MotionSystems

INTEGRATION GUIDE

Controlling a Motion Platform

Which way to connect your simulation, program or control system

Revision 1.0 Date 2026-10-20

Motion Systems | motionsystems.eu

Controlling a Motion Platform – Integration Guide

Revision 1.0, 2026-10-20 59c2cde

This guide covers ForceSeatPM 3.1, the ForceSeatMI SDK 3.1 and the ForceSeatDI SDK 3.1.

Other names may be trademarks of their respective owners.

© 2026 Motion Systems. All rights reserved. No part of this document may be reproduced without the written permission of Motion Systems. Information in this document is subject to change without notice.

motionsystems.eu | support.motionsystems.eu

All manuals and guides: https://motionsystems.eu/knowledge-hub

Contents

1 About this guide 3

1.1 Who it is for 3

1.2 What it answers 3

1.3 Other guides 3

2 The short answer 4

2.1 The four ways in 5

3 Choosing the way 6

3.1 Five questions 6

3.2 Slow or sparse commands 6

3.3 Side by side 7

4 The ports of the platform 8

4.1 USB and network 8

4.2 The network connection 8

4.3 What is not available 9

5 Typical setups 10

5.1 Your simulation on another computer 10

5.2 Your program drives the platform itself 11

5.3 Several platforms 11

5.4 Operating, not moving 12

6 Questions customers ask 13

A Revision history 14

CHAPTER 1

About this guide

Before you connect your own simulation, program or control system to a motion platform, you need to know which way in exists. This guide answers that question in a few pages; the manuals it points to describe each way in detail.

1.1 Who it is for

- Engineers who integrate a motion platform into a simulator, a test rig or a research setup.
- Developers who want their own software to move the platform.
- Anyone who has found the network or USB port of the platform and wonders what to send to it.

If you only want to use the platform with the games and simulators that ForceSeatPM supports, you do not need this guide: install ForceSeatPM, activate a profile and play. The ForceSeatPM user guide describes that.

1.2 What it answers

- Which ways exist to control the platform from your own system (Chapter 2).
- Which way fits your project: operating system, send rate, motion cueing, one or several platforms (Chapter 3).
- What the USB and network ports of the platform are for, and what they are not for (Chapter 4).
- How typical setups look (Chapter 5), and the questions customers ask most often (Chapter 6).

1.3 Other guides

All manuals are available at https://motionsystems.eu/knowledge-hub:

- the user manual of your motion platform – installation, safety, the cables and the network settings;
- the ForceSeatPM user guide – the application, its web interface and the Remote Management API;
- the ForceSeatMI manual – your application talks to ForceSeatPM;
- the ForceSeatDI manual – your application talks to the platform's controller directly.

CHAPTER 2

The short answer

Your software never talks to the platform's ports on its own terms. It talks to ForceSeatPM through ForceSeatMI, or to the controller through ForceSeatDI. Both are documented in full; ForceSeatMI is a free download, ForceSeatDI comes from the sales team together with its license.

The ports use our own protocol

The USB and network ports of the platform carry Motion Systems's own control protocol. The protocol is not published: there is no interface description to send commands to the controller yourself, no UDP or TCP command set, no Modbus, CAN or analog input for motion. Use one of the ways in Figure 2.1 instead – they cover what such a protocol would give you, and they handle the controller's safety logic for you.

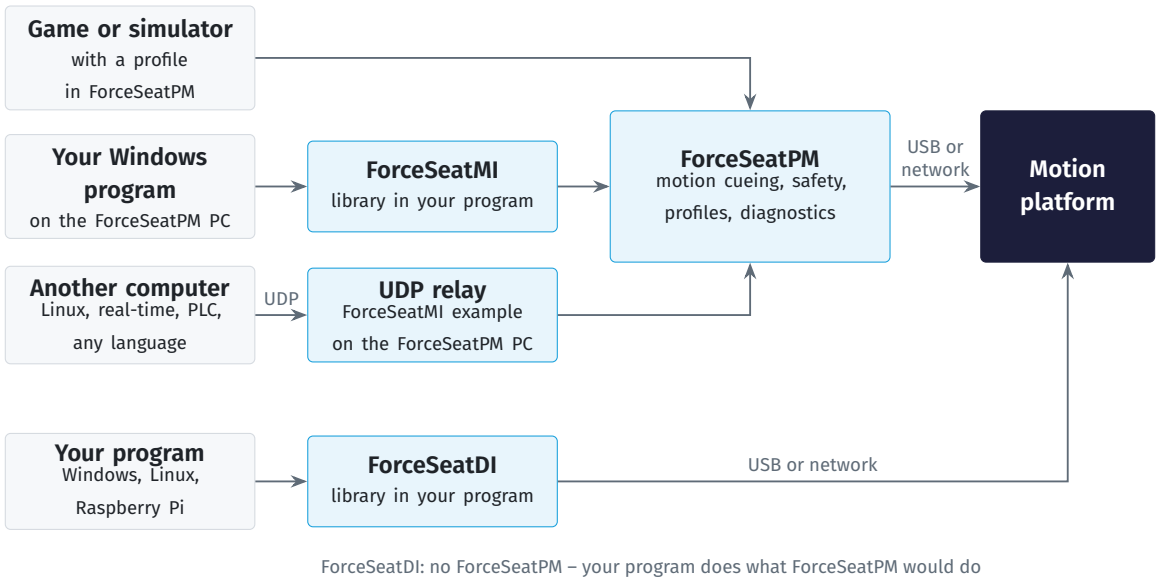

Figure 2.1. The ways to control a motion platform. Everything goes either through ForceSeatPM or through ForceSeatDI.

2.1 The four ways in

Way	When to use it
ForceSeatPM with a profile	A game or simulator that ForceSeatPM supports. Nothing to program.
ForceSeatMI	Your own Windows program on the PC that runs ForceSeatPM: a simulation, a game engine (Unity, Unreal), MATLAB/Simulink, Python. You send telemetry (accelerations, rotation rates; ForceSeatPM turns them into motion) or poses (the platform goes where you say).
ForceSeatMI UDP relay	Your system runs on another computer – Linux, a real-time target, a PLC, an embedded board – or in a language without a ForceSeatMI wrapper. It sends simple UDP frames over the network to a small receiver program on the ForceSeatPM PC. The frame format is published in the ForceSeatMI manual, with senders in C++ and Python.
ForceSeatDI	Your program commands the controller itself, over USB or the network, without ForceSeatPM: on Windows, Linux or a Raspberry Pi; several platforms each with their own commands. Your program takes over the error handling, pausing and parking that ForceSeatPM would do. A license per platform.

Send a steady stream, not single commands

Whichever way you choose, send the current target again and again at a steady rate – also when it does not change. A platform that receives nothing stops: the UDP relay pauses it after 1 s without a frame, and the controller pauses it when a ForceSeatDI target has not changed for 10 s. A system that produces a new target only once a second repeats it in between; Section 3.2 shows how.

CHAPTER 3

Choosing the way

Start from where your software runs and what it knows about the motion. The table at the end of the chapter compares the ways side by side.

3.1 Five questions

1. **Does ForceSeatPM already support your game or simulator?** Then use its profile. Done.
2. **Does your software run on Windows, on the PC that can run ForceSeatPM?** Use ForceSeatMI.
3. **Does it run elsewhere** – another computer, Linux, a real-time system, a PLC? Use the ForceSeatMI UDP relay: your system sends frames over the network, ForceSeatPM on a Windows PC drives the platform. If no Windows PC can be part of the setup, use ForceSeatDI on your system.
4. **Do you know accelerations and rotation rates, or exact positions?** Accelerations and rates of a vehicle (telemetry): ForceSeatMI or the relay, with ForceSeatPM's motion cueing – the platform gives the sensation, and returns to its center so that it does not run out of travel. Exact positions of the top frame (poses): any of the three; the platform goes where you say, within its work envelope.
5. **Must several platforms move differently, or must nothing but your program run?** ForceSeatDI. (With ForceSeatPM all configured platforms get the same motion.)

3.2 Slow or sparse commands

Some control systems produce a new target only now and then – once a second, or when an operator presses a button. The platform still needs a steady stream:

- **Through ForceSeatPM** (ForceSeatMI or the relay): send the current target again and again at a steady rate – 50 Hz or more; the relay pauses the platform when no frame arrives for 1 s. Choose the **SDK - Positioning with Trajectory** profile: it moves to each new target with limited speed and acceleration, so a target that jumps once a second gives a smooth move instead of a jerk. The ForceSeatMI manual describes its limits and a GoTo helper for moves of this kind.
- **With ForceSeatDI**: send at 100 to 250 Hz and limit the speed of the move in the command. The controller pauses the platform when the target has not changed for 10 s, and continues as soon as a new one arrives.

3.3 Side by side

	ForceSeatMI	ForceSeatMI UDP relay	ForceSeatDI
Your software runs on	The ForceSeatPM PC (Windows)	Any computer in the network	Windows, Linux, Raspberry Pi
ForceSeatPM	Required	Required, on a Windows PC	Not used
You send	Telemetry or poses	Telemetry or poses, as UDP frames	Poses, a matrix or actuator positions
Motion cueing, washout	Yes (ForceSeatPM)	Yes (ForceSeatPM)	No – your program
Safety, errors, pause, park	ForceSeatPM	ForceSeatPM; the receiver pauses when frames stop	Your program
Several platforms	All move the same	All move the same	Each on its own
To the platform	USB or network, set in ForceSeatPM	USB or network, set in ForceSeatPM	USB or network, from your program
Languages	C, C++, C#, Python, MATLAB/Simulink, Unity, Unreal	Anything that sends UDP	C, C++, C#, Python, MATLAB/Simulink, Unity, Unreal
License	ForceSeatMI SDK, per platform (or an AppID for your application)	As ForceSeatMI	ForceSeatDI, per platform, stored in its controller

Ask the Motion Systems sales team for the license of the way you choose. Your platform's dealer or the sales team also help when you are not sure which way fits.

CHAPTER 4

The ports of the platform

What each connection of the platform's controller is for, and who may use it.

4.1 USB and network

Platforms equipped with a **PS-SB230** or **PS-SB400** power cabinet have two ports on the back of the cabinet: **USB**, the default connection to the PC, and **Ethernet**, an optional network connection. The network option – **Connection type Network** in ForceSeatPM, or ForceSeatDI over the network – requires one of these cabinets. Platforms and devices whose controller is built in connect over USB only. The model of the power cabinet is on its rating plate and in the platform's user manual.

Port	Who uses it	Not for
USB	ForceSeatPM (default), or a program with ForceSeatDI	Commands of your own: the protocol is not published
Ethernet	ForceSeatPM (**Connection type Network**), or a program with ForceSeatDI over the network	Commands of your own, Modbus, other protocols

> **One program at a time**
>
> Only one program may command the controller. Close ForceSeatPM before a ForceSeatDI program connects to the same platform, and never connect one platform over USB and network at the same time.

4.2 The network connection

The network connection replaces the USB cable; it does not open the platform to other programs.

- Every power cabinet leaves the factory with the same address, 10.1.1.75 (mask 255.255.255.0). Change it over USB in ForceSeatPM when several platforms share a network. The address is static; there is no DHCP and no discovery.
- Connect the cabinet directly to a network card of the PC, or through a dedicated, wired network. A wireless network is not recommended.

> **No authentication**
>
> Anything that can reach the cabinet's port can command the platform. Never connect it to an office network or the Internet, and never forward its ports on a router.

The platform's user manual describes the network setup step by step: the address of the cabinet, the network card of the PC and the settings in ForceSeatPM. The ForceSeatDI manual describes the network connection from your program.

4.3 What is not available

- A published protocol, an interface control document or a register map of the controller.
- Commands sent to the controller by your own program without ForceSeatDI – over USB or Ethernet.
- Industrial fieldbuses (Modbus, CAN, EtherCAT, PROFINET) or analog inputs for motion.
- Changes to the firmware of the controller.

Each of these is covered by a way in Chapter 2: ForceSeatDI gives your program the controller, the ForceSeatMI UDP relay gives any computer a simple, published UDP format.

CHAPTER 5

Typical setups

Three setups that cover most projects. Each names the manual and the example to start from.

5.1 Your simulation on another computer

A simulation on Linux, a real-time target or a PLC computes the motion; a Windows PC next to the platform runs ForceSeatPM. The simulation sends UDP frames over the network; the receiver of the ForceSeatMI UDP relay example passes them to ForceSeatPM.

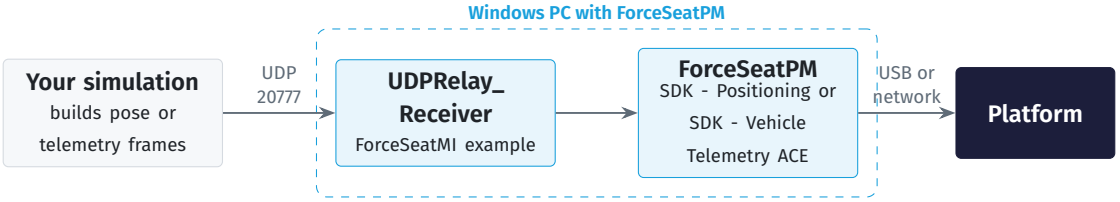

Figure 5.1. The UDP relay: one small receiver program on the ForceSeatPM PC, any sender in the network.

- One frame per datagram: a pose (roll, pitch, yaw, heave, sway, surge) or telemetry (accelerations, rotation rates and more), with a sequence number and a pause flag. The ForceSeatMI manual shows the frame byte by byte.
- Senders in C++ (Windows and Linux) and Python come with the example; any language that sends UDP will do.
- Send at a steady rate, also when the target does not change (Section 3.2). Accept frames from your simulation only (receiver option `--from`).
- Start with: ForceSeatMI manual, “Relay from another computer”; example `UDPRelay_CPP`.

5.2 Your program drives the platform itself

A test bench, a laboratory rig or a headless box replays poses with ForceSeatDI. ForceSeatPM is not running; your program connects to the controller over USB or the network.

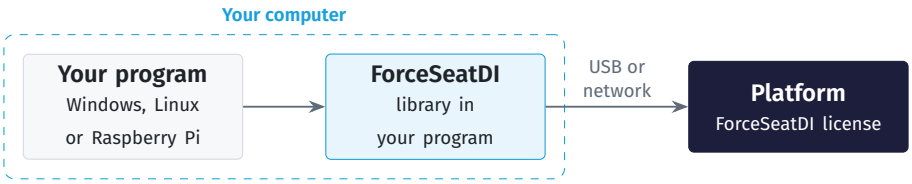

Figure 5.2. ForceSeatDI: your program talks to the controller; no ForceSeatPM at run time.

- Your program sends poses, a transformation matrix or actuator positions at a steady rate (100 to 250 Hz) and checks the platform's status itself.
- The license is stored in the platform's controller; ForceSeatPM is needed once, to enter the activation code.
- Start with: ForceSeatDI manual; the examples for your language.

5.3 Several platforms

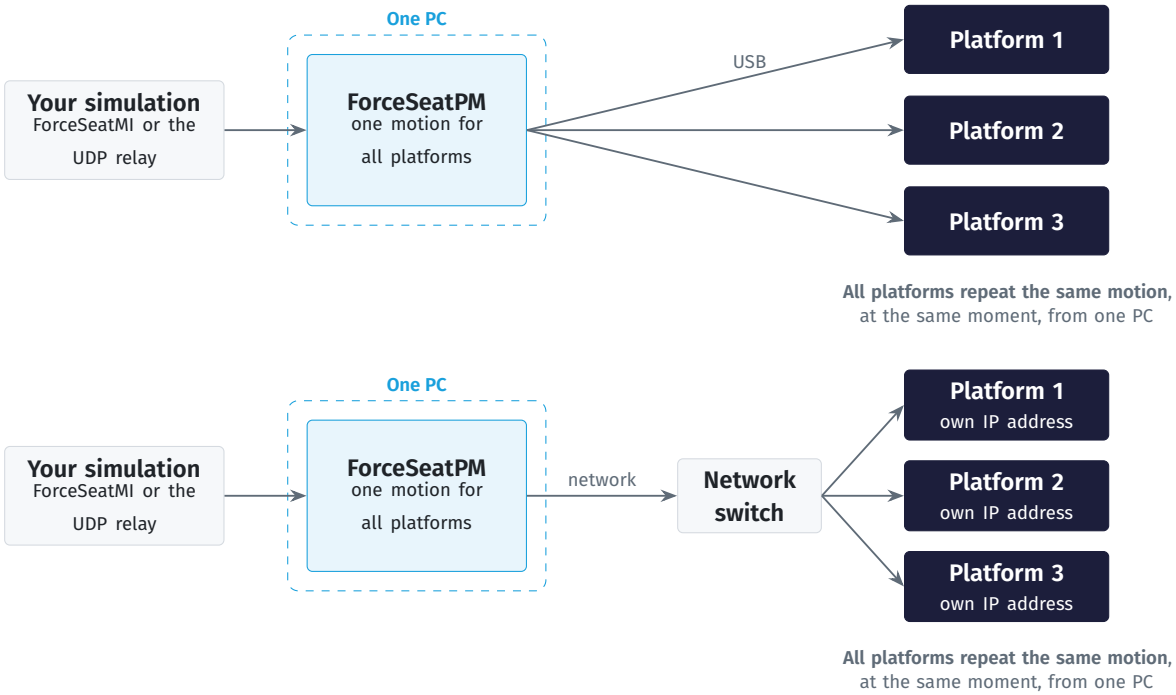

Figure 5.3. Several platforms on one ForceSeatPM PC: all on USB (top) or all on the network (bottom). Every platform makes the same motion. To move platforms differently, use ForceSeatDI.

Connect all platforms the same way: all on USB, or all on the network. On USB, ForceSeatPM drives every motion platform it finds; on the network (**Connection type Network**), the platforms whose IP addresses are listed under **Software configuration** › **Motion platform**. ForceSeatPM needs every listed platform switched on and running.

- **The same motion on every platform** – a row of identical simulators driven by one simulation: ForceSeatPM with several platforms connected, fed by ForceSeatMI or the UDP relay (Figure 5.3).
- **Different motion on each platform** – a choreography, platforms that interact: ForceSeatDI, one connection per platform.

5.4 Operating, not moving

To switch profiles, pause, park or start games from a booking or operator system – without sending motion – use the Remote Management API of ForceSeatPM; to operate from a phone or tablet, its web interface. Both are described in the ForceSeatPM user guide. Neither moves the platform freely.

CHAPTER 6

Questions customers ask

Question	Answer
The cabinet has an Ethernet port. What do I send to it?	Nothing of your own: the port carries Motion Systems's protocol for ForceSeatPM and ForceSeatDI. To control the platform over the network from your system, use the UDP relay (Section 5.1) or ForceSeatDI over the network (Section 5.2).
Can I get the protocol or an interface control document?	No, the controller's protocol is not published. ForceSeatDI and the UDP relay give you the same control, with a documented interface.
I need a UDP example to start from.	The ForceSeatMI example `UDPRelay_CPP`: a receiver for the ForceSeatPM PC and senders in C++ and Python (Section 5.1).
Our system produces a target once per second.	Send it again at a steady rate until the next one, and use the **SDK - Positioning with Trajectory** profile for smooth moves between targets (Section 3.2).
Can I control the platform without Windows?	Yes, with ForceSeatDI on Linux or a Raspberry Pi. With the UDP relay your system can run anywhere, but one Windows PC runs ForceSeatPM.
Does ForceSeatPM add motion cueing to my data?	Yes, when you send telemetry (accelerations and rotation rates). Poses are passed on as they are, limited to what the platform can do. ForceSeatDI has no motion cueing.
Can I control the platform over the Internet?	No. Keep the platform, the ForceSeatPM PC and your system in one local network; to reach it from outside, connect to that network through a VPN.
Which license do I need?	ForceSeatMI and the UDP relay: the ForceSeatMI SDK license of the platform. ForceSeatDI: the ForceSeatDI license of the platform. Ask the sales team.

APPENDIX A

Revision history

Revision	Date	Changes
1.0 59c2cde	2026-10-20	First release: the ways to control a motion platform, how to choose, the USB and network ports, typical setups, frequent questions.