

Arquitetura HIL para teste de sistemas embarcados como *vehicle interface* de veículos autônomos baseados no Autoware

Projeto – Etapa 3

Gabriel Toffanetto França da Rocha

g289320@dac.unicamp.br

Professor Dr. Rodrigo Moreira Bacurau

IM420X – Projeto de Sistemas Embarcados de Tempo Real

Faculdade de Engenharia Mecânica
Universidade Estadual de Campinas

12 de novembro de 2024



Agenda

1 Introdução

2 Atualizações no sistema embarcado

3 Testes dos módulos

4 Problemas encontrados

5 Cronograma



Introdução



Proposta

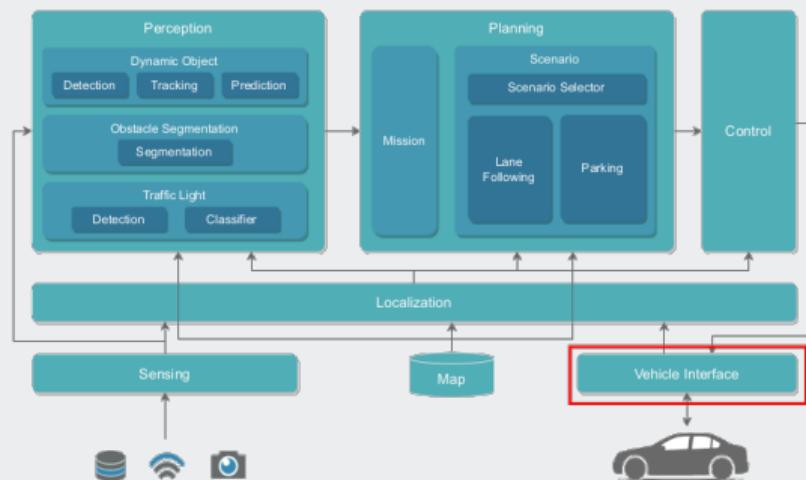


Figura 1: Escopo do projeto na arquitetura Autoware.

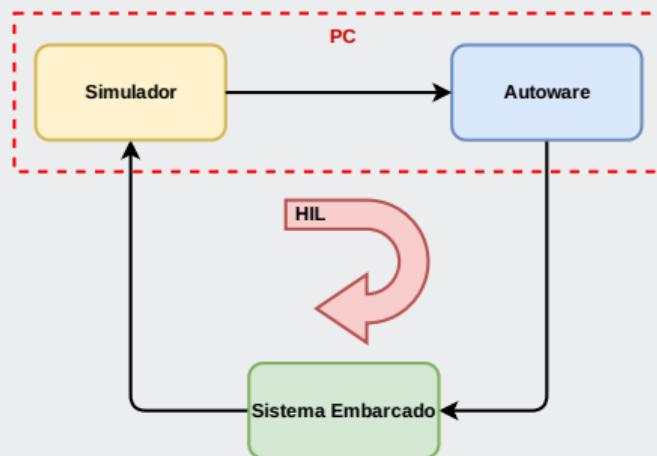


Figura 2: Arquitetura de teste do *hardware*.

Sistema embarcado



Diagrama de blocos

- Sobrecarga do micro-ros para controle e tráfego dos dados do simulador;
- Aproximação que leva à *overhead* comparado com a arquitetura real.

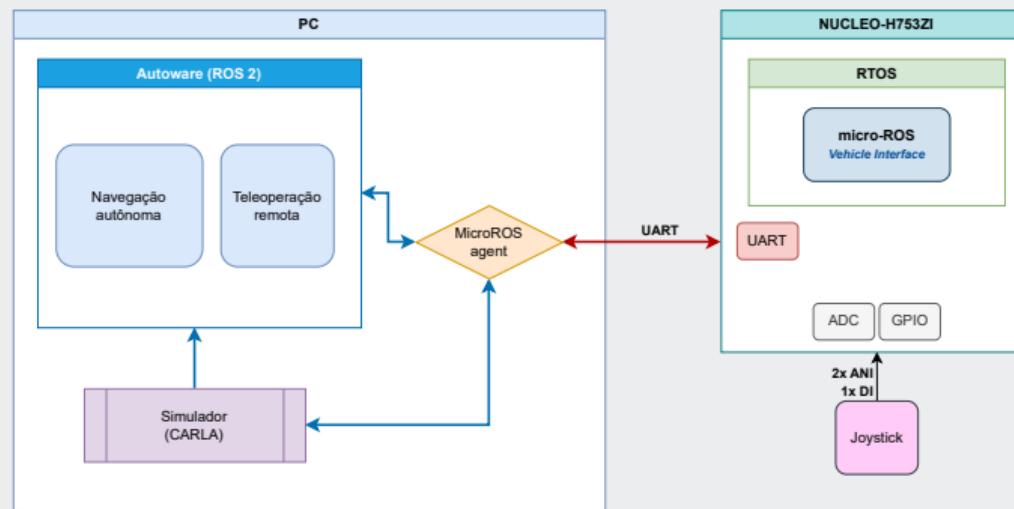


Figura 3: Diagrama de blocos atualizado.

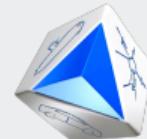


Diagrama de blocos

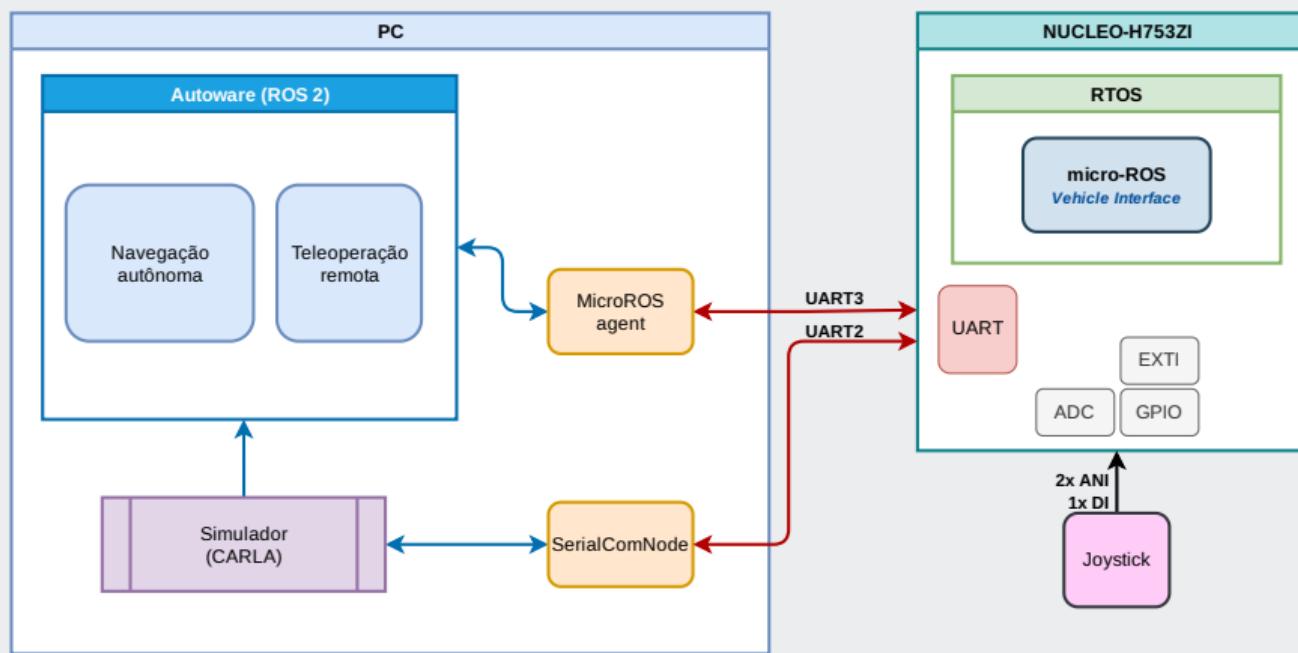


Figura 4: Diagrama de blocos atualizado.

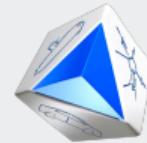


Diagrama de tarefas

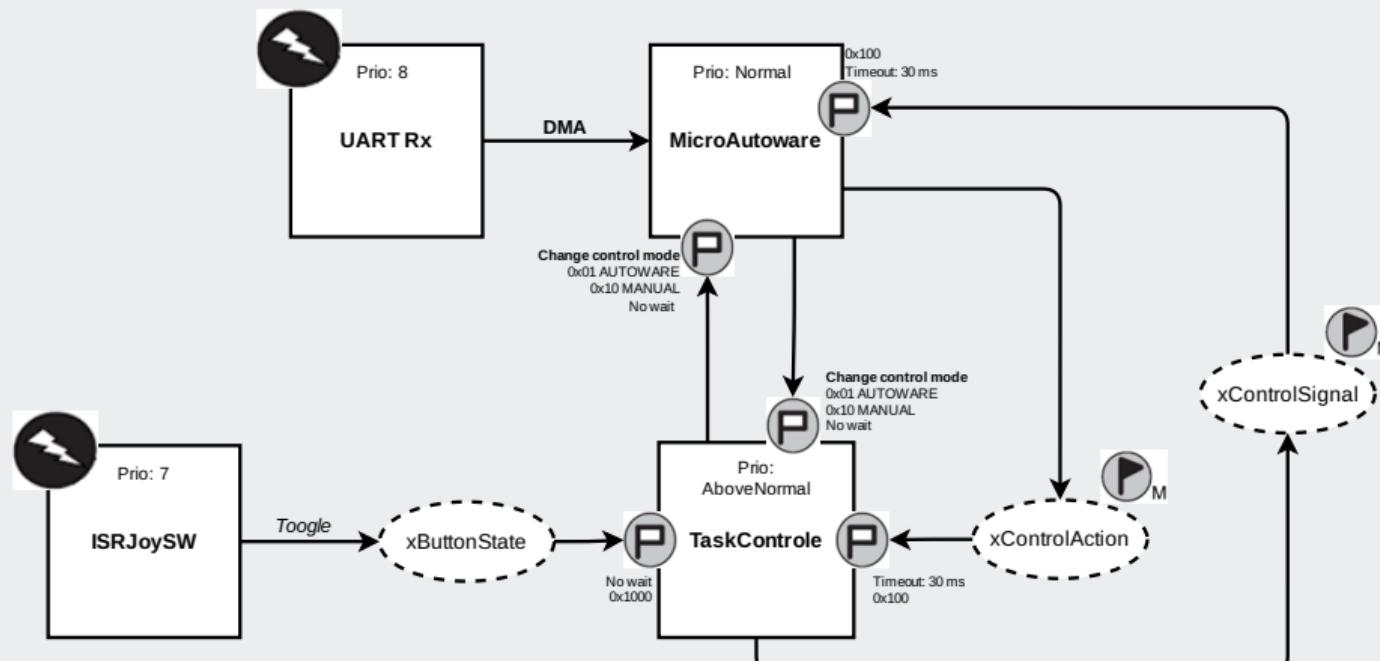


Figura 5: Diagrama de tarefas antigo.

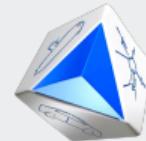


Diagrama de tarefas

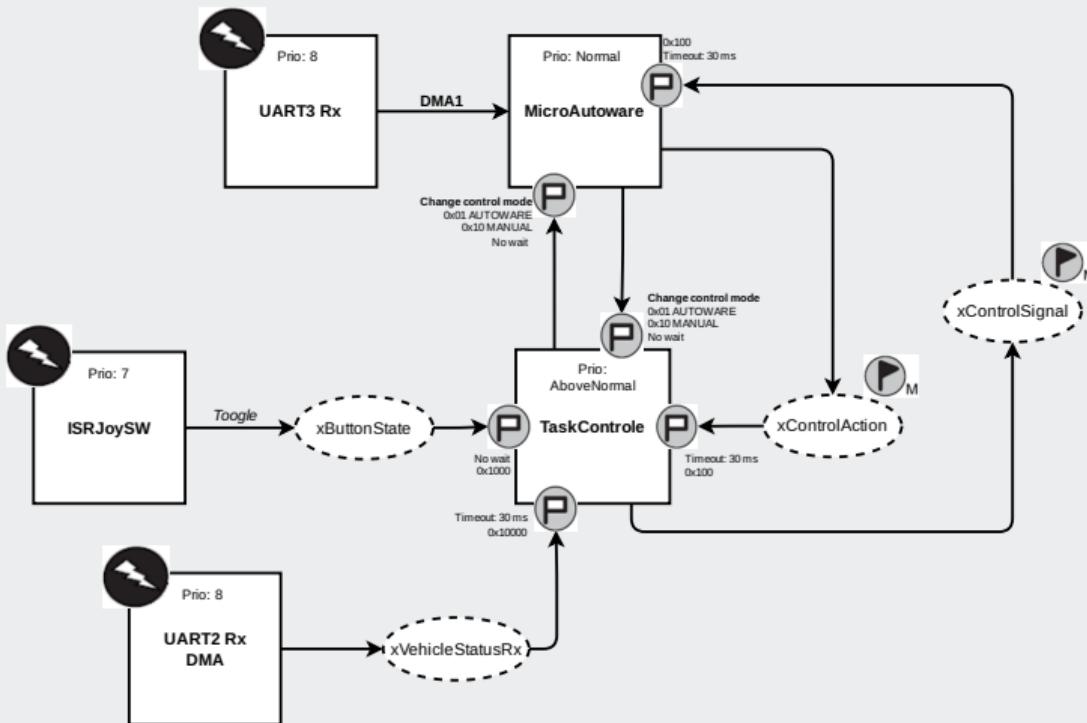
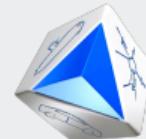


Figura 6: Diagrama de tarefas atualizado.



Comunicação com o simulador

Sistema embarcado → CARLA

- float fSteeringAngle;
- float fSteeringVelocity;
- float fSpeed;
- float fAcceleration;
- float fJerk;
- unsigned char ucControlMode;

CARLA → Sistema embarcado

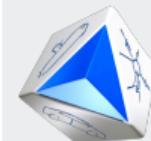
- float fLongSpeed;
- float fLatSpeed;
- float fHeadingRate;
- float fSteeringStatus;

Padrão de Mensagem: Sistema embarcado → CARLA

- #S%c%c%c%cW%c%c%c%cV%c%c%c%cA%c%c%c%cJ%c%c%c%cM%c\$
- 30 bytes

Padrão de Mensagem: CARLA → Sistema embarcado

- #A%c%c%c%cB%c%c%c%cC%c%c%c%cD%c%c%c%c\$
- 22 bytes



Máquinas de estados de comunicação

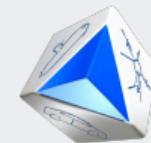
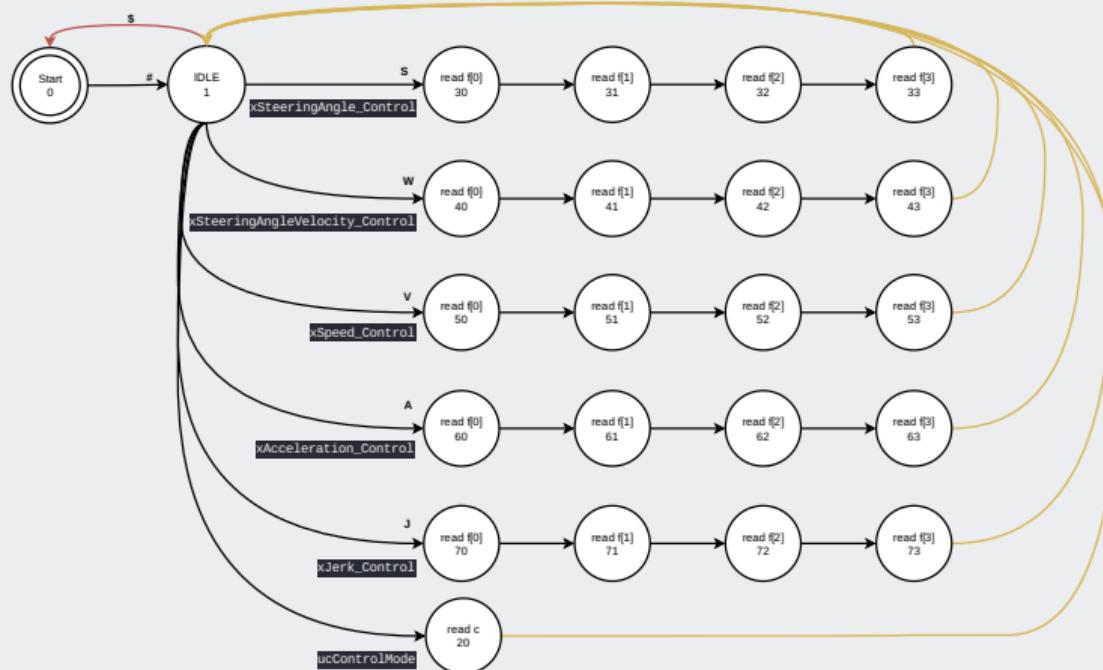


Figura 7: Maquina de estados da comunicação sistema embarcado → CARLA.

Máquinas de estados de comunicação

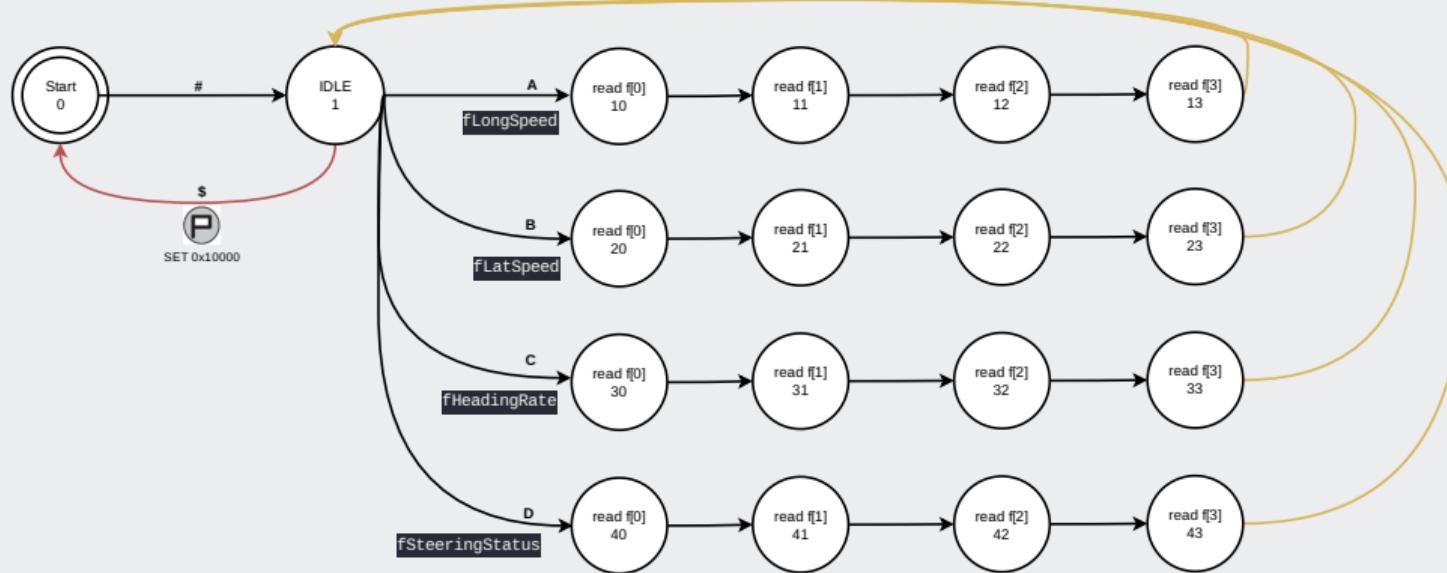
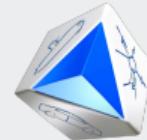
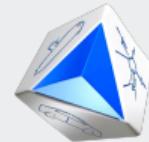


Figura 8: Maquina de estados da comunicação CARLA → sistema embarcado.



Testes dos módulos



Montagem do *hardware*

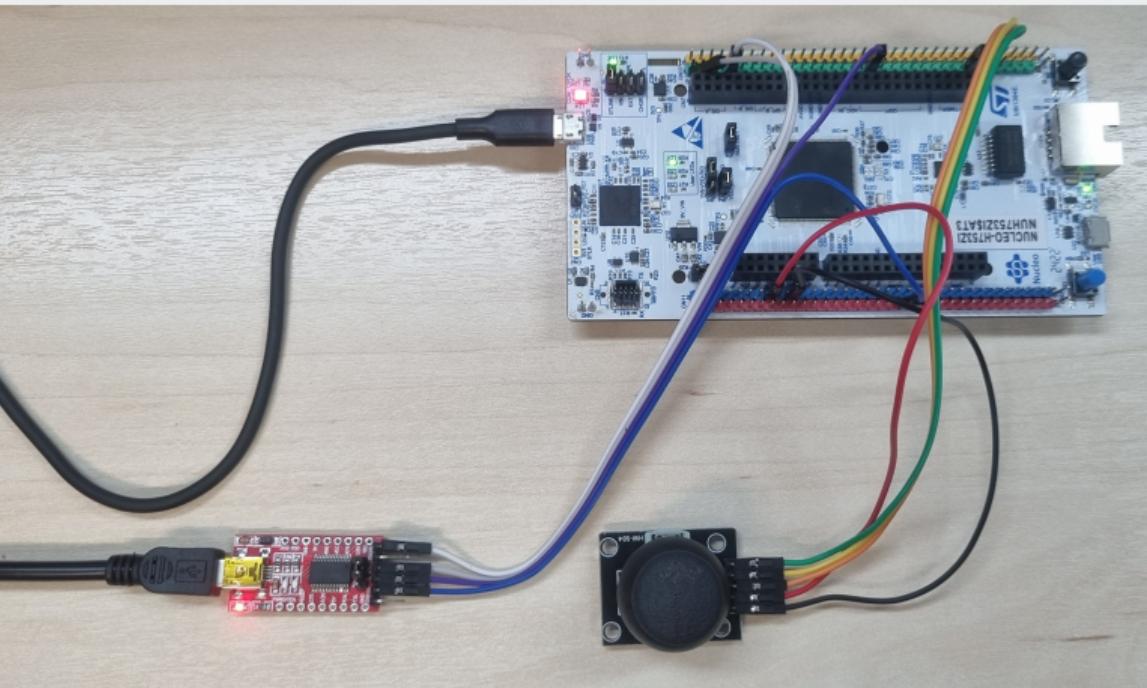
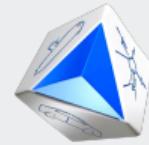
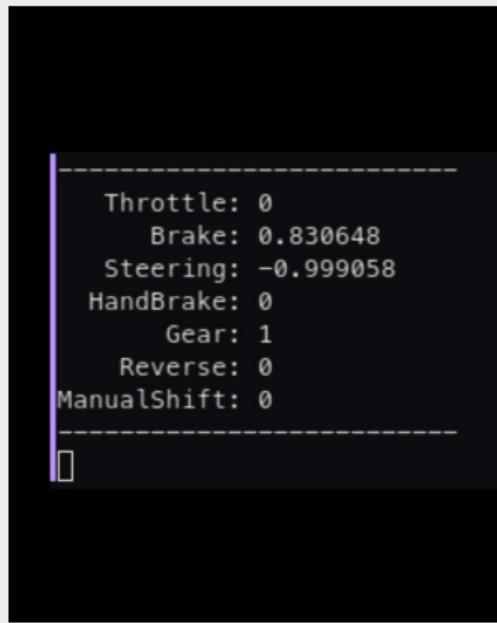


Figura 9: Montagem física dos componentes utilizados.



Leitura do Joystick + Comunicação serial com o CarlaSerialBridge



```
-----  
    Throttle: 0  
    Brake: 0.830648  
    Steering: -0.999058  
    HandBrake: 0  
    Gear: 1  
    Reverse: 0  
    ManualShift: 0  
-----
```

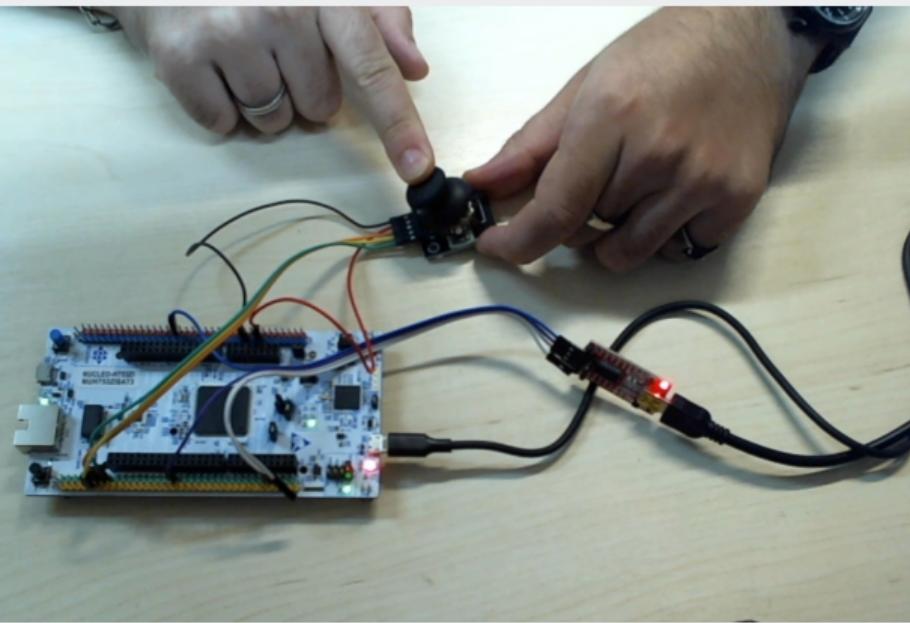
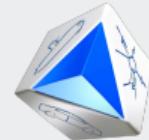


Figura 10: Leitura do Joystick pelo *node* CarlaSerialBridge.



Interrupção JoySW

```
Brake: 0
Steering: -0.000577279
HandBrake: 0
Gear: 1
Reverse: 0
ManualShift: 0
-----
4 Error reading from serial p
4 Error reading from serial p
[]
```

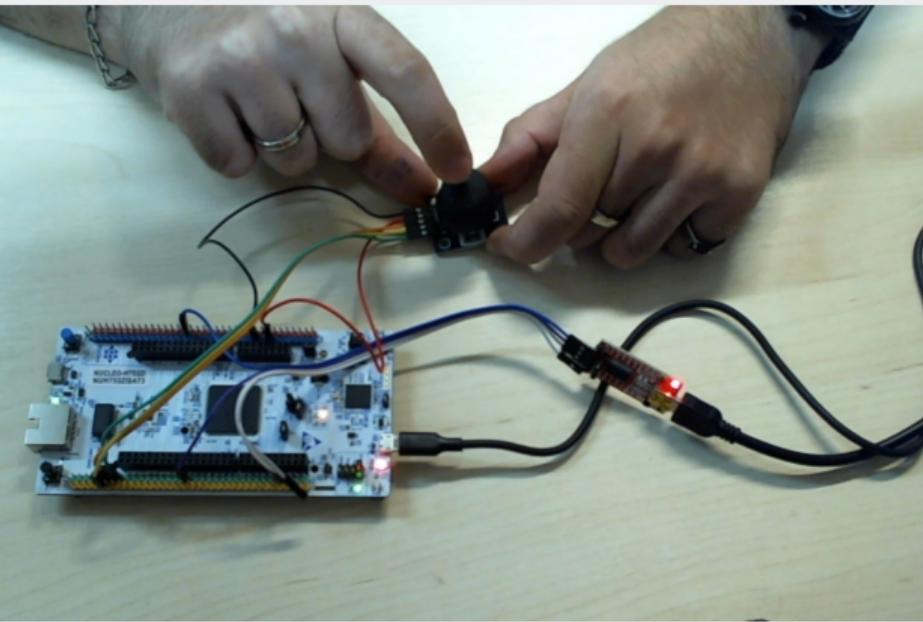
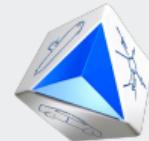


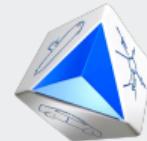
Figura 11: Troca de modo de condução pela interrupção EXTI JoySW.



Modo de operação manual



Figura 12: Validação da comunicação serial com o simulador e controle manual.



micro-ROS Vehicle Interface

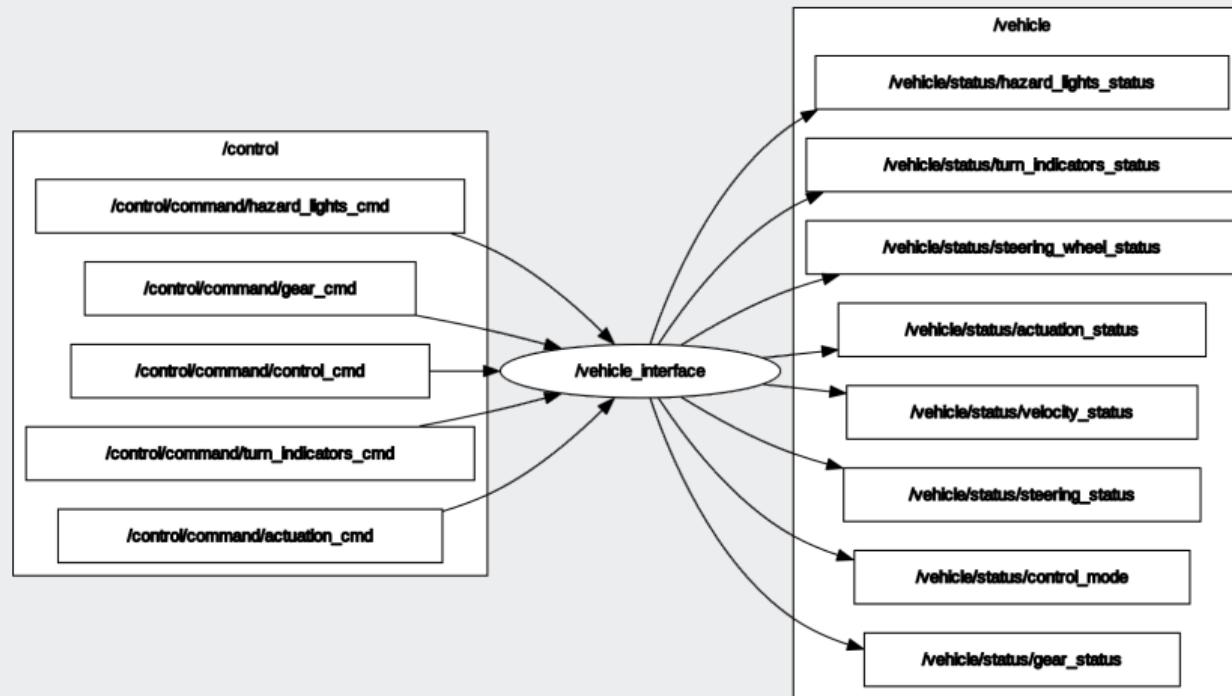


Figura 13: Diagrama de *nodes* e *topics* para a *vehicle interface* construída.



ROS

Figura 14: Diagrama de *nodes* e *topics* da arquitetura HIL.

ROS: micro-ROS Vehicle Interface

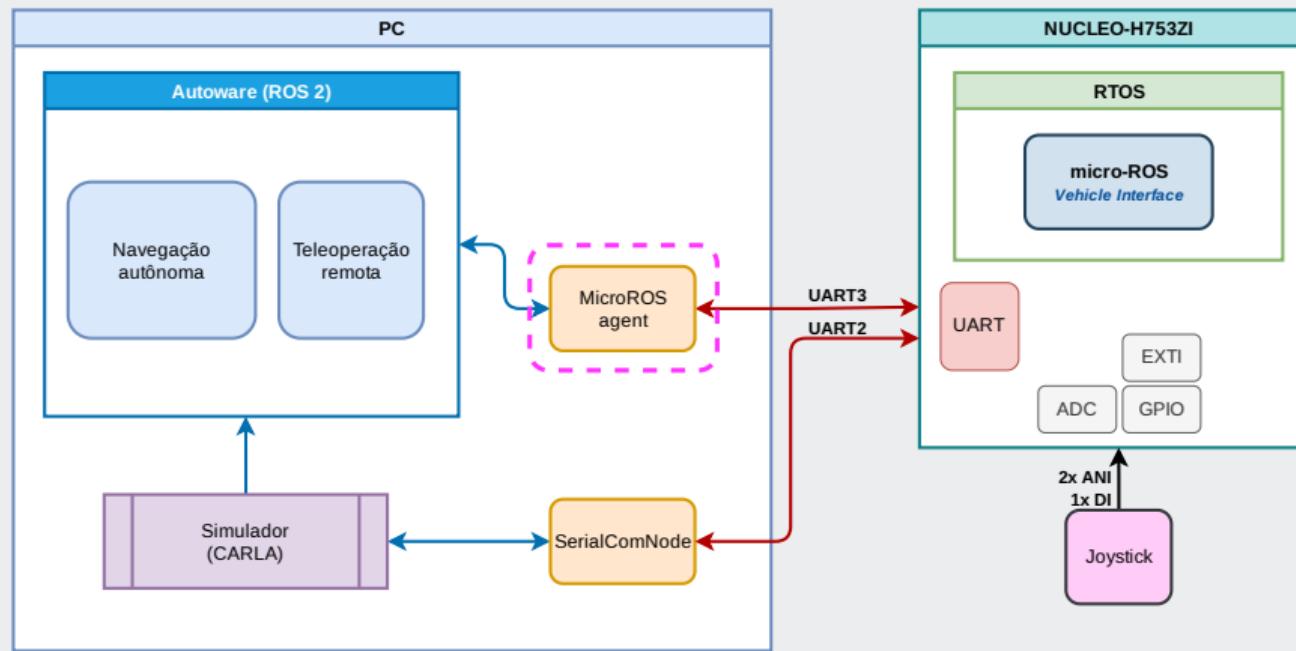
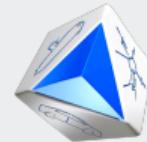


Figura 15: Diagrama de blocos – micro-ros.



ROS: micro-ROS Vehicle Interface

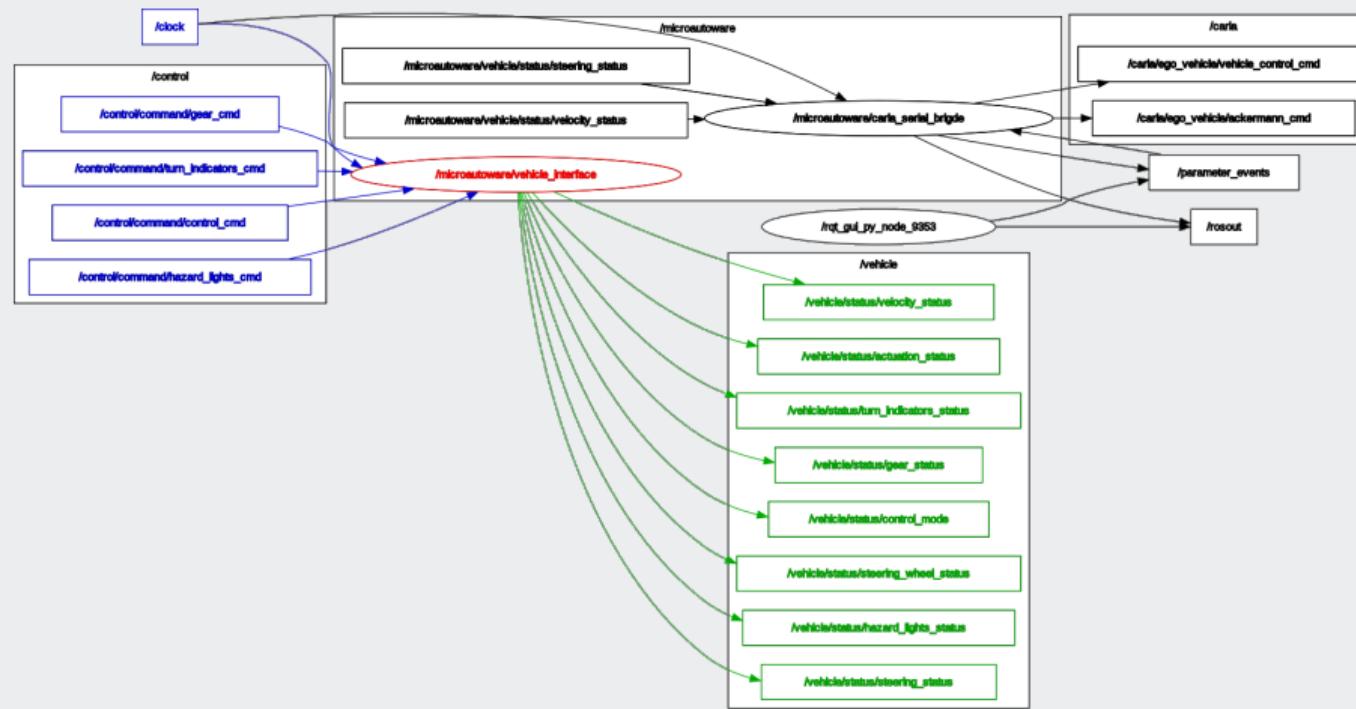
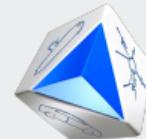


Figura 16: Diagrama de *nodes* e *topics* da arquitetura HIL (*vehicle interface*).



ROS: CARLA-Serial-Bridge

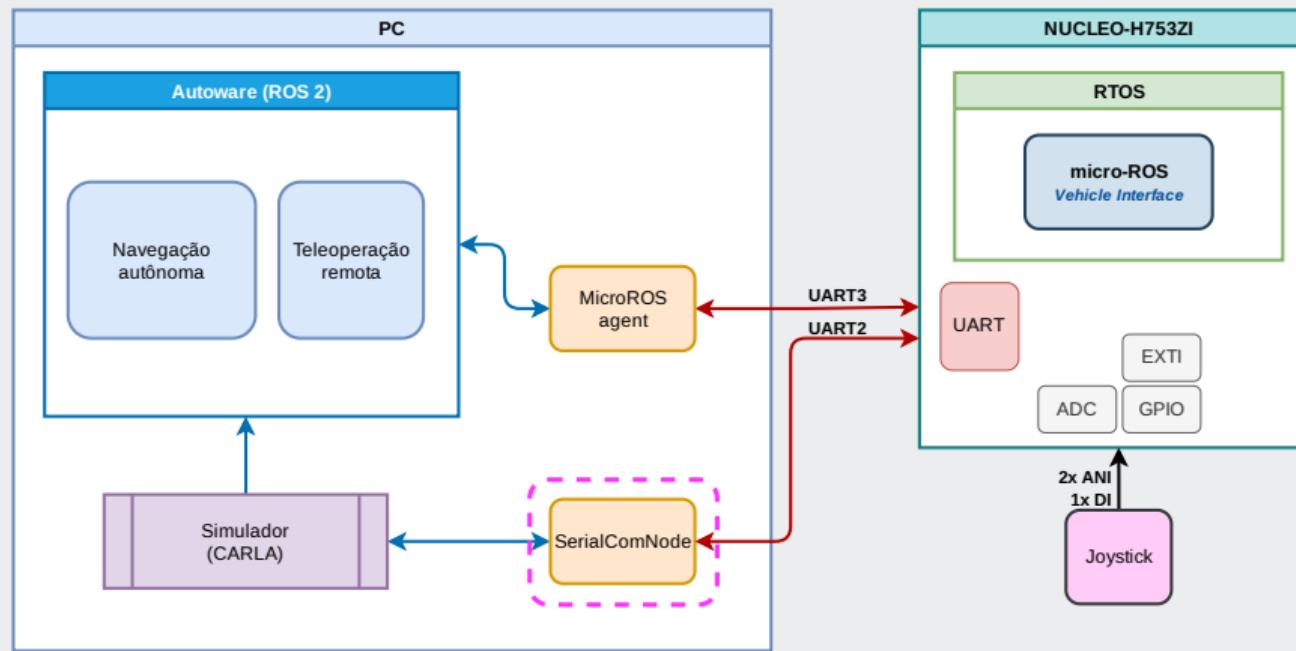
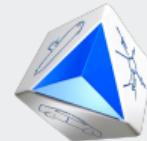


Figura 17: Diagrama de blocos – CARLA-Serial-Bridge.



ROS: CARLA-Serial-Bridge

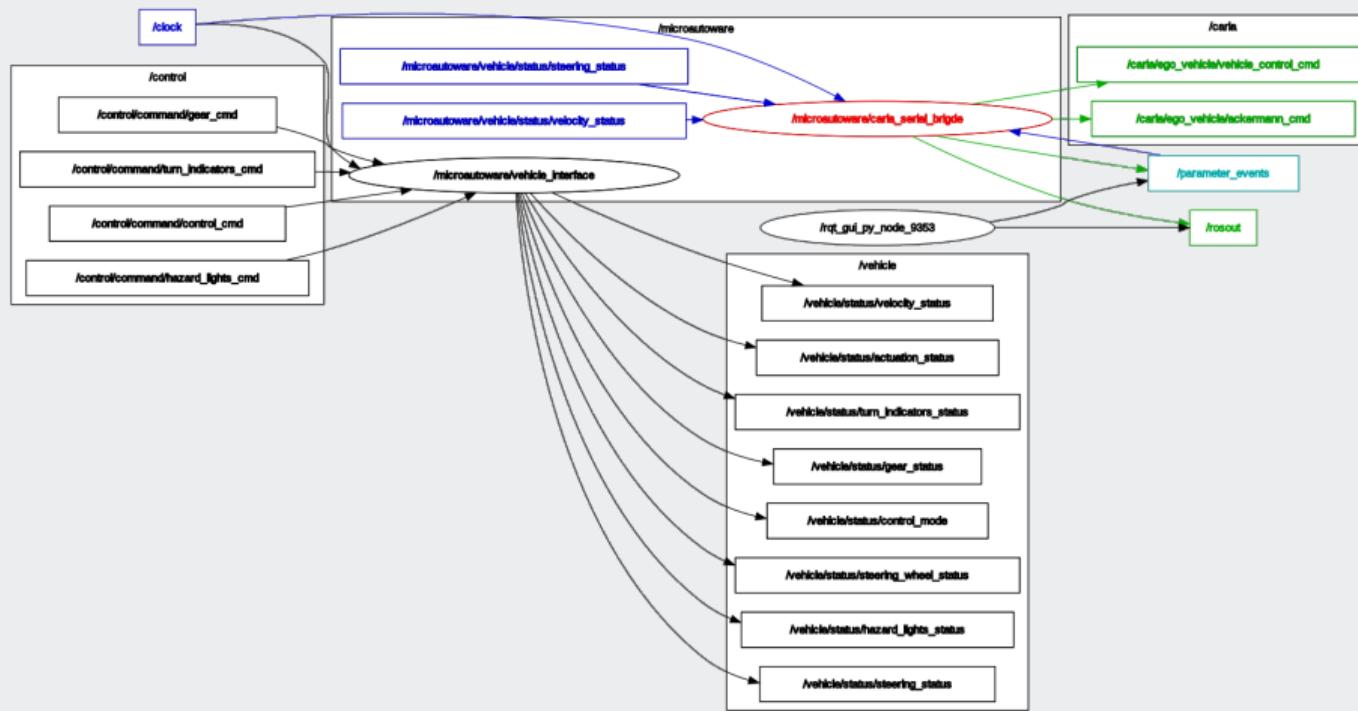
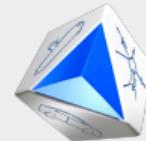


Figura 18: Diagrama de *nodes* e *topics* da arquitetura HIL (ponte serial com o CARLA).



Teste do fluxo de dados ROS × micro-ROS

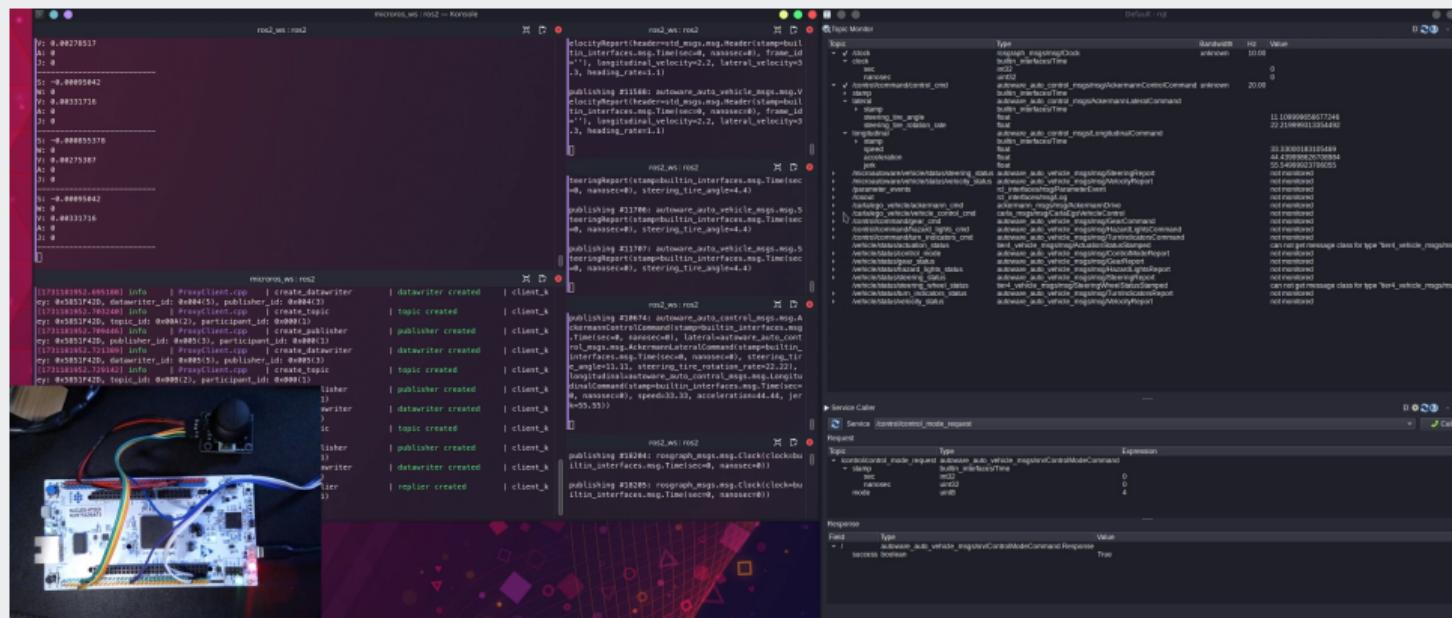
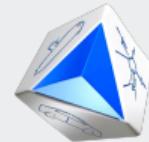


Figura 19: Validação da comunicação dos tópicos e serviços no micro-ROS.



Problemas encontrados



Problemas encontrados

Problemas diagnósticados

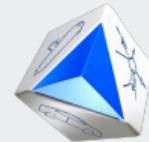
- Perda de ThreadFlags;
 - Leitura de *flags* com `osThreadFlagGet()` antes de `osThreadFlagWait()`.
- *Bounce* no botão JoySW.
 - *Debounce* em *software* por meio de bloqueio por 1000 ticks.

Problemas à serem verificados

- Escolha dos *timeouts*;
- Escolha das taxas de envio de dados;
 - CARLA ↔ microAutoware ↔ Autoware.
- Política de entrada e saída do modo de emergência.



Cronograma

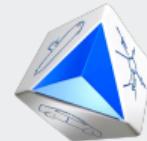


Cronograma

Atividade/Semana	1	2	3	4	5	6	7	8	9
Proposta do projeto		■							
Projeto de <i>hardware e software</i>			■						
Integração do STM com o micro-ROS			■						
Integração do micro-ROS com o Autoware				■	■				
Implementação das tarefas do sistema embarcado					■	■	■		
Construção do ambiente de testes					■	■	■		
Realização dos testes						■	■	■	
Escrita do relatório		■	■	■	■	■	■	■	

Tabela 1: Cronograma de atividades.

- Semana 2: Apresentação Etapa 1
- Semana 4: Apresentação Etapa 2
- Semana 7: Apresentação Etapa 3
- Semana 9: Apresentação Final



Obrigado!

Dúvidas?

