

Servo Hub

- Visão Geral do Servo Hub
 - Fiação de Servo
 - Visão Geral do Servo Hub
 - Canal de Energia
 - Pulso de canal
 - Configurando Servo Hub
 - Padrões de LEDs de status do Servo Hub
 - Servos de Comando
 - Solução de problemas
 - Configuração do Servo Hub com a Control Hub

Visão Geral do Servo Hub

Fiação de Servo

Este documento foi desenvolvido em parceria com a equipe JUSTICE FTC TEAM #21036

Diagramas de fiação

FTC/RS485

Conectando o Servo Hub em uma Control Hub

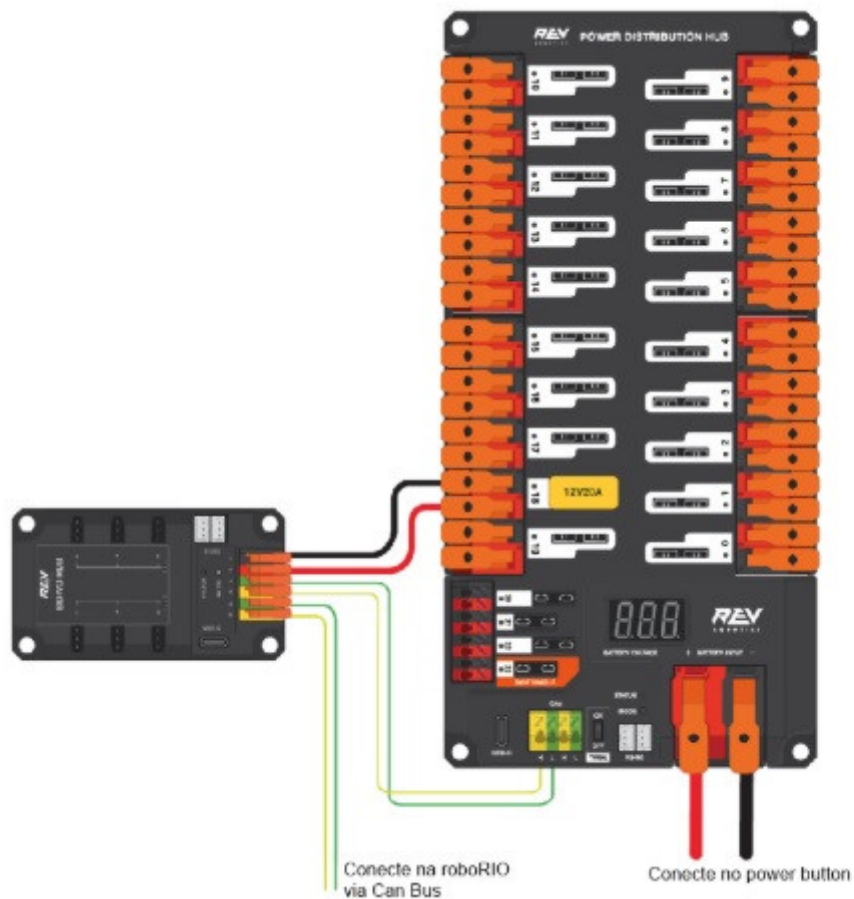
Observe que o pino de aterramento de cada motor servo ficará voltado para a borda externa do Servo Hub e o pino de sinal ficará mais próximo do centro. Os servos conectados às portas 0, 1 e 2 serão espelhados pelos servos conectados às portas 3, 4 e 5.



FRC/CAN

Conectando o Servo Hub para controle via CAN

Observe que o pino de aterramento de cada motor servo ficará voltado para a borda externa do Servo Hub e o pino de sinal ficará mais próximo do centro. Os servos conectados às portas 0, 1 e 2 serão espelhados pelos servos conectados às portas 3, 4 e 5.



Visão Geral do Servo Hub

Este documento foi desenvolvido em parceria com a equipe JUSTICE FTC TEAM #21036

O REV Servo Hub é compatível com ambos os sistemas REV ION e REV DUO. Por meio de uma única interface de comunicação, ele pode fornecer controle avançado de até seis (6) servos. Isso significa que o Servo Hub não precisa de cabeamento PWM adicional entre ele e o controlador do robô, simplificando bastante a fiação.

Futuras atualizações de firmware desbloquearão recursos adicionais, como medições de corrente para cada canal de saída, comunicação CAN, controle de potência de canal individual para permitir estados de servo desligado e até mesmo tensão de saída ajustável para proporcionar adaptabilidade a uma ampla gama de classes de servo.



O Servo Hub é fácil de atualizar e configurar por meio da conexão USB-C, utilizando o REV Hardware Client. Com uma saída de corrente total de 15A compartilhada em todos os canais, o Servo Hub lhe dará a potência de que você precisa para ter sucesso no campo!

Recursos

Conectividade

- USB

- RS485
- CAN

Canais de servo avançados

- O LED de status indica o status do sinal PWM e falhas
- Medição de corrente de canal individual †
- Potência de saída de canal individualmente comutável †
- Tensão de saída configurável †
- Proteção contra sobrecorrente
- Proteção contra polaridade reversa
- Proteção ESD
- † Recursos disponíveis após futuras atualizações de software.

Principais especificações elétricas

Parametro	Mínimo	Tipo	Máximo	Unidades
Faixa de tensão operacional	7	-	15	V
Tensão de saída	0.5	-	7,4	V
Saída total máxima	-	-	15	A
Canal Corrente máxima de saída †	-	-	3	A

Conector WAGO com trava	Mínimo	Tipo	Máximo	Unidades
Bitola do fio suportado (sólido desencapado/trançado)	26	-	14	AWG
Comprimento da tira de fio desencapado	0,31	0,33	0,355	pol
Bitola do fio suportado (com fio, com ponteira)	24	-	18	Awg

Conectores de servo	Mínimo	Tipo	Máximo	Unidades
Entrada	-	Cabo PWM (padrão de 3 fios com passo de 0,1")	-	-

Conectores de servo	Mínimo	Tipo	Máximo	Unidades
Saída	-	Cabo PWM (padrão de 3 fios com passo de 0,1")	-	-

Consulte as observações sobre as [especificações de corrente máxima](#) para obter mais informações.

Sobre as Especificações de Corrente Máxima

Cada um dos pinos de porta individuais do Servo Hub é classificado para aproximadamente 3 A. Essa classificação, da própria porta, depende muito da qualidade da conexão entre o Servo Hub e o conector do servo que está sendo acionado.

O Servo Hub foi projetado com servos potentes em mente. Muitos dos servos favoritos dos clientes da REV têm uma corrente de parada de 4A ou mais. Embora não acreditemos que a corrente de parada de 4A produza calor suficiente para causar problemas com uma conexão de qualidade e adequadamente assentada, uma conexão ruim pode causar superaquecimento e fuga térmica que pode levar a danos.

A melhor maneira de garantir que você esteja aproveitando ao máximo a saída do seu módulo de alimentação de Servo é verificar se todas as conexões de entrada e saída estão totalmente encaixadas, sem folgas.

Cálculos de Corrente de Saída

É importante garantir que você não exceda a corrente de saída total máxima do seu módulo de alimentação de Servo. Para isso, some a corrente de parada de cada servo que está sendo alimentado pelo módulo de alimentação de Servo. Se a corrente de perda total for superior a 15A, você corre o risco de acionar a proteção contra sobrecorrente. Considere a possibilidade de reduzir o número de servos conectados para evitar o acionamento das proteções de sobrecorrente.

Especificações Mecânicas

Parametro	Mínimo	Tipo	Máximo	Unidades
Número de canais de servo	-	6	-	-
Comprimento	-	85,7 (3,374)	-	mm (pol.)
Largura	-	47,6 (1,874)	-	mm (pol.)
Altura	-	18,3 (0,72)	-	mm (pol.)
Diâmetro do furo de montagem	-	#10 Folga	-	-

Parametro	Mínimo	Tipo	Máximo	Unidades
Padrão de furo de montagem	-	3 por 1,5	-	pol.

Canal de Energia

Este documento foi desenvolvido em parceria com a equipe JUSTICE FTC TEAM #21036

Canal de energia

O REV Servo Hub permite que cada canal de servo seja alimentado individualmente. Isso permite que os usuários alimentem apenas os canais necessários em um determinado momento. Um canal pode ter alimentação sem sinal ou vice-versa, dependendo da configuração.

FTC SDK com Control Hub

No momento, o FTC SDK não oferece apoio ao controle de tempo de execução da energia do canal.

- Os canais são sempre alimentados quando ativados.
- Quando desativados, os canais seguem o comportamento de desativação configurado.

REVLlib para FRC e outros Controles

O REVLlib oferece controle completo do tempo de execução sobre a energia do canal.

- Os canais habilitados podem ser ativados ou desativados a qualquer momento por meio do programa do robô.
- Os canais desativados sempre aderem à sua configuração de comportamento de desativação, independentemente das configurações de tempo de execução.

Comportamento de desativação

Um programa pode controlar dinamicamente se um canal é energizado. No entanto, alguns usuários podem querer que um canal forneça energia mesmo quando estiver desativado. Para resolver isso, é fornecida a configuração “Disable Behavior”. Cada canal tem sua própria configuração Disable Behavior, o que permite um controle detalhado.

Importante: Certifique-se de entender o comportamento do seu servo quando ele tem energia, mas não tem sinal, pois isso pode variar entre os modelos.

kSupplyPower: A alimentação é fornecida ao servo enquanto estiver desativado, mas nenhum sinal é enviado. **Observação:** a configuração **kSupplyPower** é mais semelhante ao comportamento do módulo de alimentação de servo REV, enquanto a configuração **kDoNotSupplyPower** está mais próxima do comportamento das portas de servo da Control Hub. Entretanto, nenhuma dessas configurações é exatamente igual a essas

dispositivos. `kDoNotSupplyPower` : **A energia não é fornecida ao servo enquanto estiver desativado.**

Quando usar cada comportamento de desativação

A seleção do comportamento de desativação apropriado depende das necessidades específicas da sua equipe e do caso de uso de cada canal servo. Abaixo estão alguns cenários em que cada comportamento pode ser vantajoso no **FIRST Tech Challenge (FTC)** ou na **FIRST Robotics Competition (FRC)**.

Ao escolher cuidadosamente o comportamento de desativação apropriado para cada canal de servo, as equipes podem otimizar o desempenho do robô e garantir uma operação confiável em várias condições. É altamente recomendável testar sua configuração durante a prática para evitar surpresas durante a competição.

`kSupplyPower`

Quando a manutenção da posição do servo é crítica: use esse comportamento se o servo precisar manter a posição mesmo quando desativado. Por exemplo:

- Manter uma garra fechada em torno de um elemento do jogo enquanto o robô estiver temporariamente desativado.
- Garantir que um mecanismo, como um braço ou elevador, permaneça no lugar quando o robô estiver desativado durante o teste ou uma pausa na partida.

Quando a transição de desativado para ativado precisa ser perfeita: Se o servo deve manter um estado consistente (por exemplo, evitar movimentos bruscos) quando reativado, o fornecimento de energia garante que o servo permaneça estável.

Quando você conhece o comportamento do servo sem sinal: Os servos se comportam de maneira diferente quando são alimentados sem sinal. Alguns podem manter sua posição, enquanto outros podem se desviar ou “ficar fracos”, e outros podem retornar à posição central. Certifique-se de testar seus servos e entender o comportamento deles nesse modo.

`kDoNotSupplyPower`

Quando o modelo do servo exibe um comportamento indesejável com alimentação, mas sem sinal: Alguns modelos de servo se comportam de forma imprevisível quando estão ligados, mas não estão recebendo sinal. Por exemplo, alguns servos podem apresentar instabilidade ou desvio incontrolável nesse estado, enquanto outros retornam à posição central. Nesses casos, o uso do `kDoNotSupplyPower` garante que o servo não seja ligado até que um sinal válido esteja presente.

Quando o movimento do servo enquanto desativado é aceitável: Se o mecanismo conectado ao servo não exigir posicionamento ou travamento preciso, a remoção da alimentação pode reduzir o desgaste do servo.

- Exemplo: Permitir que um braço de admissão caia em uma posição “neutra” quando não estiver energizado.

Para proteger os servos do uso excessivo: Em alguns casos, a alimentação contínua de um servo quando desativado pode contribuir para o superaquecimento ou o desgaste. Use esse modo para prolongar a vida útil do servo.

Pulso de canal

Este documento foi desenvolvido em parceria com a equipe JUSTICE FTC TEAM #21036

Controle da Largura de Pulso do Servo

Os servomotores são controlados por meio de uma técnica chamada PWM (Modulação por largura de pulso), em que a largura de um pulso determina o comportamento do servo. Normalmente, o pulso é enviado ao servo a cada 20 milissegundos, e a duração do pulso (medida em microssegundos) comunica a posição ou a velocidade desejada.

Servos Angulares

Para servomotores angulares, a largura do pulso corresponde diretamente à posição-alvo do braço do servo. O Servo Hub permite que os usuários personalizem a faixa de largura de pulso para se adequar ao seu servo específico com uma posição mínima, máxima e central configurável. Os usuários podem ajustar esses valores para que correspondam aos recursos do servo ou ao comportamento desejado.

- Uma largura de pulso mais curta, normalmente 1.000 μ s (1 ms) ou 500 μ s (0,5 ms), move o servo para sua posição mínima (por exemplo, totalmente no sentido anti-horário).
- Uma largura de pulso mais longa, normalmente 2.000 μ s (2 ms) ou 2.500 μ s (2,5 ms), move o servo para sua posição máxima (por exemplo, totalmente no sentido horário).
- Uma largura de pulso próxima ao ponto médio, normalmente 1.500 μ s (1,5 ms), posiciona o braço do servo no centro.

Exemplo de Intervalo

Largura de pulso (μ s)	Posição
50 μ s	Totalmente no sentido anti-horário
1,500 μ s	Centro
2,500 μ s	Totalmente no sentido horário

Servos de Rotação Contínua

Para servos de rotação contínua, a largura do pulso determina a direção e a velocidade da rotação, e não a posição. Os ajustes mínimo, máximo e central configuráveis do Servo Hub também podem ajudar a calibrar servos de rotação contínua. Ajustes finos podem ser feitos na largura de pulso central para garantir que o servo pare com precisão no ponto neutro.

- Uma largura de pulso de 1.500 μ s (1,5 ms) normalmente interrompe o servo (sem movimento).
- As larguras de pulso mais curtas (por exemplo, 1.000 μ s) fazem com que o servo gire em uma direção, com a velocidade aumentando à medida que a largura do pulso diminui.
- As larguras de pulso mais longas (por exemplo, 2.000 μ s) fazem com que o servo gire na direção oposta, com o aumento da velocidade à medida que a largura do pulso aumenta.

Exemplo de faixa (típica):

Largura de pulso (μ s)	Ação
500 μ s	Velocidade máxima (sentido horário)
1,500 μ s	Parada
2,500 μ s	Velocidade máxima (sentido anti-horário)

Dicas e Truques

Compreender como a largura de pulso controla o servo e aproveitar os recursos do REV Servo Hub pode ajudar a otimizar o desempenho do servo para o seu robô. Teste cada servo para confirmar seu comportamento e a faixa de largura de pulso suportada antes de integrá-lo ao seu sistema. Sugerimos o seguinte como um bom ponto de partida para entender seus servomotores:

- Variações do servo: Sempre verifique a documentação de seu servo específico, pois as faixas de largura de pulso e o comportamento podem variar.
- Integridade do sinal: Certifique-se de que o sinal PWM esteja limpo e consistente para evitar tremulações ou comportamento errático.

Configurando Servo Hub

Esta página abordará informações sobre conceitos de configuração específicos do Servo Hub. Para obter mais informações sobre a configuração geral no REVLib, consulte esta página.

Classes de Configuração

Embora o Servo Hub tenha sua própria classe de configuração, `ServoHubConfig`, a maior parte da configuração ocorre na classe `ServoChannelConfig`.

API Documentation

ServoHubConfig	Java	C++
----------------	------	-----

Parâmetros Persistentes

A configuração de um Servo Hub mantém automaticamente as definições de configuração entre os ciclos de energia ao chamar o método `configure()`.

A persistência dos parâmetros envolve salvá-los na memória do Servo Hub, o que consome muito tempo e bloqueia a comunicação com o dispositivo.

Casos de uso

Recomenda-se atualizar a maioria dos parâmetros do dispositivo durante a configuração inicial do dispositivo no início do programa para garantir que o controlador mantenha sua configuração no caso de um ciclo de energia durante a operação, por exemplo, devido a um disparo do disjuntor ou a uma queda de energia.

Em geral, recomenda-se não fazer atualizações na configuração durante a operação para evitar o bloqueio do programa e afetar o desempenho do robô.

Abaixo está um exemplo de um desses casos:

■ ■

JAVA

```
Robot() {  
    ServoHubConfig config = new ServoHubConfig();  
    config
```

```

        .channel0.pulseRange(500, 1500, 2500)
        .disableBehavior(ServoChannelConfig.BehaviorWhenDisabled.kSupplyPower);

// Persist parameters and reset any not explicitly set above to
// their defaults.
servoHub.configure(config, ServoHub.ResetMode.kResetSafeParameters);
}

void reduceRange() {
    ServoHubConfig config = new ServoHubConfig();
    config.channel0.pulseRange(500, 1500, 2500)

    // Don't reset the parameters the at are not explicitly set above
    servoHub.configure(config, ServoHub.ResetMode.kNoResetSafeParameters);
}

```

Copy

```

using namespace rev::servohub;

Robot() {
    ServoHubConfig config;
    config
        .channel0.pulseRange(500, 1500, 2500)
        .disableBehavior(ServoChannelConfig::BehaviorWhenDisabled::kSupplyPower);

    // Persist parameters and reset any not explicitly set above to
    // their defaults.
    servoHub.configure(config, ServoHub::ResetMode::kResetSafeParameters);
}

void reduceRange() {
    ServoHubConfig config;
    config.channel0.pulseRange(500, 1500, 2500)

    // Don't reset the parameters the at are not explicitly set above
    servoHub.configure(config, ResetMode.kNoResetSafeParameters);
}

```

■ ■

JAVA


```

Robot() {
    ServoHubConfig config = new ServoHubConfig();
    config
        .channel0.pulseRange(500, 1500, 2500)
        .disableBehavior(ServoChannelConfig.BehaviorWhenDisabled.kSupplyPower);

    // Persist parameters and reset any not explicitly set above to
    // their defaults.
    servoHub.configure(config, ServoHub.ResetMode.kResetSafeParameters);
}

void reduceRange() {
    ServoHubConfig config = new ServoHubConfig();
    config.channel0.pulseRange(500, 1500, 2500)

    // Don't reset the parameters the at are not explicitly set above
    servoHub.configure(config, ServoHub.ResetMode.kNoResetSafeParameters);
}

```

C++

```

using namespace rev::servohub;

Robot() {
    ServoHubConfig config;
    config
        .channel0.pulseRange(500, 1500, 2500)
        .disableBehavior(ServoChannelConfig::BehaviorWhenDisabled::kSupplyPower);

    // Persist parameters and reset any not explicitly set above to
    // their defaults.
    servoHub.configure(config, ServoHub::ResetMode::kResetSafeParameters);
}

void reduceRange() {
    ServoHubConfig config;
    config.channel0.pulseRange(500, 1500, 2500)

    // Don't reset the parameters the at are not explicitly set above
    servoHub.configure(config, ResetMode.kNoResetSafeParameters);
}

```


Padrões de LEDs de status do Servo Hub

Este documento foi desenvolvido em parceria com a equipe JUSTICE FTC TEAM #21036

Indicadores de LED

O REV Servo Hub usa LEDs para fornecer feedback visual sobre o status do dispositivo e os estados dos canais individuais. A compreensão desses indicadores ajuda a diagnosticar problemas e a monitorar o desempenho do sistema.

Importante: esses padrões de led se aplicam somente à versão de firmware 24.0.0 e posterior

Status geral do LED

O LED de status principal do Servo Hub comunica o estado geral do dispositivo. Abaixo, há uma tabela que explica os vários padrões e seus significados:

Status do LED	Descrição do LED	Quando	Status do Hub
	Magenta Piscando	Qualquer Momento	O Servo Hub está ligado, mas não está conectado a um controlador ou ao REV Hardware Client.
	Azul Sólido	Qualquer Momento	O Servo Hub está conectado à comunicação estabelecida pelo REV Hardware Client.
	Ciano Sólido	Qualquer Momento	O Servo Hub está conectado a um roborIO ou a outro controlador CAN. †
	Verde Sólido	Qualquer Momento	O Servo Hub está conectado a um Control Hub ou a outro controlador RS-485. O número de piscadas azuis é o mesmo que o endereço do Servo Hub. O endereço padrão de fábrica é 3.
	Laranja/Ciano Piscando	Qualquer Momento	A tensão da bateria é inferior a 5,5 V. Verifique a fonte de alimentação do Servo Hub. Essa falha será eliminada quando a tensão de entrada for aumentada para mais de 6,5 V.
	Laranja/Amarelo Piscando	Qualquer Momento	Foi detectada uma falha no CAN. Verifique a fiação e as conexões do barramento CAN.
	Laranja/Magenta Piscando	Qualquer Momento	Ocorreu uma falha de sobrecorrente. Verifique os servos conectados e reduza a carga, se necessário.

† As falhas não são relatadas aos registros quando o Servo Hub está conectado via RS485.

LED de Status do Canal






Cada canal de servo no REV Servo Hub tem seu próprio LED dedicado, que fornece feedback sobre o estado do canal e o sinal PWM atual. Veja a seguir o significado de cada padrão de LED:

Observações:

- Os LEDs dos canais refletem o estado atual do sinal PWM, ajudando os usuários a verificar o comportamento do servo em tempo real.

- Os canais desativados ainda mostram feedback por meio de um LED âmbar piscante, facilitando a diferenciação dos canais inativos.

Servos Angulares

Status do LED	Descrição do LED	Quando	Status do Hub
	Magenta Piscando	Qualquer Momento	O Servo Hub está ligado, mas não está conectado a um controlador ou ao REV Hardware Client.
	Azul Sólido	Qualquer Momento	O Servo Hub está conectado à comunicação estabelecida pelo REV Hardware Client.
	Ciano Sólido	Qualquer Momento	O Servo Hub está conectado a um robeRIO ou a outro controlador CAN. †
	Verde Sólido	Qualquer Momento	O Servo Hub está conectado a um Control Hub ou a outro controlador RS-485. O número de piscadas azuis é o mesmo que o endereço do Servo Hub. O endereço padrão de fábrica é 3.
	Laranja/Ciano Piscando	Qualquer Momento	A tensão da bateria é inferior a 5,5 V.Verifique a fonte de alimentação do Servo Hub.Essa falha será eliminada quando a tensão de entrada for aumentada para mais de 6,5 V.
	Laranja/Amarelo Piscando	Qualquer Momento	Foi detectada uma falha no CAN. Verifique a fiação e as conexões do barramento CAN.
	Laranja/Magenta Piscando	Qualquer Momento	Ocorreu uma falha de sobrecorrente. Verifique os servos conectados e reduza a carga, se necessário.
†	As falhas não são relatadas aos registros quando o Servo Hub está conectado via RS485.		

Servos de rotação contínua

Status do LED	Descrição do LED	Quando	Status do Hub
	Âmbar Sólido	Qualquer Momento	O servo é interrompido (normalmente 1.500 μ s de largura de pulso).
	Verde Piscando	Qualquer Momento	O servo está andando para frente (por exemplo, 1.500-2.000 μ s).
	Verde Sólido	Qualquer Momento	O servo está andando para frente na velocidade máxima (por exemplo, 2.000 μ s).
	Vermelho Piscando	Qualquer Momento	O servo está andando em sentido inverso (por exemplo, 1.500-1.000 μ s).
	Vermelho Sólido	Qualquer Momento	O servo está andando em sentido reverso na velocidade máxima (por exemplo, 1.000 μ s).
	Âmbar Piscando	Qualquer Momento	O canal está desativado. O pino de sinal é puxado para baixo e nenhum sinal PWM está sendo enviado.
	Âmbar Piscando Rapidamente	Qualquer Momento	O canal está com falha. Isso pode significar que o dispositivo não tem energia suficiente, que nem o RoboRIO nem o sinal do Control Hub estão presentes ou que o canal está sofrendo um evento de sobrecorrente.

Servos de Comando

Este documento foi desenvolvido em parceria com a equipe JUSTICE FTC TEAM #21036

Controles Básicos do Servo

Os servomotores são motores especializados que podem ser controlados para se mover em um ângulo específico em vez de girar continuamente como um motor CC. Para obter mais informações gerais sobre servos, consulte esta página; para obter informações mais detalhadas, consulte https://en.wikipedia.org/wiki/Servo_control

Configuração do Servo Hub

O controle do servo no REVLib é acessado por meio do objeto ServoHub. Esse objeto gerencia os objetos ServoChannel individuais e monitora o dispositivo geral. O objeto ServoChannel controla cada servomotor e monitora sua operação. Ele também contém todos os métodos para configurar e controlar o servo. Ele pode ser acessado conforme mostrado abaixo:

■ ■

JAVA

```
// Initialize the servo hub
ServoHub m_servoHub = new ServoHub(deviceID);

// Obtain a servo channel controller
ServoChannel m_channel0 = m_servoHub.getServoChannel(ChannelId.kChannelId0);
ServoChannel m_channel1 = m_servoHub.getServoChannel(ChannelId.kChannelId1);
...
ServoChannel m_channel5 = m_servoHub.getServoChannel(ChannelId.kChannelId5);
```

API Docs: [ServoHub](#), [ServoChannel](#)

JAVA

```
using namespace rev::servohub;
using namespace rev::servohub::ServoChannel;
```

```
// Initialize the servo hub
ServoHub m_servoHub{ deviceID };

//Obtain a reference to a servo channel controller
ServoChannel& m_channel0 = m_servoHub.GetServoChannel(ChannelId.kChannelId0);
ServoChannel& m_channel1 = m_servoHub.GetServoChannel(ChannelId.kChannelId1);
...
ServoChannel& m_channel5 = m_servoHub.GetServoChannel(ChannelId.kChannelId5);
```

API Docs: [ServoHub](#), [ServoChannel](#)

□ □

Copy

```
// Initialize the servo hub
ServoHub m_servoHub = new ServoHub(deviceID);

// Obtain a servo channel controller
ServoChannel m_channel0 = m_servoHub.getServoChannel(ChannelId.kChannelId0);
ServoChannel m_channel1 = m_servoHub.getServoChannel(ChannelId.kChannelId1);
...
ServoChannel m_channel5 = m_servoHub.getServoChannel(ChannelId.kChannelId5);
```

API Docs: [ServoHub](#), [ServoChannel](#)

C++

```
using namespace rev::servohub;
using namespace rev::servohub::ServoChannel;

// Initialize the servo hub
ServoHub m_servoHub{ deviceID };

//Obtain a reference to a servo channel controller
ServoChannel& m_channel0 = m_servoHub.GetServoChannel(ChannelId.kChannelId0);
ServoChannel& m_channel1 = m_servoHub.GetServoChannel(ChannelId.kChannelId1);
...
ServoChannel& m_channel5 = m_servoHub.GetServoChannel(ChannelId.kChannelId5);
```


Configuração do Período de Pulso do Servo

Em um servomotor, a largura do pulso determinará a distância de giro do motor. O período de pulso, por outro lado, determinará a frequência com que o pulso é enviado ao servo. O ServoHub é compatível com um período de pulso de 4,5 a 20 ms (especificado em microssegundos). O ServoHub é compatível com um período de pulso separado para cada banco, compondo os servos 0-2 e 3-5, respectivamente. Essas configurações podem ser acessadas conforme mostrado abaixo:

■ ■

JAVA

```
// Set the pulse period for channels 0-2 to 5ms (5000 microseconds)
m_servoHub.setBankPulsePeriod(ServoHub.Bank.kBank0_2, 5000);

// Set the pulse period for channels 3-5 to 20ms (20000 microseconds)
m_servoHub.setBankPulsePeriod(ServoHub.Bank.kBank3_5, 20000);
```

API Docs: [setBankPulsePeriod](#)

Copy

```
using namespace rev::servohub;

// Set the pulse period for channels 0-2 to 5ms (5000 microseconds)
m_servoHub.SetBankPulsePeriod(ServoHub::Bank::kBank0_2, 5000);

// Set the pulse period for channels 3-5 to 20ms (20000 microseconds)
m_servoHub.SetBankPulsePeriod(ServoHub::Bank::kBank3_5, 20000);
```

API Docs: [SetBankPulsePeriod](#)

■ ■

Copy

```
// Set the pulse period for channels 0-2 to 5ms (5000 microseconds)
m_servoHub.setBankPulsePeriod(ServoHub.Bank.kBank0_2, 5000);

// Set the pulse period for channels 3-5 to 20ms (20000 microseconds)
m_servoHub.setBankPulsePeriod(ServoHub.Bank.kBank3_5, 20000);
```

API Docs: [setBankPulsePeriod](#)

C++

```
using namespace rev::servohub;

// Set the pulse period for channels 0-2 to 5ms (5000 microseconds)
m_servoHub.SetBankPulsePeriod(ServoHub::Bank::kBank0_2, 5000);

// Set the pulse period for channels 3-5 to 20ms (20000 microseconds)
m_servoHub.SetBankPulsePeriod(ServoHub::Bank::kBank3_5, 20000);
```

API Docs: [SetBankPulsePeriod](#)

Controle de um Servo Individual

Os servos individuais são controlados por meio dos objetos ServoChannel. Como mostrado acima, você obtém uma referência a um objeto ServoChannel chamando o método `getServoChannel()` no ServoHub. Você pode definir o seguinte em um ServoChannel:

- **PulseWidth** (Largura de pulso) - determina a posição do servo (500 a 2500 microssegundos).
- **Enabled** - ativa/desativa o servo - quando desativado, o servo manterá a energia de acordo com o `DisableBehavior` configurado para o canal específico.
- **Powered** (Alimentado) - liga/desliga a alimentação do servo

■ ■

JAVA

```
// Power on channels 0, 1, and 4
m_channel0.setPowered(true);
m_channel1.setPowered(true);
m_channel4.setPowered(true);

// Enabled them as well
m_channel0.setEnabled(true);
m_channel1.setEnabled(true);
m_channel4.setEnabled(true)

// Set the servo on channel 0 to the center (1500 microseconds)
m_channel0.setPulseWidth(1500);
```

```
// Set the servo on channel 1 to the far left (500 microseconds)
m_channel1.setPulseWidth(500);

// Set the servo on channel 4 to the far right(2500 microseconds)
m_channel4.setPulseWidth(2500);
```

API Docs: [setPulseWidth](#), [setPowered](#), [setEnabled](#)

Copy

```
using namespace rev::servohub;

// Power on channels 0, 1, and 4
m_channel0.SetPowered(true);
m_channel1.SetPowered(true);
m_channel4.SetPowered(true);

// Enabled them as well
m_channel0.SetEnabled(true);
m_channel1.SetEnabled(true);
m_channel4.SetEnabled(true)

// Set the servo on channel 0 to the center (1500 microseconds)
m_channel0.SetPulseWidth(1500);

// Set the servo on channel 1 to the far left (500 microseconds)
m_channel1.SetPulseWidth(500);

// Set the servo on channel 4 to the far right(2500 microseconds)
m_channel4.SetPulseWidth(2500);
```

API Docs: [SetPulseWidth](#), [SetEnabled](#), [SetPowered](#)

■ ■

Copy

```
// Power on channels 0, 1, and 4
m_channel0.setPowered(true);
m_channel1.setPowered(true);
m_channel4.setPowered(true);
```

```
// Enabled them as well
m_channel0.setEnabled(true);
m_channel1.setEnabled(true);
m_channel4.setEnabled(true)

// Set the servo on channel 0 to the center (1500 microseconds)
m_channel0.setPulseWidth(1500);

// Set the servo on channel 1 to the far left (500 microseconds)
m_channel1.setPulseWidth(500);

// Set the servo on channel 4 to the far right(2500 microseconds)
m_channel4.setPulseWidth(2500);
```

API Docs: [setPulseWidth](#), [setPowered](#), [setEnabled](#)

C++

```
using namespace rev::servohub;

// Power on channels 0, 1, and 4
m_channel0.SetPowered(true);
m_channel1.SetPowered(true);
m_channel4.SetPowered(true);

// Enabled them as well
m_channel0.SetEnabled(true);
m_channel1.SetEnabled(true);
m_channel4.SetEnabled(true)

// Set the servo on channel 0 to the center (1500 microseconds)
m_channel0.SetPulseWidth(1500);

// Set the servo on channel 1 to the far left (500 microseconds)
m_channel1.SetPulseWidth(500);

// Set the servo on channel 4 to the far right(2500 microseconds)
m_channel4.SetPulseWidth(2500);
```

API Docs: [SetPulseWidth](#), [SetEnabled](#), [SetPowered](#)

Solução de problemas

Este documento foi desenvolvido em parceria com a equipe JUSTICE FTC TEAM #21036

Guia de Solução de Problemas

Este guia de solução de problemas ajuda a diagnosticar e resolver problemas com o REV Servo Hub.

Problemas de energia

Se o Servo Hub não estiver respondendo (sem luzes):

1. Verifique a fonte de alimentação:
 - Verifique se a fonte de alimentação fornece tensão suficiente (recomenda-se 6-12 V).
 - Se estiver usando um REV PDH, verifique o disjuntor do canal que está alimentando o hub de servo.
2. Inspeção a fiação:
 - Verifique se todos os fios estão conectados de forma segura.
 - Faça um teste de tração nas conexões de energia.
3. Entre no modo de recuperação:
 - Se o Servo Hub tiver energia, mas nenhum LED acender, siga as instruções do modo de recuperação abaixo.

Falhas de sobrecorrente

O Servo Hub protege a si mesmo e aos servos conectados contra condições de sobrecorrente. Há dois tipos de falhas de sobrecorrente:

1. Channel Overcurrent Fault
2. Total Device Overcurrent Fault

Falha de sobrecorrente do canal

Condição: Um canal excede 6A por um período prolongado ou apresenta picos curtos acima de 7A.

Indicadores: O LED do canal pisca na cor âmbar em uma frequência alta e/ou a alimentação do canal afetado é removida.

Resolução:

1. Remova a carga do servo
2. Permita que a corrente caia para eliminar a falha.

Falha de sobrecorrente total do dispositivo

Condição: A corrente total em todos os seis canais excede 15A.

Indicadores: Todos os LEDs dos canais piscam na cor âmbar em uma frequência alta e/ou a alimentação de todos os canais foi removida.

Resolução:

1. Desconecte os servos e verifique se há falhas ou consumo excessivo de corrente.
2. Certifique-se de que nenhum canal esteja em curto-circuito.
3. A falha será eliminada 1 segundo depois que a corrente total cair abaixo de 15A.

Causas comuns:

1. As falhas de sobrecorrente podem indicar carga excessiva ou mau funcionamento do servo. Desconecte e teste os servos individualmente.
2. Servos de alta potência paralisados (por exemplo, Axon Max com uma corrente de paralisação de ~4A).
3. Curto-circuito na fiação do servo 43.

Avisos de Bateria Fraca

O Servo Hub alternará entre azul e laranja no LED de status principal quando a tensão de entrada estiver baixa:

- Limite de baixa tensão: Abaixo de 5,5V.
- Tensão de compensação: Acima de 6,5 V.

Resolução:

- Verifique a voltagem da bateria que está alimentando o Servo Hub e recarregue-a, se necessário.
- Certifique-se de que as conexões com a bateria estejam seguras.
- A baixa tensão pode causar um comportamento inesperado.

Falhas no Barramento CAN

Uma falha de CAN ocorre quando o Servo Hub detecta uma comunicação não confiável no barramento CAN. O LED de status principal alternará entre amarelo e laranja.

Etapas de solução de problemas:

1. Inspecione a fiação:
 - Faça um teste de puxão para garantir que as conexões estejam seguras.
 - Verifique se há fio desencapado suficiente nos conectores Wago.
2. Verifique os resistores de terminação:
 - Garanta a terminação adequada em ambas as extremidades do barramento CAN.
3. Teste de curtos:
 - Inspecione se há curtos na fiação CAN.

Nenhuma Conexão Detectada

Quando o Servo Hub não puder detectar uma conexão com um controlador ou com o REV Hardware Client, o LED de status principal piscará em magenta.

Etapas de solução de problemas:

1. Verifique o Hardware Client:
 - Abra o REV Hardware Client e verifique se ele reconhece o Servo Hub.
2. Verifique a conexão CAN:
 - Verifique a fiação do barramento CAN.
 - Use o Hardware Client para verificar se outros dispositivos no barramento CAN estão visíveis.
3. Inspecione o roboRIO:
 - Verifique se a roboRIO tem energia.
 - Verifique a configuração e as conexões da roboRIO.

Problemas de Servo e Canal

O Servo Não está Respondendo

- Verifique se o canal está ativado (o LED não está piscando na cor âmbar).

- Verifique se há conexões soltas ou danos na fiação do servo.
- Verifique se o servo é compatível com a faixa de largura de pulso configurada.

Movimento Incorreto ou Instável do Servo

- Inspecione a configuração Disable Behavior (Desativar comportamento):
- Alguns servos podem apresentar tremores ou comportamento inadequado quando estão ligados, mas não estão recebendo sinal. Considere o uso do `kDoNotSupplyPower`
- Teste com outro servo para descartar problemas de hardware.

O LED do Canal Não Acende

- Se o LED do canal estiver apagado, verifique se o canal está configurado corretamente no programa.
- Verifique a fiação e a funcionalidade do servo.
- Teste com outro servo para confirmar a operação do canal.

Problemas de Software

Não é possível configurar vários hubs de servo para uso com um hub de controle

- Verifique novamente se cada Servo Hub tem uma ID CAN exclusiva, conectando-o via USB ao REV Hardware Client.
- Os servo hubs devem ter uma ID diferente da de um hub de expansão. Os hubs de expansão têm como padrão a ID 1 ou 2.

O Servo Hub não está aparecendo no menu de configuração

Você deve estar executando o aplicativo Robot Controller e o aplicativo Driver Station versão 10.0 para usar um Servo Hub.

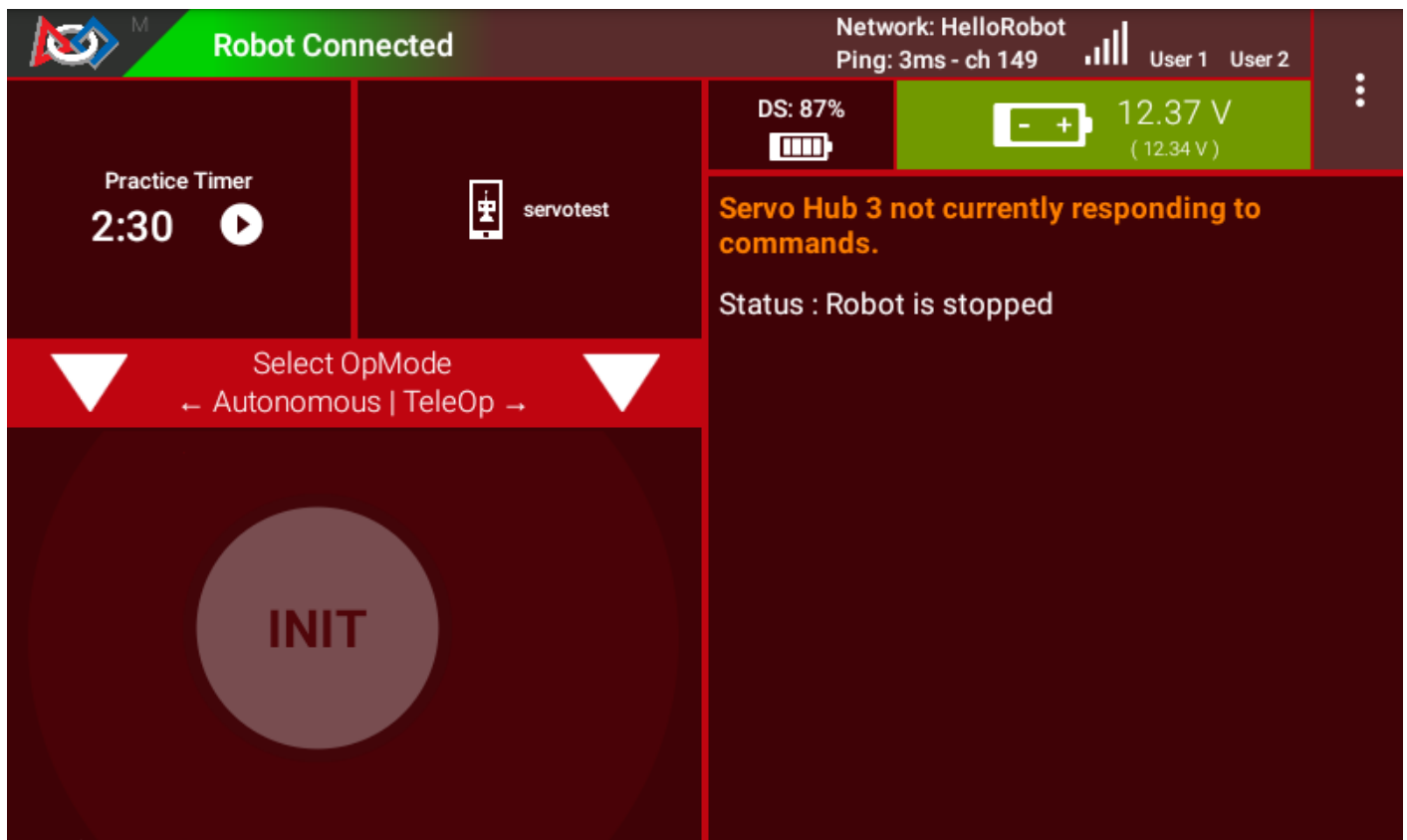
- Verifique primeiro se o Servo Hub está recebendo alimentação adequada e se o cabo RS485 está seguro.
- Verifique novamente se cada Servo Hub tem uma ID CAN exclusiva, conectando-o via USB ao REV Hardware Client.
- Os servo hubs devem ter uma ID diferente da de um hub de expansão. Os hubs de expansão têm como padrão a ID 1 ou 2.

Não é possível definir a ID do CAN/ID do CAN não está sendo salva no Servo Hub

Quando um hub de servo é conectado a um hub de controle, diretamente ou por meio de outro hub, ele será definido para o modo "somente leitura" ao interagir com o REV Hardware Client. Isso significa que o Client não pode atualizar a ID ou o firmware, e os recursos, como a capacidade de executar servos, não estarão disponíveis.

- Desligue o robô ou desconecte o cabo RS845 que leva ao Servo Hub
- Faça um ciclo de energia do Servo Hub.
- Agora você deve conseguir conectar o Servo Hub via USB-C ao REV. Hardware Client para definir a ID ou usar os recursos do Client.

Driver Hub mostrando o erro "Servo Hub não está respondendo aos comandos no momento"

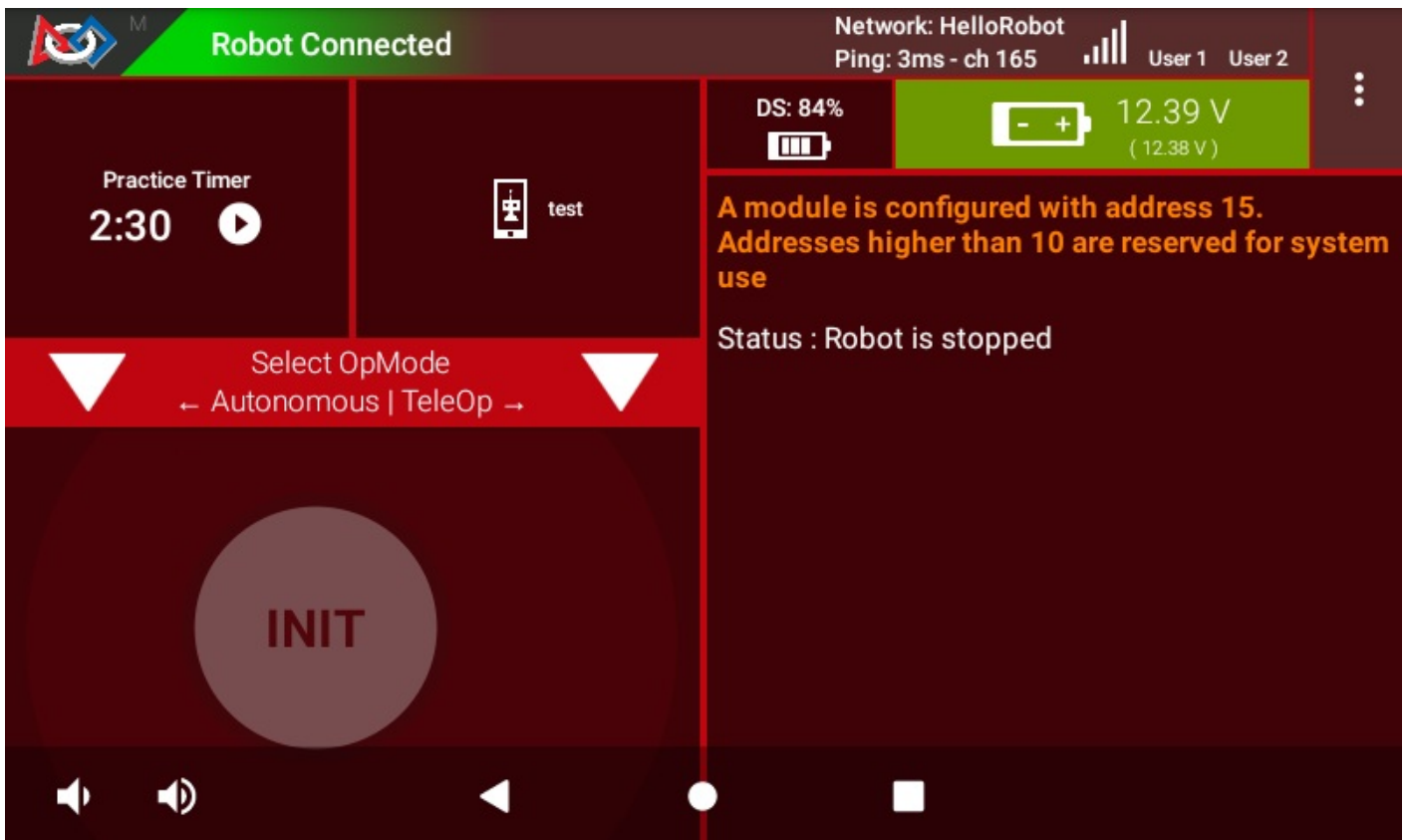


Driver Hub mostrando o erro "Servo Hub não está respondendo aos comandos no momento"

- Verifique novamente se o Servo Hub está recebendo a fiação adequada e segura
- Se a ID do Servo Hub tiver sido alterada ou se um Servo Hub diferente com uma ID diferente tiver sido conectado como substituto:
 - Altere a ID do Servo Hub para que corresponda à da configuração OU
 - Crie um novo arquivo de configuração com a ID atualizada

Se estiver removendo completamente o Servo Hub, crie um novo arquivo de configuração.

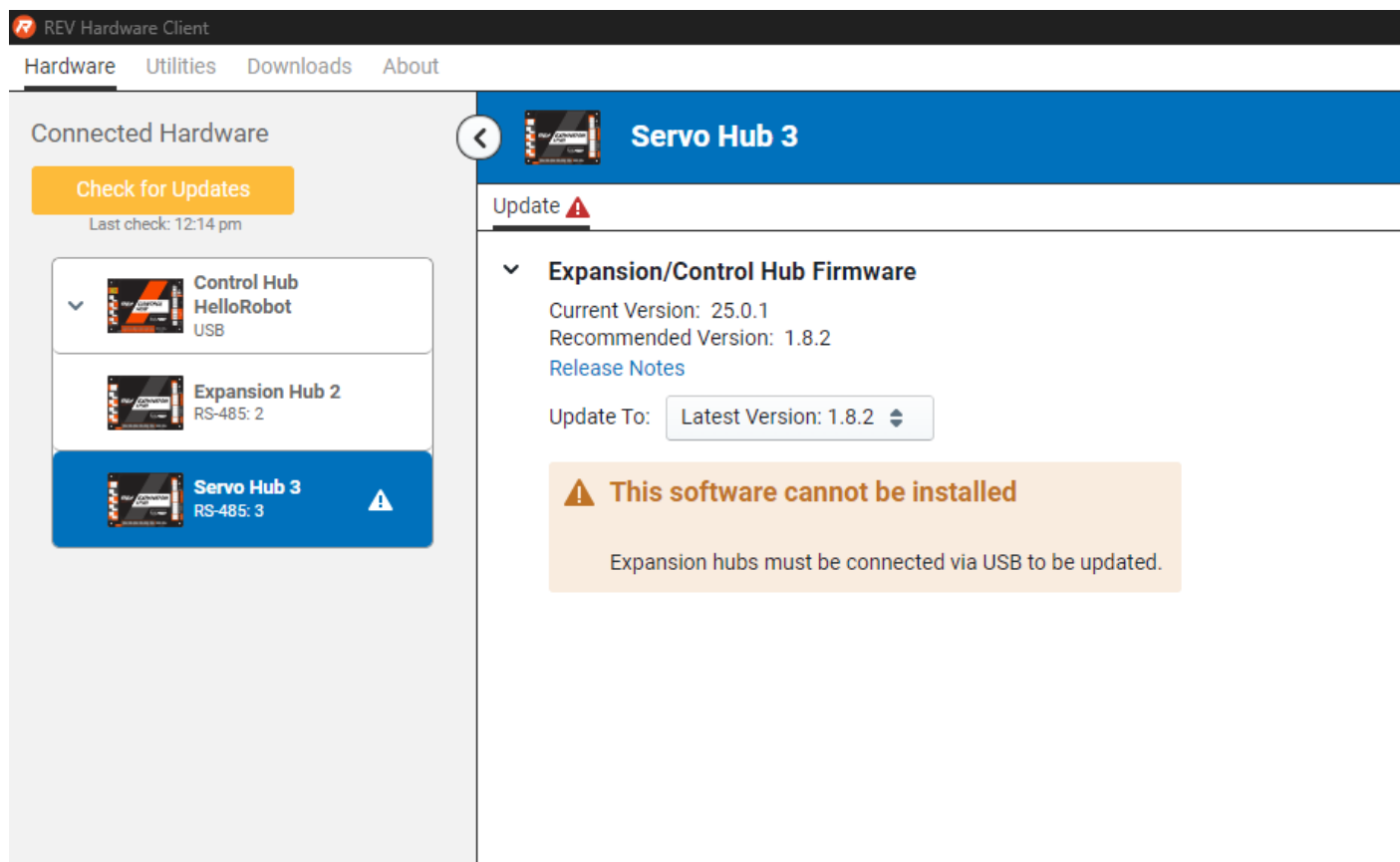
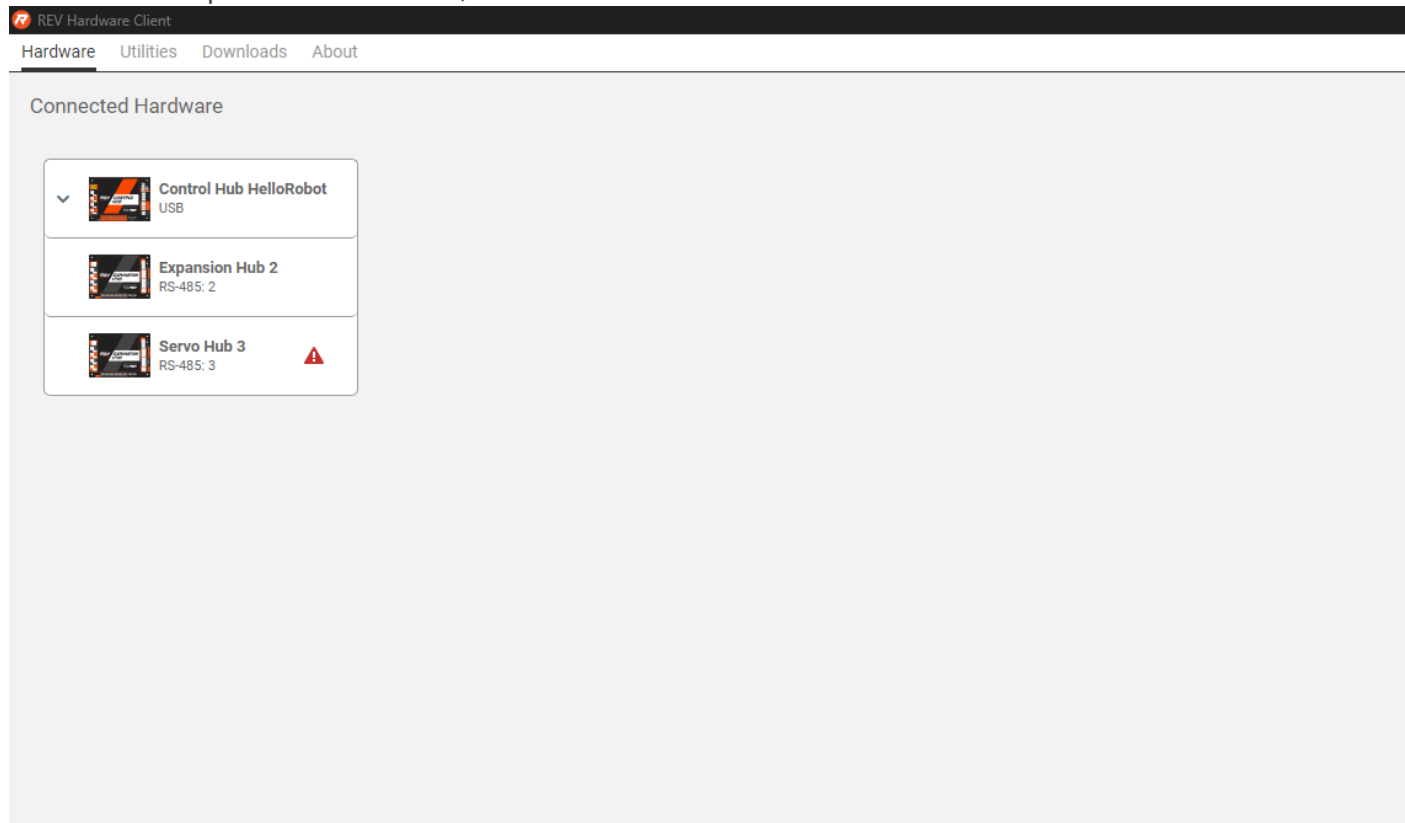
Driver Hub mostrando o aviso "Endereços maiores que 10 são reservados para uso do sistema"



Altere a ID do Servo Hub para um intervalo entre 1 e 10 para evitar possíveis conflitos. Por padrão, os hubs de expansão têm ID 1 ou 2.

O servo hub aparece como um hub de expansão no cliente de hardware REV

Ao se conectar por USB-C a um Hub de controle com um Servo Hub conectado, ele aparecerá como um Hub de expansão no Cliente, conforme mostrado abaixo:



Esse é o comportamento esperado a partir da versão 1.7.0 do RHC. Para atualizar o hub de servo, desconecte-o do hub de controle e desligue-o antes de conectar apenas o hub de servo via USB-C.

Modo de Recuperação de Firmware

Se o Servo Hub não estiver respondendo, use o modo de recuperação para restaurar a funcionalidade.

Etapas para entrar no modo de recuperação:

1. Desligue o Servo Hub.
2. Pressione e mantenha pressionado o botão de modo no Servo Hub.
3. Enquanto estiver segurando o botão, ligue o Servo Hub.
4. Solte o botão quando o LED vermelho do canal 4 e o LED verde do canal 5 estiverem acessos.

O Servo Hub agora estará pronto para ser recuperado por meio do REV Hardware Client.

Ao seguir este guia, os usuários podem diagnosticar e resolver os problemas mais comuns com o REV Servo Hub. Se os problemas persistirem, entre em contato com o suporte da REV Robotics para obter mais assistência.

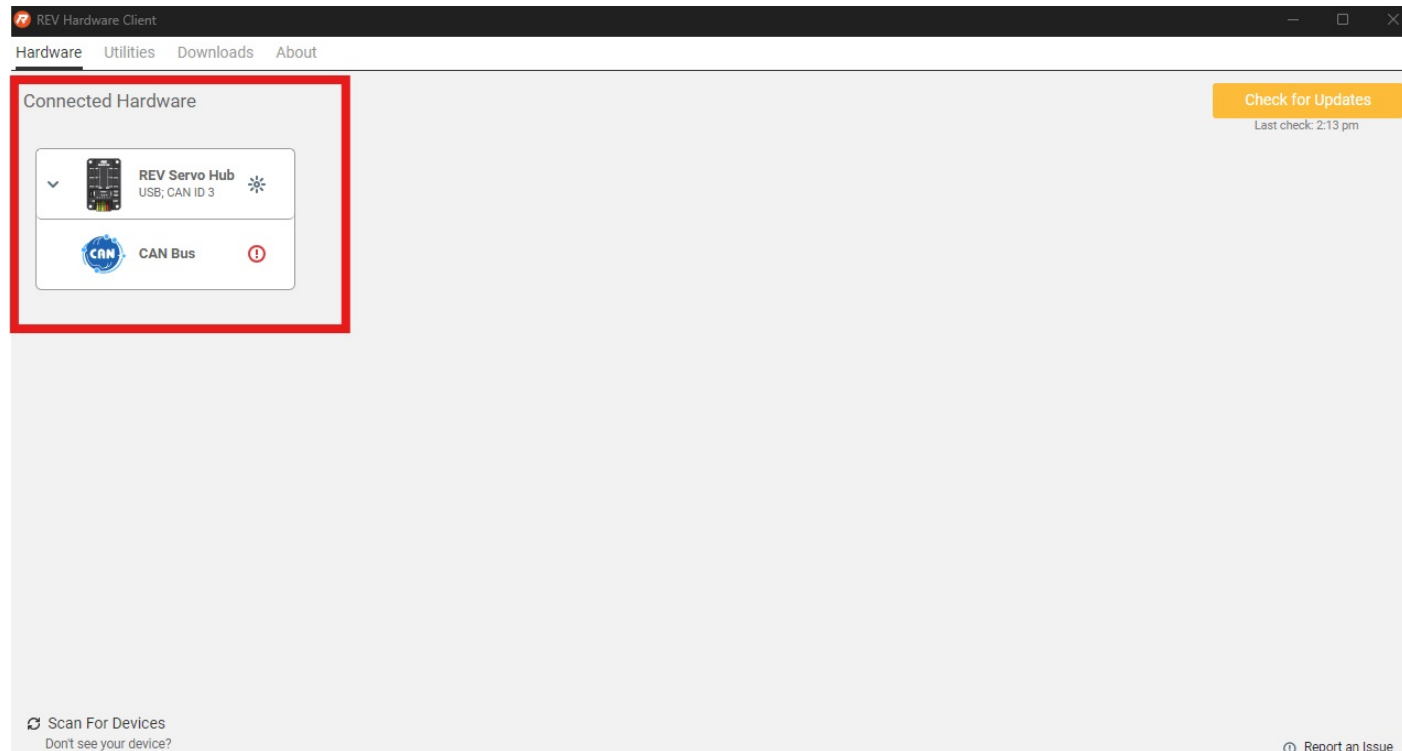
Configuração do Servo Hub com a Control Hub

Este documento foi desenvolvido em parceria com a equipe JUSTICE FTC TEAM #21036

Configuração de IDs do Servo Hub

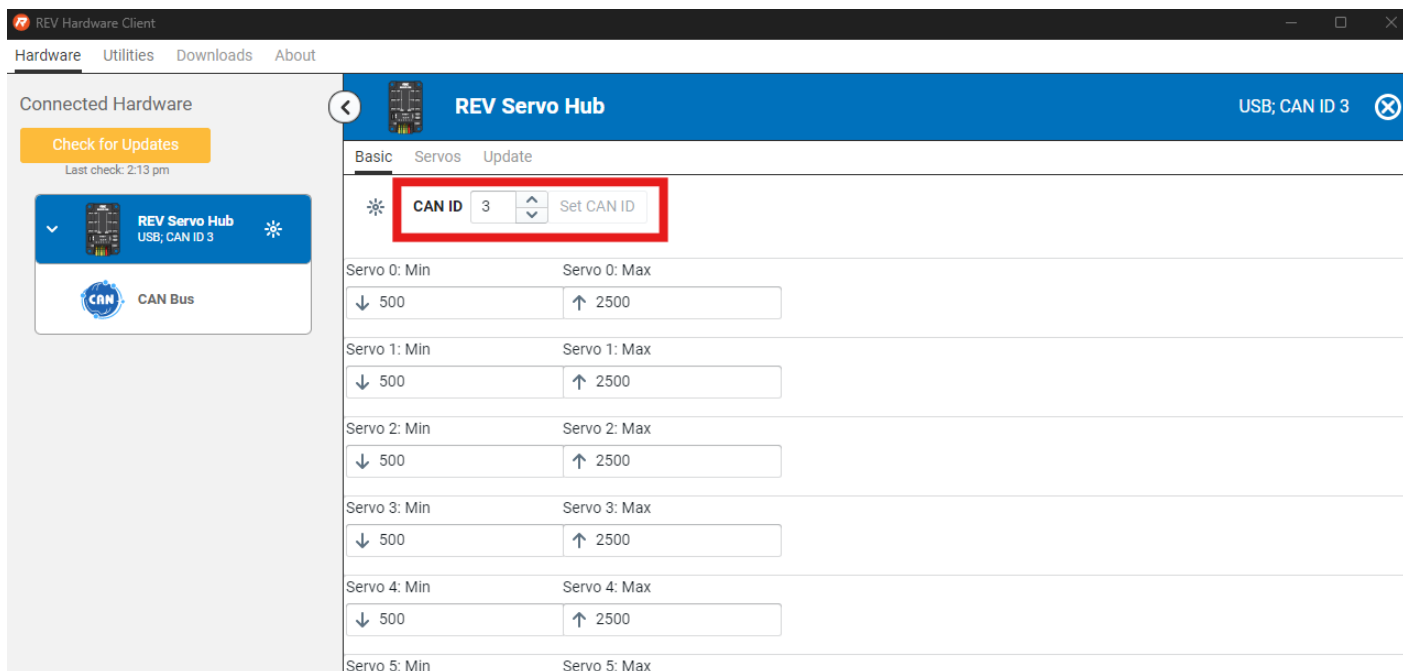
As IDs do Servo Hub devem ser definidas enquanto o Servo Hub estiver desconectado de uma Control Hub OU antes de ligar o robô. Os Servo Hubs conectados anteriormente a um robô podem precisar de um ciclo de energia antes de alterar as IDs.

Por padrão, o ID do Servo Hub deve ser definido como 3. Isso pode ser alterado conectando o Servo Hub diretamente ao REV Hardware Client usando um cabo USB-C. O Servo Hub aparecerá na lista de hardware, conforme mostrado abaixo:



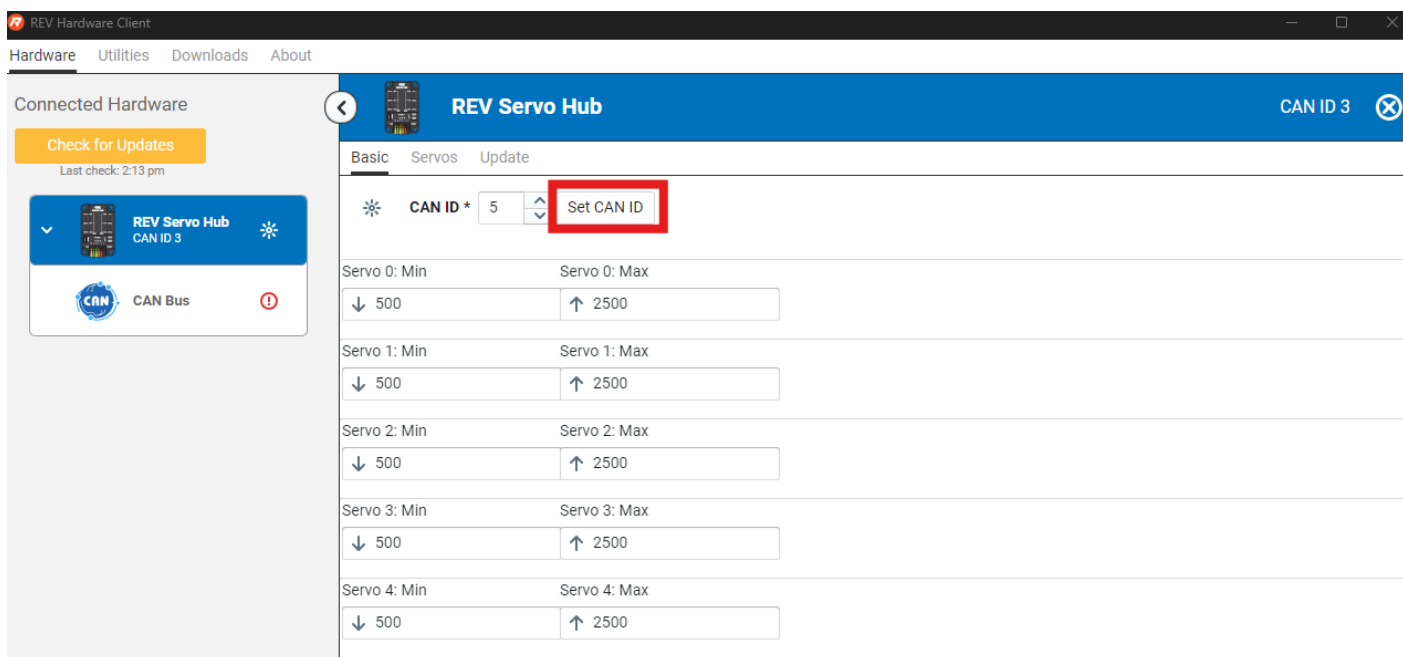
A "CAN ID" é a ID individual do Servo Hub

Cada Servo Hub e Expansion Hub deve ter uma ID exclusiva antes de poder concluir o processo de configuração.



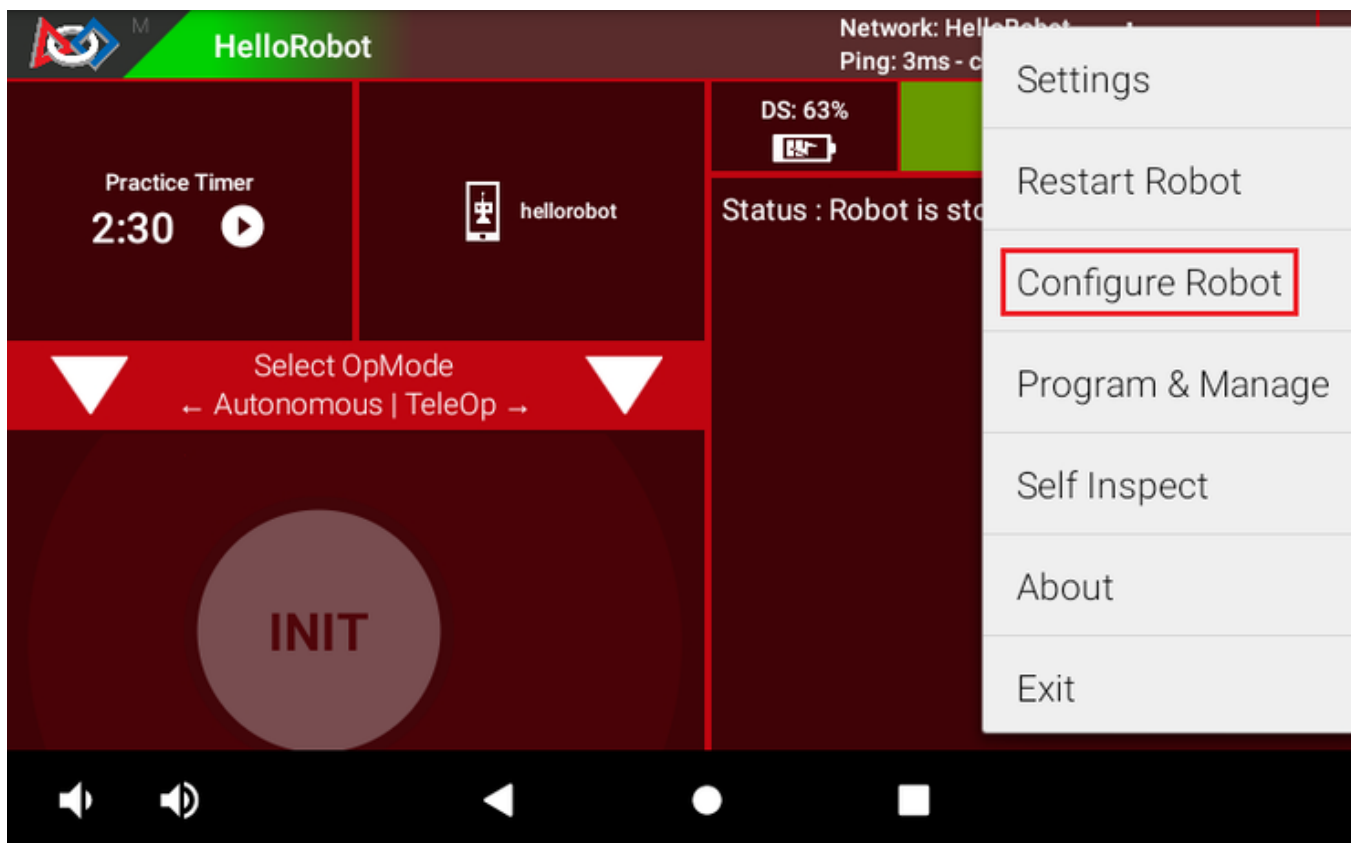
O ID CAN pode ser definido entre 1 e 10 para FTC. Após escolher o ID, clique em "Definir ID CAN".

As Expansions Hub têm como padrão a ID 1 ou 2.

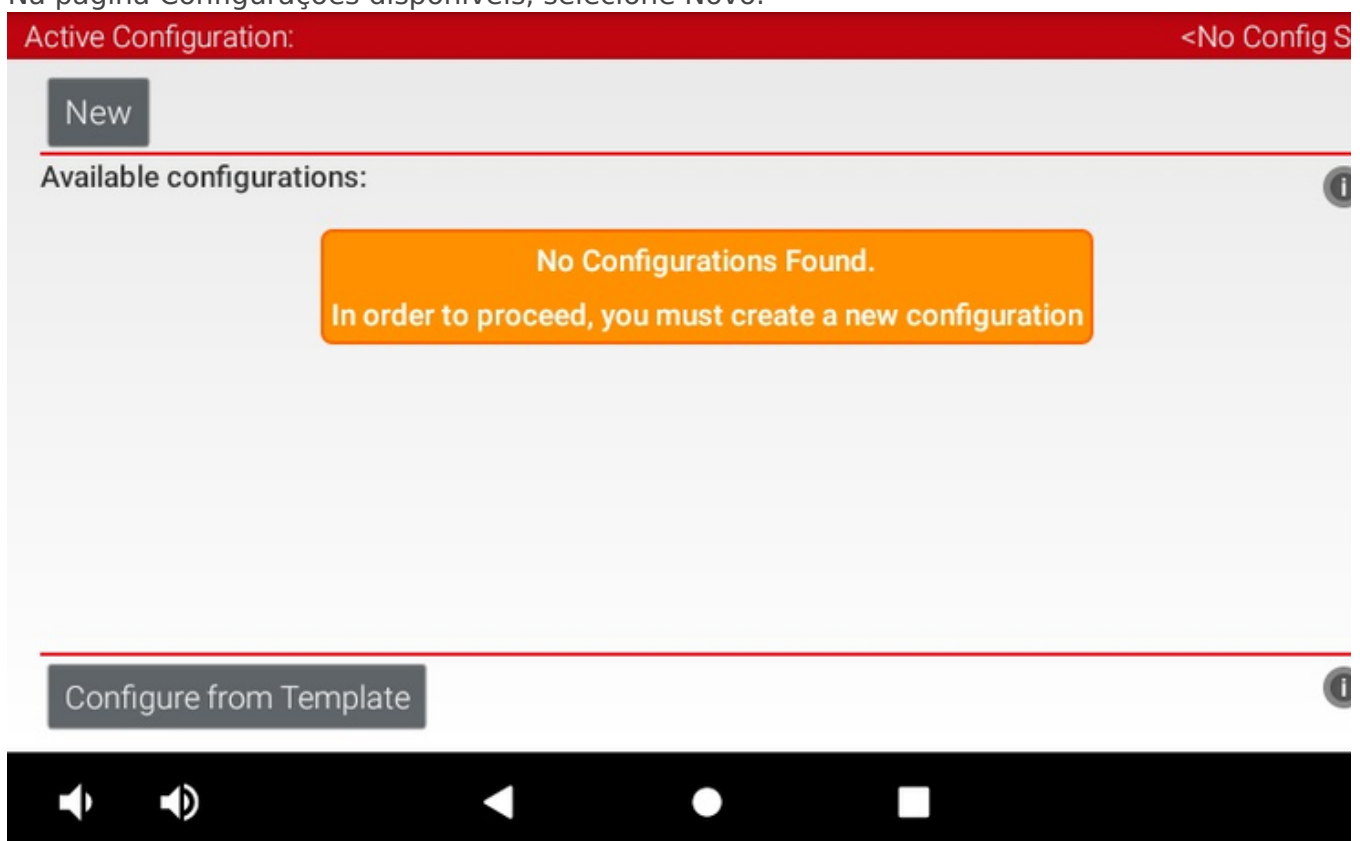


Acesso ao Utilitário de Configuração

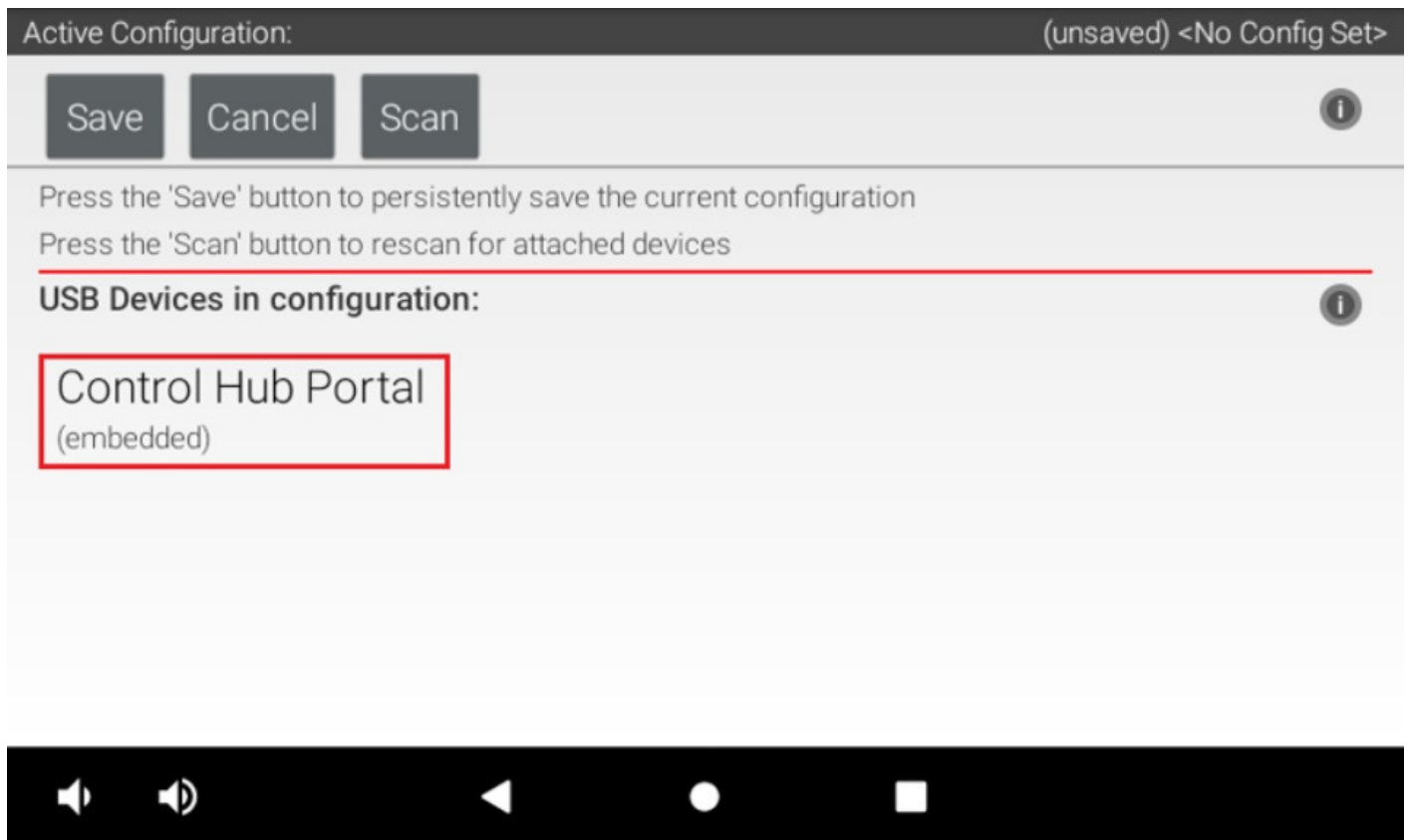
1. Selecione o menu no canto superior direito do aplicativo Driver Station. Em seguida, selecione Configurar robô.



2. Na página Configurações disponíveis, selecione Novo.



3. Na página Dispositivos USB na configuração, selecione o Control Hub Portal. Observação: Se você tiver uma Expansion Hub conectado via USB, ele aparecerá como um Expansion Hub Portal.



Pressionar “Scan” em uma configuração existente pode fazer com que os dispositivos já nomeados sejam apagados. É necessário um novo arquivo de configuração ao adicionar uma câmera ou uma Expansion Hub por USB.

4. Todos os Servo Hubs ou Expansion Hubs conectados usando RS485 serão exibidos no menu do portal. Se você estiver usando vários Servo Hubs, eles poderão ser identificados pelo número de identificação.

Menu ao usar um único Servo Hub:

Active Configuration: (unsaved) <No Config Set>

Done Cancel


Control Hub Portal

(embedded)

Expansion Hub 2

Servo Hub 3

Control Hub



Menu ao usar vários Servo Hubs:

Active Configuration: (unsaved) <No Config Set>

Done Cancel

Control Hub Portal


(embedded)

Expansion Hub 2

Servo Hub 3

Servo Hub 5

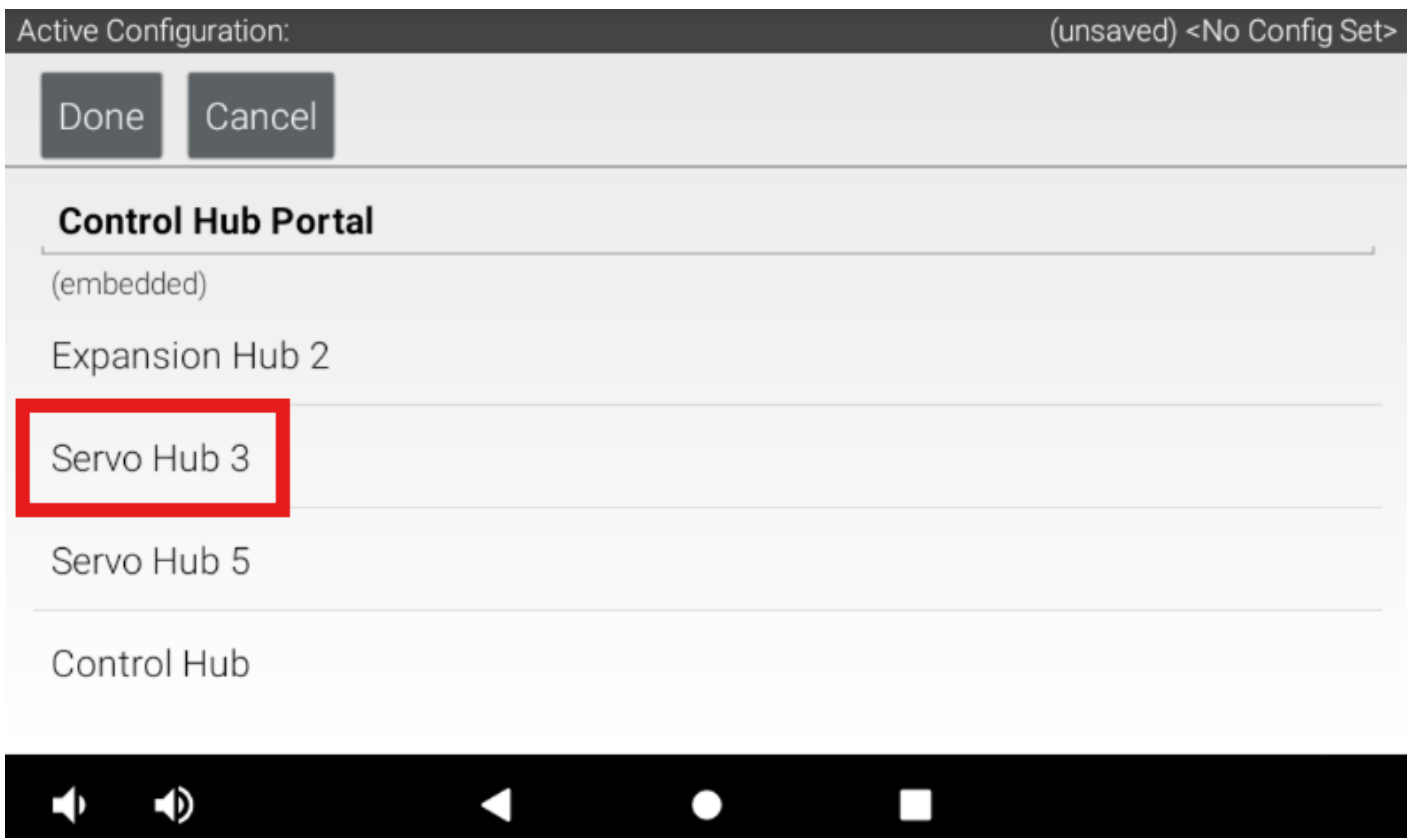
Control Hub



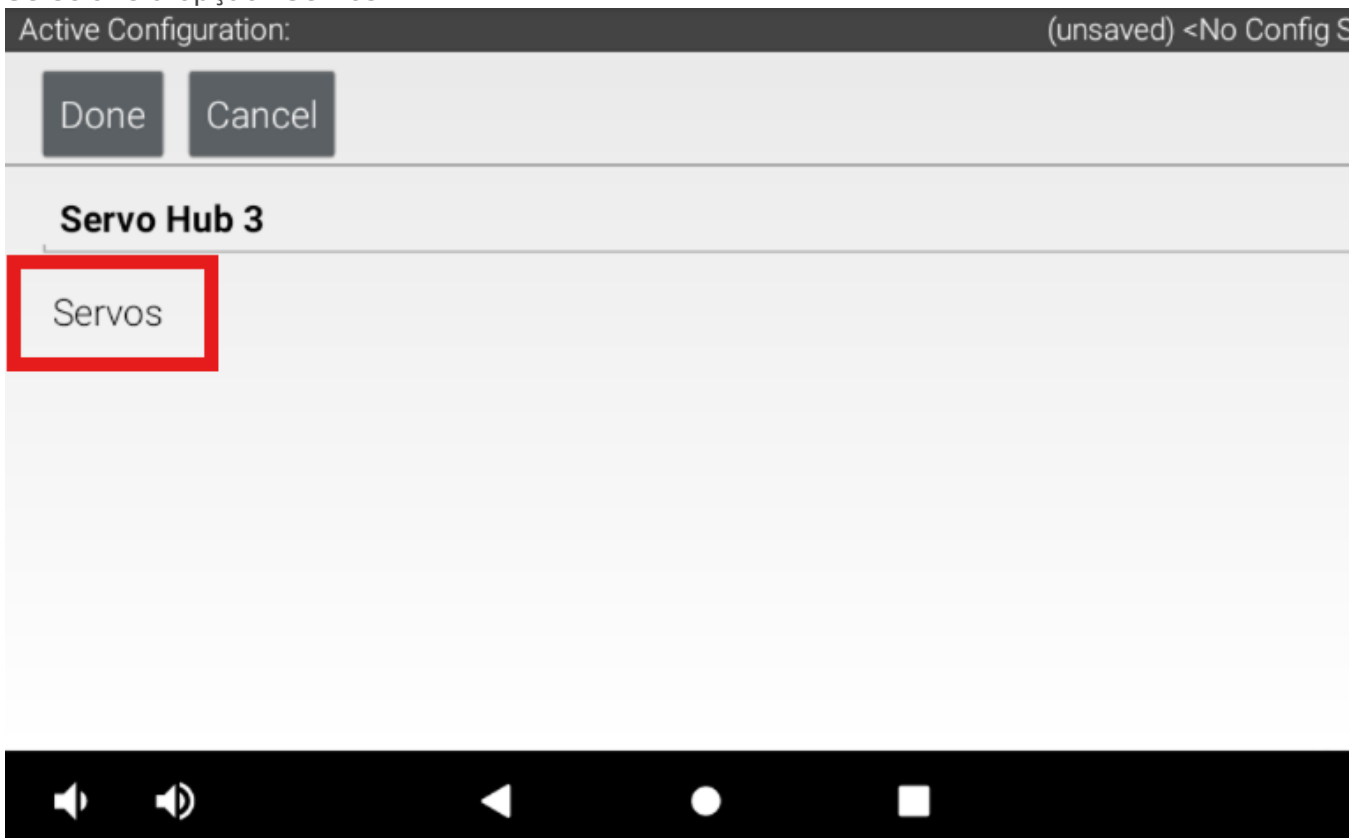
Esse menu aparecerá da mesma forma se o Servo Hub estiver conectado a outro Servo Hub ou a um Hub de expansão conectado usando RS485.

Configurando Servos

1. Selecione o Servo Hub onde você está adicionando servos.



2. Selecione a opção "Servos".



3. Isso abrirá um menu de configuração semelhante ao que é usado para motores e sensores!

Active Configuration: (unsaved) <No Config S

Done Cancel

Port	Attached
0	Nothing
NO DEVICE ATTACHED	
Servo name	
1	Nothing
NO DEVICE ATTACHED	

Speaker icon, Volume icon, Back arrow, Home circle, Recent apps square

4. Selecione a opção desejada no menu suspenso.

Active Configuration: (unsaved) <No Config Set>

Done Cancel

Port	Attached
0	Nothing
Continuous Rotation Servo	
REV Blinkin LED Driver	
1	REV SPARKmini Controller
Servo	

Speaker icon, Volume icon, Back arrow, Home circle, Recent apps square

5. Atribua um nome apropriado ao dispositivo.

Active Configuration: (unsaved) <No Config S

Done Cancel

Port	Attached
0	Servo claw Servo name
1	Nothing

NO DEVICE ATTACHED




6. Clique em “Done” (Concluído) quando todos os nomes forem inseridos para retornar ao menu principal do Servo Hub.

Active Configuration: (unsaved) <No Config S

Done Cancel

Port	Attached
0	Servo claw Servo name
1	Nothing

NO DEVICE ATTACHED



7. Clique novamente em “Done” (Concluído) para retornar à lista de todos os hubs conectados.