# Getting Started with UHD and C++

Application Note Number: AN-5501

Authors: John Smith and Jane Smith

Last Modified Date: 2016/04/15

This Application Note will walk through building a basic C++ program with UHD. This program will initialize, configure the USRP device, set the sample rate, frequency, gain, bandwidth, and select the antenna.

Example code:

```
#include <uhd/utils/thread_priority.hpp>
#include <uhd/utils/safe_main.hpp>
#include <uhd/usrp/multi_usrp.hpp>
#include <uhd/exception.hpp>
#include <uhd/types/tune_request.hpp>
#include <boost/program_options.hpp>
#include <boost/format.hpp>
#include <boost/thread.hpp>
#include <iostream>

int UHD_SAFE_MAIN(int argc, char *argv[]) {
    uhd::set_thread_priority_safe();

    std::string device_args("addr=192.168.10.2");
    std::string subdev("A:0");
    std::string ant("TX/RX");
    std::string ref("internal");

    double rate(1e6);
    double freq(915e6);
    double gain(10);
    double bw(1e6);

    //create a usrp device
    std::cout << std::endl;
    std::cout << boost::format("Creating the usrp device with: %s...") % device_args << std::endl;
    uhd::usrp::multi_usrp::sptr usrp = uhd::usrp::multi_usrp::make(device_args);

    // Lock mboard clocks
    std::cout << boost::format("Lock mboard clocks: %f") % ref << std::endl;
    usrp->set_clock_source(ref);

    //always select the subdevice first, the channel mapping affects the other settings
    std::cout << boost::format("subdev set to: %f") % subdev << std::endl;
    usrp->set_rx_subdev_spec(subdev);
    std::cout << boost::format("Using Device: %s") % usrp->get_pp_string() << std::endl;

    //set the sample rate
    if (rate <= 0.0) {
        std::cerr << "Please specify a valid sample rate" << std::endl;
        return ~0;
    }

    // set sample rate
    std::cout << boost::format("Setting RX Rate: %f Msps...") % (rate / 1e6) << std::endl;
    usrp->set_rx_rate(rate);
    std::cout << boost::format("Actual RX Rate: %f Msps...") % (usrp->get_rx_rate() / 1e6) << std::endl << std::endl;

    // set freq
    std::cout << boost::format("Setting RX Freq: %f MHz...") % (freq / 1e6) << std::endl;
    uhd::tune_request_t tune_request(freq);
    usrp->set_rx_freq(tune_request);
    std::cout << boost::format("Actual RX Freq: %f MHz...") % (usrp->get_rx_freq() / 1e6) << std::endl << std::endl;

    // set the rf gain
    std::cout << boost::format("Setting RX Gain: %f dB...") % gain << std::endl;
    usrp->set_rx_gain(gain);
    std::cout << boost::format("Actual RX Gain: %f dB...") % usrp->get_rx_gain() << std::endl << std::endl;

    // set the IF filter bandwidth
    std::cout << boost::format("Setting RX Bandwidth: %f MHz...") % (bw / 1e6) << std::endl;
    usrp->set_rx_bandwidth(bw);
    std::cout << boost::format("Actual RX Bandwidth: %f MHz...") % (usrp->get_rx_bandwidth() / 1e6) << std::endl << std::endl;

    // set the antenna
    std::cout << boost::format("Setting RX Antenna: %s") % ant << std::endl;
    usrp->set_rx_antenna(ant);
    std::cout << boost::format("Actual RX Antenna: %s") % usrp->get_rx_antenna() << std::endl << std::endl;

    return EXIT_SUCCESS;
}
```

Use the uhd/host/examples/init_usrp/CMakeLists.txt file as template - Add the names of your C++ source files to the add_executable(...) section - Put both modified CMakeLists.txt file and C++ file into an empty folder - Create a ?build? folder and invoke CMake the usual way:

```
mkdir build
cd build
cmake ../
make -j4
```