Here's a **complete working setup for RHEL** with all the files and the exact commands you need to build and run your Python extension module using **Qt + pybind11**.

📂 **Project Layout**

~/Downloads/Binding/

├── CMakeLists.txt

├── serialmanager.h

├── serialmanager.cpp

├── bindings.cpp

└── test.py

📄 **Files**

1. CMakeLists.txt

```
cmake_minimum_required(VERSION 3.18)
project(serial_module)
# Add pybind11 from the cloned repo
add_subdirectory(pybind11)
find_package(Qt5Core REQUIRED)
pybind11_add_module(serial_module bindings.cpp serialmanager.cpp)
target_link_libraries(serial_module PRIVATE Qt5::Core)
```

---

2. serialmanager.h

```
#ifndef SERIALMANAGER_H
#define SERIALMANAGER_H
#include <QString>
#include <QByteArray>
class SerialManager {
public:
    SerialManager() {}
    int checksum(const QByteArray &data, int length);
```

```
    int stepcmd(unsigned char chanNum, unsigned char *stepval);
    double act(int value);
    int relayO(char chnum);
    int relayO(QString str);
    int setIp(int a, int b);
    int STABCmd();
    int init(QString portName);
    int relayf(char chanNum, QByteArray data);
};
#endif
```

---

3. serialmanager.cpp

```
#include "serialmanager.h"
int SerialManager::checksum(const QByteArray &data, int length) {
    int sum = 0;
    for (int i = 0; i < length && i < data.size(); ++i) {
        sum += static_cast<unsigned char>(data[i]);
    }
    return sum;
}
int SerialManager::stepcmd(unsigned char chanNum, unsigned char *stepval) {
    return chanNum + (stepval ? *stepval : 0);
}
double SerialManager::act(int value) {
    return value * 1.5; // demo logic
}
int SerialManager::relayO(char chnum) {
    return static_cast<int>(chnum);
}
int SerialManager::relayO(QString str) {
```

```
    return str.length();
}
int SerialManager::setIp(int a, int b) {
    return a + b;
}
int SerialManager::STABCmd() {
    return 42;
}
int SerialManager::init(QString portName) {
    return portName.length();
}
int SerialManager::relayf(char chanNum, QByteArray data) {
    return chanNum + data.size();
}
```

---

4. bindings.cpp

```
#include <pybind11/pybind11.h>
#include <pybind11/stl.h>
#include <QString>
#include <QByteArray>
#include "serialmanager.h"
namespace py = pybind11;
PYBIND11_MODULE(serial_module, m) {
    py::class_<SerialManager>(m, "SerialManager")
        .def(py::init<>())
        .def("checksum", [](SerialManager &self, py::bytes pydata, int length) {
            std::string s = pydata;
            QByteArray qdata(s.data(), s.size());
            return self.checksum(qdata, length);
        })
```

```
    .def("stepcmd", [](SerialManager &self, unsigned char chanNum, std::vector<unsigned
char> stepval) {
        return self.stepcmd(chanNum, stepval.data());
    })
    .def("act", &SerialManager::act)
    .def("relayO_char", static_cast<int (SerialManager::*)(char)>(&SerialManager::relayO))
    .def("setIp", &SerialManager::setIp)
    .def("STABCmd", &SerialManager::STABCmd)
    .def("init", [](SerialManager &self, const std::string &portName) {
        QString qport = QString::fromStdString(portName);
        return self.init(qport);
    })
    .def("relayO_str", [](SerialManager &self, const std::string &pystr) {
        QString qstr = QString::fromStdString(pystr);
        return self.relayO(qstr);
    })
    .def("relayf", [](SerialManager &self, char chanNum, py::bytes pydata) {
        std::string s = pydata;
        QByteArray qdata(s.data(), s.size());
        return self.relayf(chanNum, qdata);
    });
}
```

---

5. test.py

```
import serial_module
mgr = serial_module.SerialManager()
print("Checksum:", mgr.checksum(b"\x01\x02\x03", 3))
print("Stepcmd:", mgr.stepcmd(5, [0x10]))
print("Act:", mgr.act(42))
print("RelayO(char):", mgr.relayO_char('A'))
```

```
print("SetIp:", mgr.setIp(10, 200))
print("STABCmd:", mgr.STABCmd())
print("Init:", mgr.init("COM3"))
print("RelayO(QString):", mgr.relayO_str("Relay1"))
print("Relayf:", mgr.relayf('B', b"\x01\x02"))
```

Commands for RHEL

```
# 1. Install dependencies
sudo dnf groupinstall "Development Tools"
sudo dnf install cmake gcc-c++ python3-devel qt5-qtbase-devel

# 2. Clone pybind11 inside project
cd ~/Downloads/Binding
git clone https://github.com/pybind/pybind11.git

# 3. Build
rm -rf build
mkdir build && cd build
cmake ..
make

# 4. Copy module next to test.py
cp serial_module*.so ..
cd ..

# 5. Run Python test
python3 test.py
```
