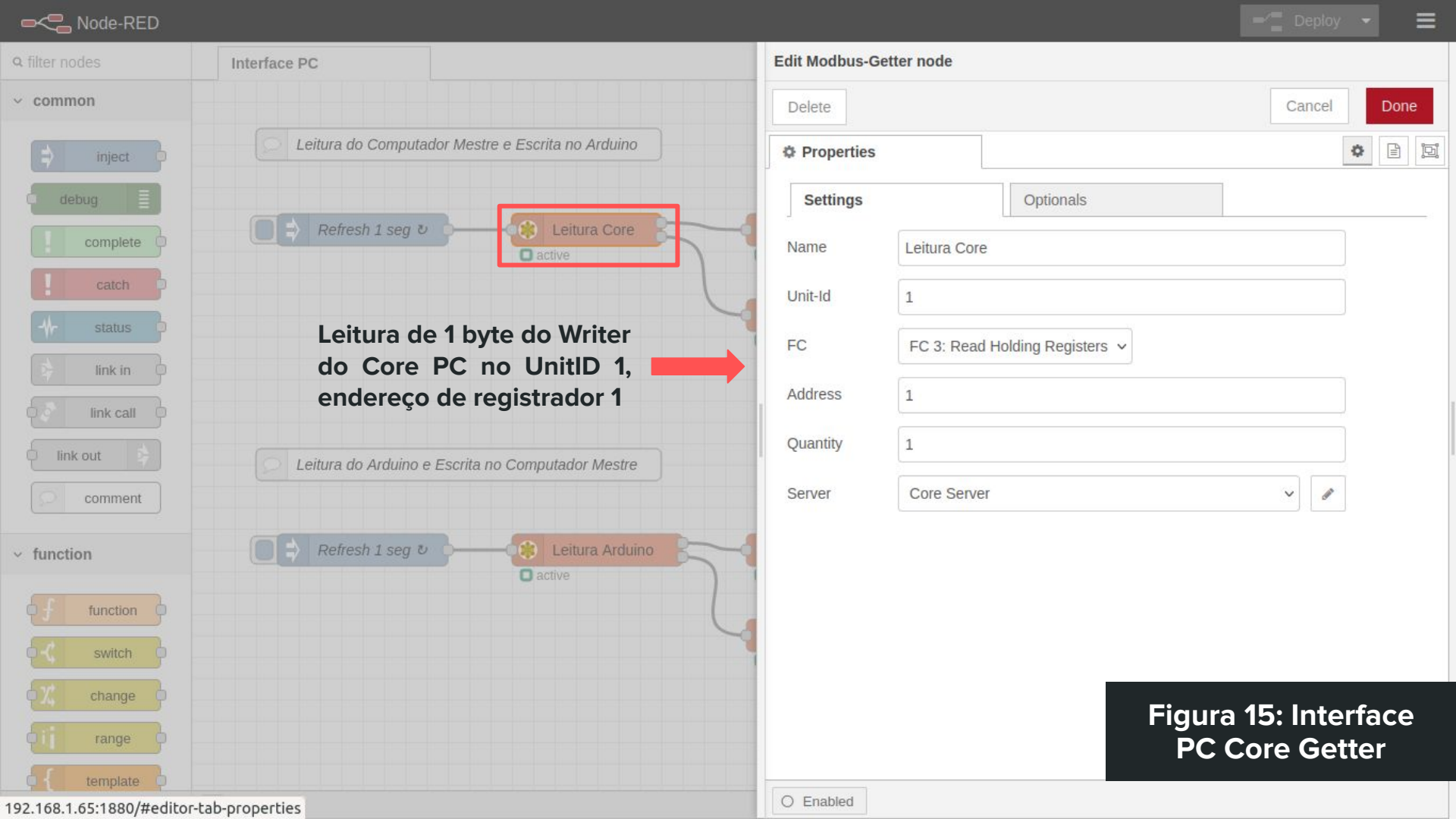



**Figura 15: Interface PC Core Getter**

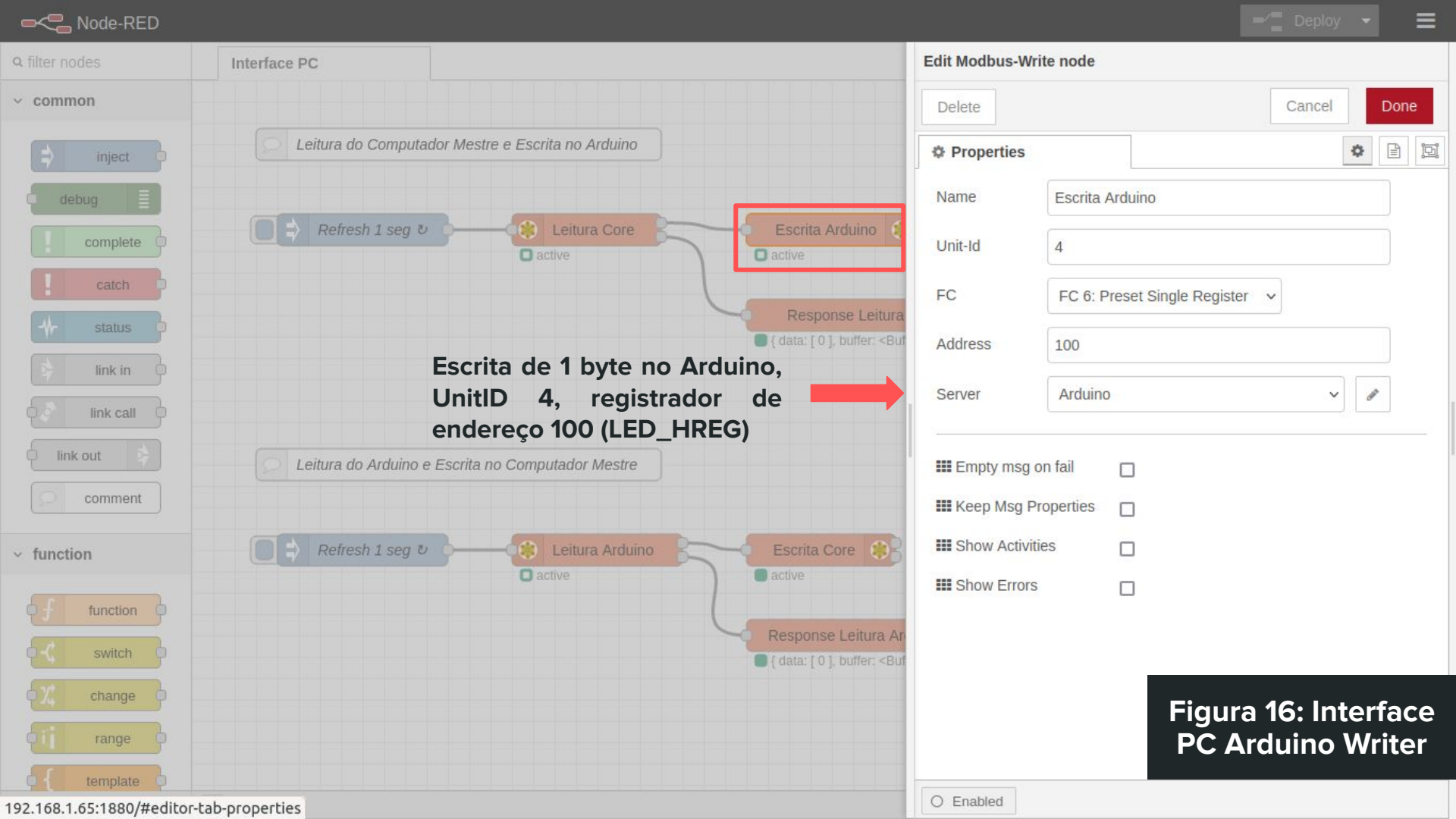



**Figura 16: Interface PC Arduino Writer**

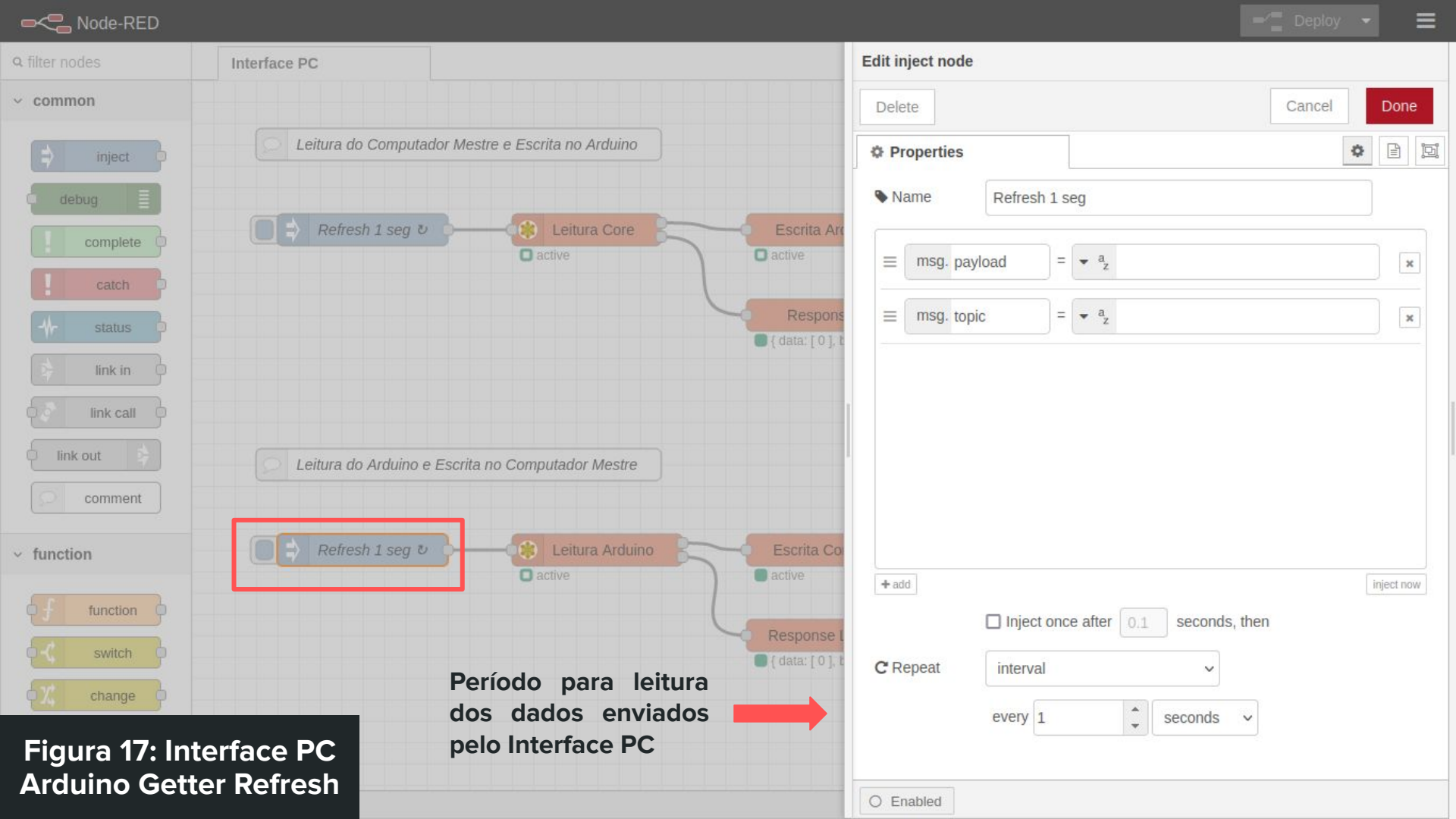



**Figura 17: Interface PC Arduino Getter Refresh**

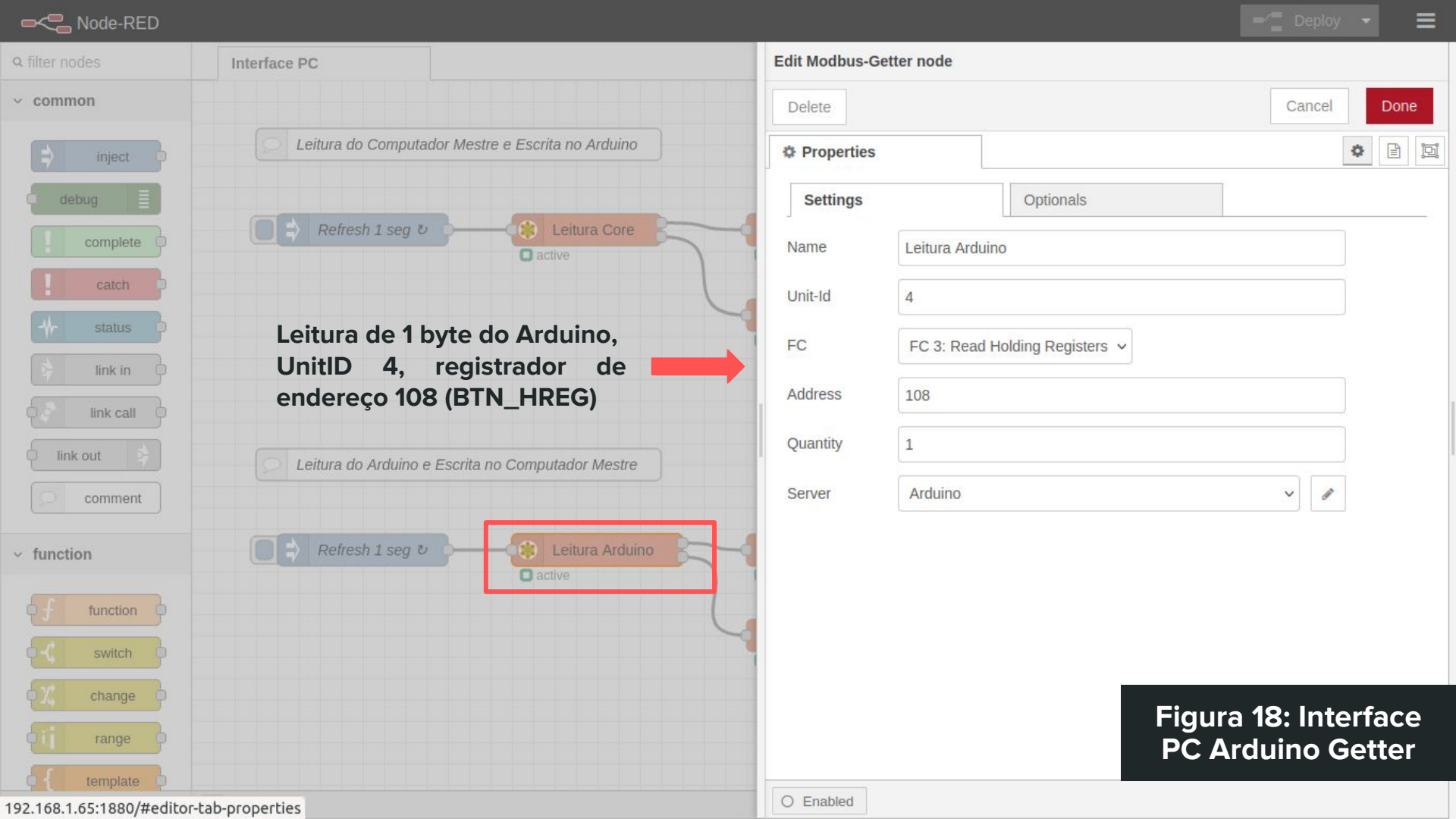



**Figura 18: Interface PC Arduino Getter**

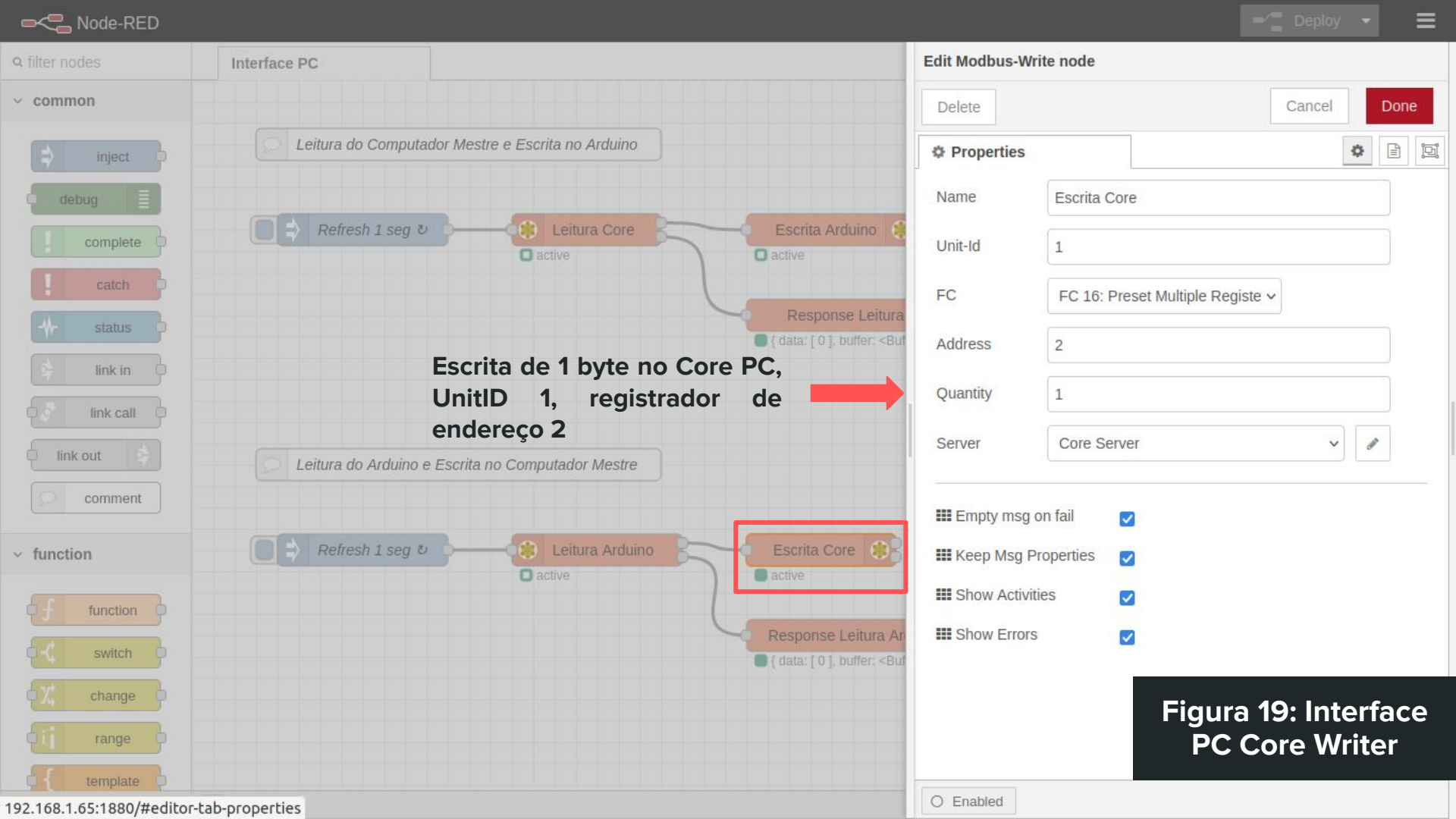



**Figura 19: Interface PC Core Writer**

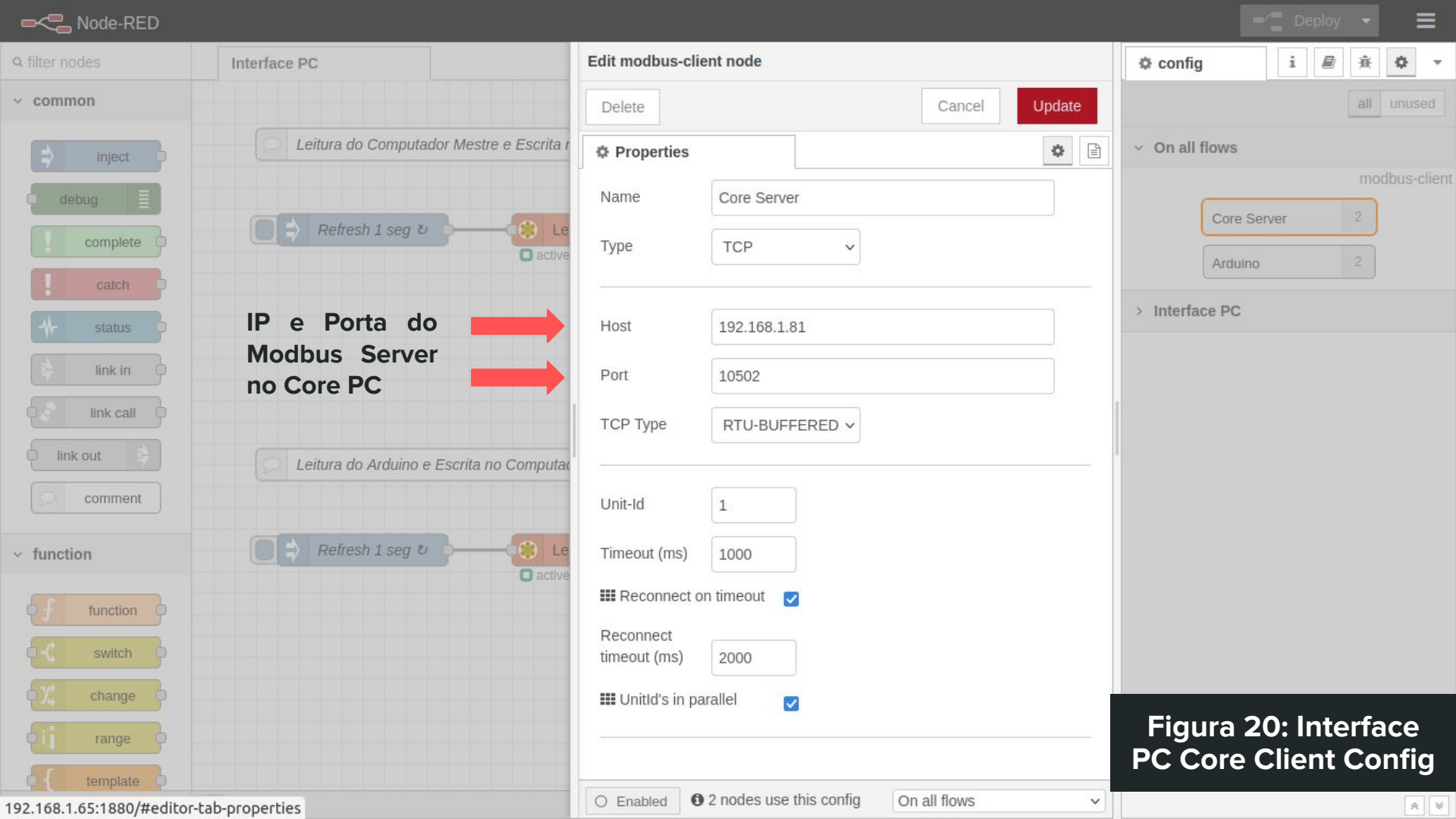



**Figura 20: Interface PC Core Client Config**

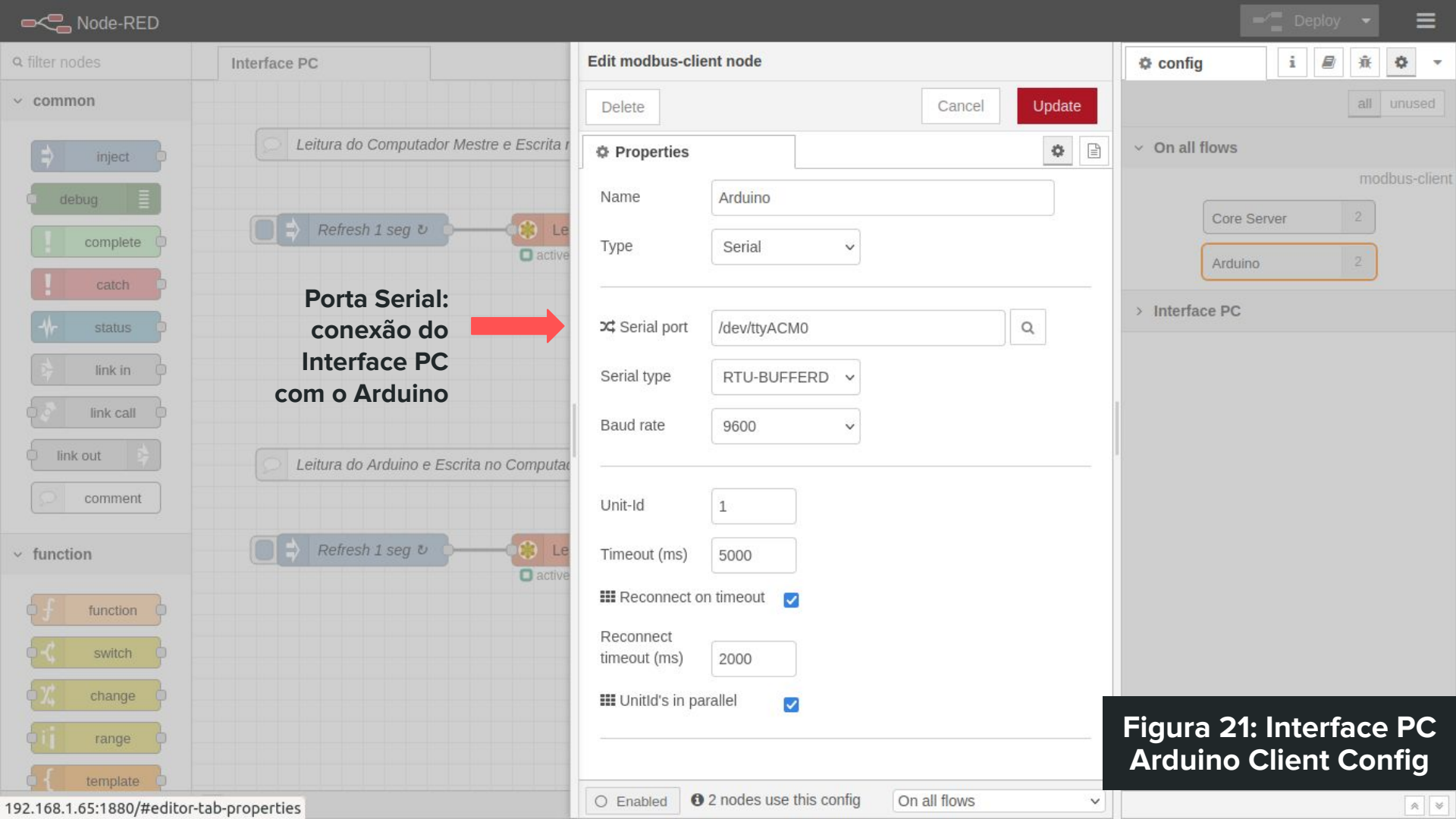



**Figura 21: Interface PC Arduino Client Config**

# Arduino

O Arduino é a extremidade da rede que se conecta diretamente com o circuito. A nível industrial, seria equivalente ao dispositivo responsável por ler informações de sensores e mandar os comandos para os atuadores.

```
// Definição de constantes
dos pinos utilizados
#define LED 13
#define BTN 8

#include <Modbus.h>
#include <ModbusSerial.h>

// Offset de registradores
const int LED_HREG = 100;
const int BTN_HREG = 108;

// Objeto ModbusSerial
ModbusSerial mb;
```

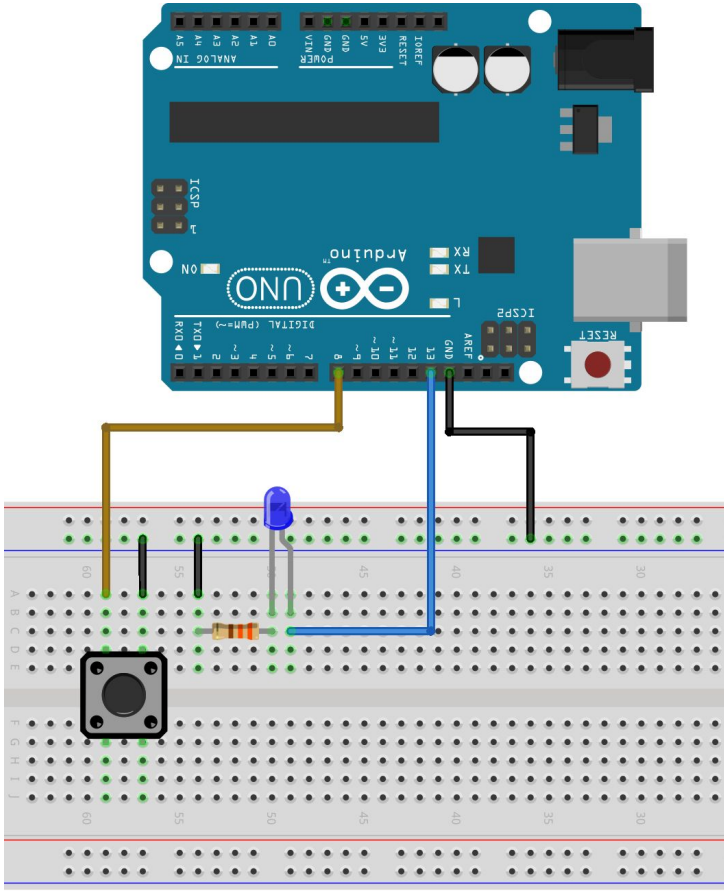
```
void setup() {
  // Configuração da comunicação Modbus
  mb.config(&Serial, 9600, SERIAL_8N1);

  // Configuração de ID
  mb.setSlaveId(4);

  // Definição de pinos
  pinMode(BTN, INPUT_PULLUP);
  pinMode(LED, OUTPUT);

  // Configuração de registradores
  mb.addHreg(LED_HREG, 0);
  mb.addHreg(BTN_HREG, 0);
}
```

```
void loop() {
  // Chamada da task Modbus
  mb.task();

  // Rotina de atualização do LED
  if(mb.Hreg(LED_HREG) == 1){
    digitalWrite(LED, true);
  }else if(mb.Hreg(LED_HREG) == 0){
    digitalWrite(LED, false);
  }

  // Rotina de leitura do botão
  mb.Hreg(BTN_HREG, !digitalRead(BTN));
}
```

# Código do Arduino

# Circuito

O circuito representa os sensores que podem ser lidos e atuadores que podem ser controlados. Na nossa montagem, o botão simula um sensor, e o LED, um atuador.



# Conexões com o Arduino

- Pino 8: Botão, INPUT_PULLUP
- Pino 13: LED, OUTPUT

# Vídeo - Demonstração do funcionamento

# Considerações Finais

- Conseguimos fazer uma prova de conceito demonstrando didaticamente algumas aplicações de Redes Industriais.

- Apesar de em indústrias reais a rede geralmente ser de uma escala bem maior, conseguimos reproduzir conceitos essenciais para uma rede industrial, como uma interface de supervisão e controle, dispositivos mediadores, controladores, sensores e atuadores.

# Obrigado pela atenção!