The Bowler RPC

A zeroconf protocol for creating PC-microcontroller networks

Ryan Benasutti
Common Wealth Robotics Cooperative

March 2019

1 Overview

1.1 Motivation

- We want a PC to talk to any device which is running some Bowler software. We want this software to require minimal to no configuration from the user.
- We want to support any resource attached to the device with minimal to no configuration from the user.

Contents

2 Requirements

2.1 Project Goals

The goals of this project are as follows, in order of importance:

1. Support real-time PC-device communication with Round Trip Time (RTT) $\leq$ 5ms, timeout $\leq$ 100ms. This requirement is the minimum, implementations ideally have a lower RTT depending on the physical layer used, e.g. an HID-based implementation should tend to be faster and have less jitter than a UDP-based implementation.
2. Support packet groups. For example, two quadrature encoders on opposite sides of a skid-steer drivetrain must be read from at the exact same time; it is not possible to split these two reads into two packets.
3. Optional RPC packet ordering. Some RPC packets (typically writes) need strong ordering. Some physical layers (e.g. UDP) do not guarantee packet ordering.
4. Support dynamic configuration of devices and resources and support dynamically resetting any configured state on the device.
5. Support a wide variety of devices and resources without requiring device-side code modifications.

2.2 Network Requirements

This protocol makes the following assumptions about the network implementation it runs on top of:

- Data corruption is handled. Corrupt packets can be corrected or dropped.
- No packet can be duplicated in the network.

2.3 Configuration Process

The PC-device Remote Prodecure Call (RPC) is established in the following order

1. The PC connects to the device using some physical layer.
2. The PC sends discovery packets to the device to tell the device which resources are connected to it. The device may reject any packet if it deems the resource invalid.
3. Once all resources have been discovered, the discovery process is finished and the PC and device may use the configured RPC.
4. At any time, the PC may tell the device to discard everything which was allocated or configured during discovery. The device must then be ready to start the discovery process again.

3 Discovery

Before the PC can communicate with the device, it must first configure the device using the discovery process. This section describes that process and the packets involved.

3.1 Packet Format

3.1.1 General Discovery Packet Format

Figure 1 shows what the PC sends the device to run discovery. Any additional operation-specific data is sent in the Payload section. The entire packet is 60 bytes. Typically, this protocol is implemented using `SimplePacketComs`, which uses the first 4 bytes of the packet for its header, leaving this protocol with 60 bytes.

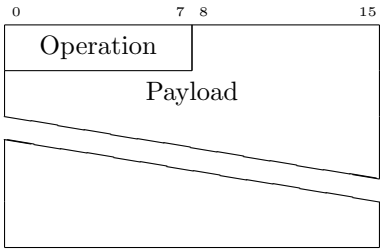


Figure 1: Discovery-time send packet format.

- Operation: 1 byte
  - The Operation field states the operation the packet performs.

Figure 2 shows what the device sends the PC to complete discovery. Any additional operation-specific data is sent in the Payload section. The entire packet is 60 bytes. Typically, this protocol is implemented using `SimplePacketComs`, which uses the first 4 bytes of the packet for its header, leaving this protocol with 60 bytes.

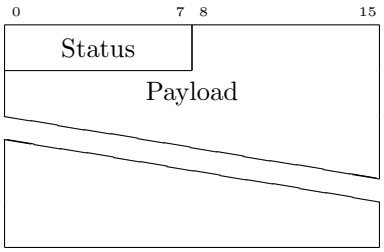


Figure 2: Discovery-time receive packet format.

- Status: 1 byte
  - The Status field encodes the status of the discovery operation. The status codes are documented in Table 1.

3.1.2 Discovery Packet

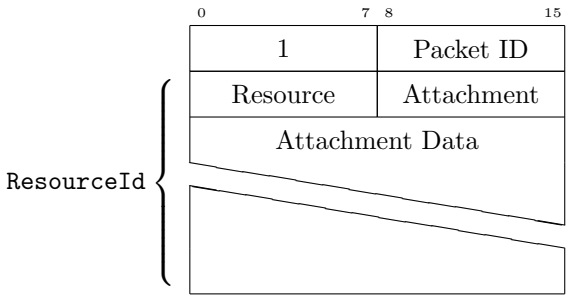


Figure 3: Discovery send packet.

- Packet ID: 1 byte
  - The Packet ID field is a new ID for the packet being discovered.
- Resource: 1 byte
  - The Resource field is the type of the resource. It is the `ResourceId.resourceType.type`.
- Attachment: 1 byte
  - The Attachment field is the type of the attachment point. It is the `ResourceId.attachmentPoint.type`.
- Attachment Data: 1+ bytes
  - The Attachment Data field is any data needed to fully describe the Attachment. It is the `ResourceId.attachmentPoint.data`.

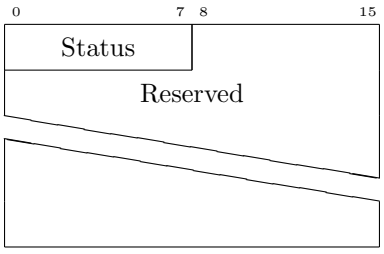


Figure 4: Discovery receive packet.

- Status: 1 byte
  - The Status field encodes the status of the discovery operation. The status codes are documented in Table 1.

3.1.3 Group Discovery Packet

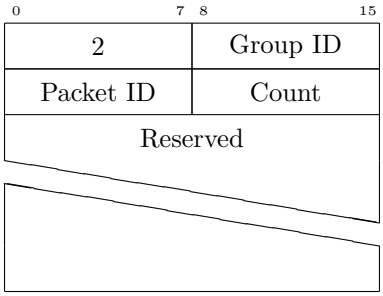


Figure 5: Group discovery send packet.

- Group ID: 1 byte
  - The Group ID field is the ID for the group being made. Future group member discovery packets will need this ID to add resources to the correct group.
- Packet ID: 1 byte
  - The Packet ID field is the ID for the packet the group will use. All resources in the group get packed into one packet.
- Count: 1 byte
  - The Count field is the number of resources that will be added to the group.

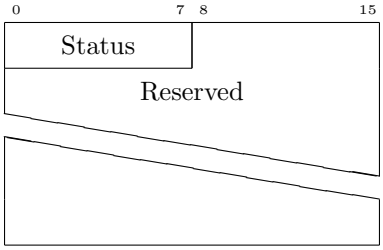


Figure 6: Group discovery receive packet.

- Status: 1 byte
  - The Status field encodes the status of the discovery operation. The status codes are documented in Table 1.

3.1.4 Group Member Discovery Packet

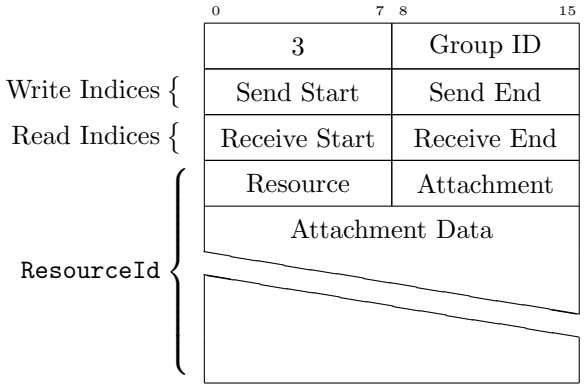


Figure 7: Group member discovery send packet.

- Group ID: 1 byte
  - The Group ID field is the ID for the group that this resource will be added to.
- Send Start: 1 byte
  - The Send Start field is the starting byte index in the send Payload for this resource's write data (i.e. from the PC perspective).
- Send End: 1 byte
  - The Send End field is the ending byte index in the send Payload for this resource's write data (i.e. from the PC perspective).
- Receive Start: 1 byte
  - The Receive Start field is the starting byte index in the receive Payload for this resource's read data (i.e. from the PC perspective).
- Receive End: 1 byte
  - The Receive End field is the ending byte index in the receive Payload for this resource's read data (i.e. from the PC perspective).
- Resource: 1 byte
  - The Resource field is the type of the resource. It is the `ResourceId.resourceType.type`.
- Attachment: 1 byte

  - The Attachment field is the type of the attachment point. It is the `ResourceId.attachmentPoint.type`.

- Attachment Data: 1+ bytes
  - The Attachment Data field is any data needed to fully describe the Attachment. It is the `ResourceId.attachmentPoint.data`.

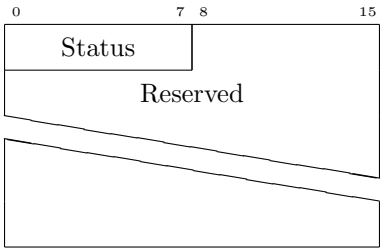


Figure 8: Group member discovery receive packet.

- Status: 1 byte
  - The Status field encodes the status of the discovery operation. The status codes are documented in Table 1.

3.1.5 Discard Discovery Packet

At any time, the PC may tell the device to discard everything which was allocated or configured during discovery. The device must then be ready to start the discovery process again. To initiate this process, the PC may send the device a discard discovery packet and continue polling the device until it reports that the discard process is complete.

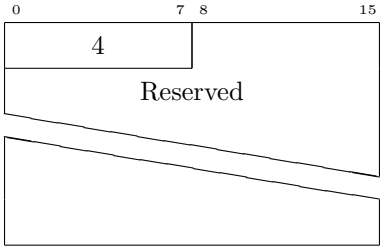


Figure 9: Discard discovery send packet.

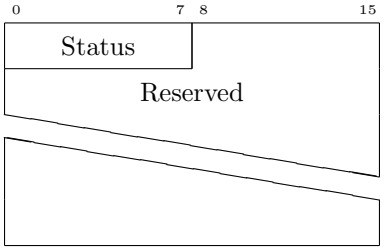


Figure 10: Discard discovery receive packet.

- Status: 1 byte
  - The Status field indicates the status of the discard operation. The status codes are documented in Table 1.

3.2 Discovery Process

3.2.1 Discovery

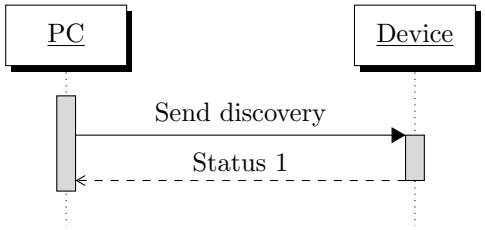


Figure 11: Successful discovery.

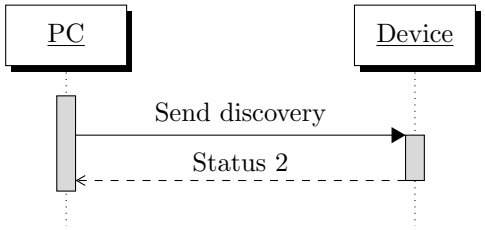


Figure 12: Unsuccessful discovery.

3.2.2 Group Discovery

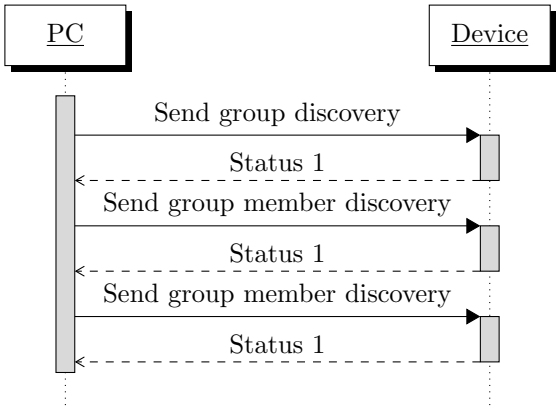


Figure 13: Successful group discovery.

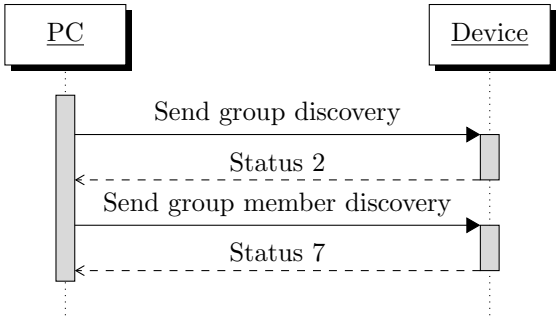


Figure 14: Unsuccessful group discovery.

3.2.3 Discard Discovery

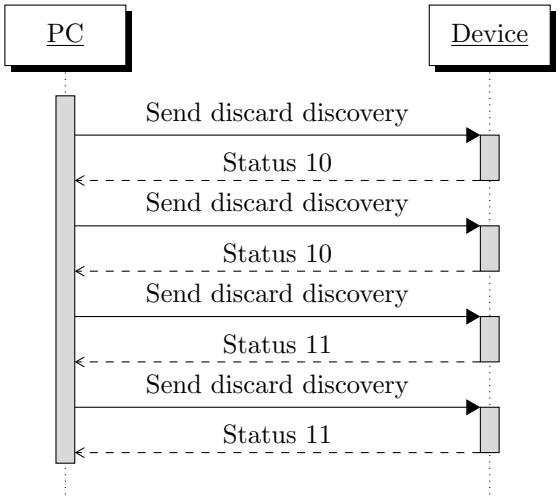


Figure 15: Discard discovery.

4 RPC

After the discovery process is complete and all resources have been discovered, the PC may begin "talking" with the device. This section describes the format of the configured packets.

4.1 Packet Format

4.1.1 Non-Group

Packets for non-group resources correspond to a single resource. These resources do not have any timing constraints.

The non-group send packet format consists of the write payload.

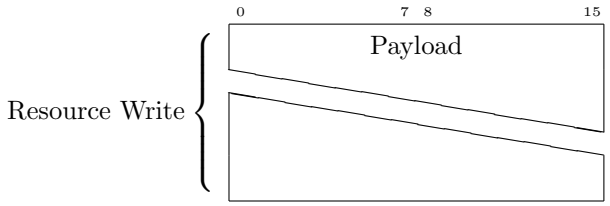


Figure 16: Non-group RPC send packet format.

The non-group receive packet format consists of the read payload.

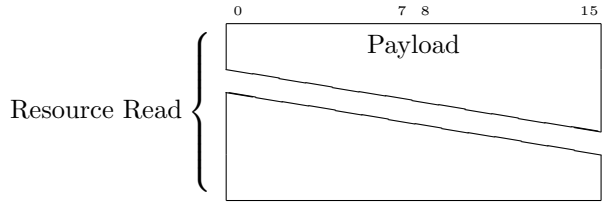
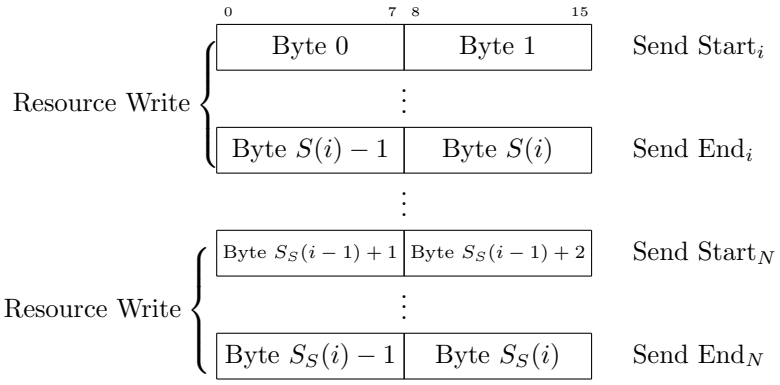


Figure 17: Non-group RPC receive packet format.

4.1.2 Group

Packets for group resources correspond to multiple resources whose write and read payloads are packed into single packets. These resources typically have timing constraints and are therefore put into a group.

The group send packet format consists of packed resource write payloads as specified by the Send Start and Send End indices from the group member discovery packet in Figure 7.



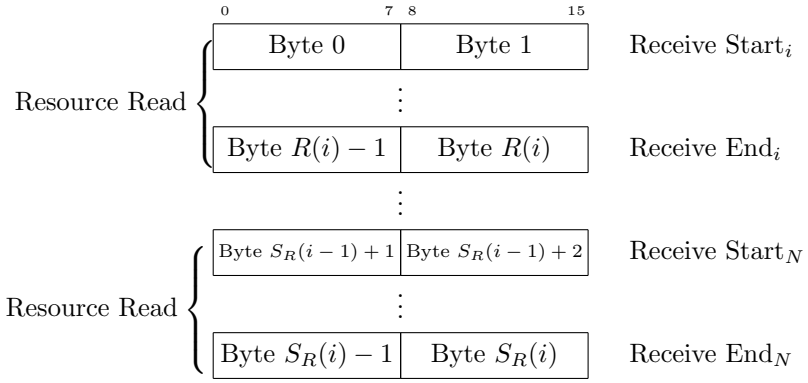
$$S_S(n) = \sum_{i=1}^{n} S(i) \tag{1}$$

$$S(i) = \text{Send End}_i - \text{Send Start}_i \tag{2}$$

$$i \in [1, N], N = \text{Count} \tag{3}$$

Figure 18: Group RPC send packet format.

The group receive packet format consists of packed resource read payloads as specified by the Receive Start and Receive End indices from the group member discovery packet in Figure 7.



$$S_R(n) = \sum_{i=1}^{n} R(i) \tag{4}$$

$$R(i) = \text{Receive End}_i - \text{Receive Start}_i \tag{5}$$

$$i \in [1, N], N = \text{Count} \tag{6}$$

Figure 19: Group RPC receive packet format.

A Tables

Status Code	Meaning
1	Accepted
2	Rejected: Generic
3	Rejected: Unknown Resource
4	Rejected: Unknown Attachment
5	Rejected: Invalid Attachment
6	Rejected: Invalid Attachment Data
7	Rejected: Invalid Group ID
8	Rejected: Group Full
9	Rejected: Unknown Operation
10	Discard in progress
11	Discard complete
12	Rejected: Invalid Packet ID

Table 1: Status codes.

Glossary

device A microcontroller running an implementation of the Bowler RPC. 1, 3, 4, 10, 14, 18

discovery A procedure in which the PC tells the Device which Resources are connected to it. 3, 4, 6, 7, 9, 10, 14

group A collection of Resources which must be written to/read from at the same exact time.. 3, 7, 8, 14, 15, 18

group member discovery A procedure in which the PC tells the Device about a Resource which is part of a Group. 7, 15

operation An action a device can take; typically writing to an actuator or reading from a sensor. 4

PC The host computer running the Bowler stack. 1, 3, 4, 8, 10, 14, 18

resource A hardware element connected to a Device; typically an actuator or sensor. 1, 3, 6, 7, 8, 14, 15, 18

Acronyms

RPC Remote Prodecure Call. 3

RTT Round Trip Time. 3