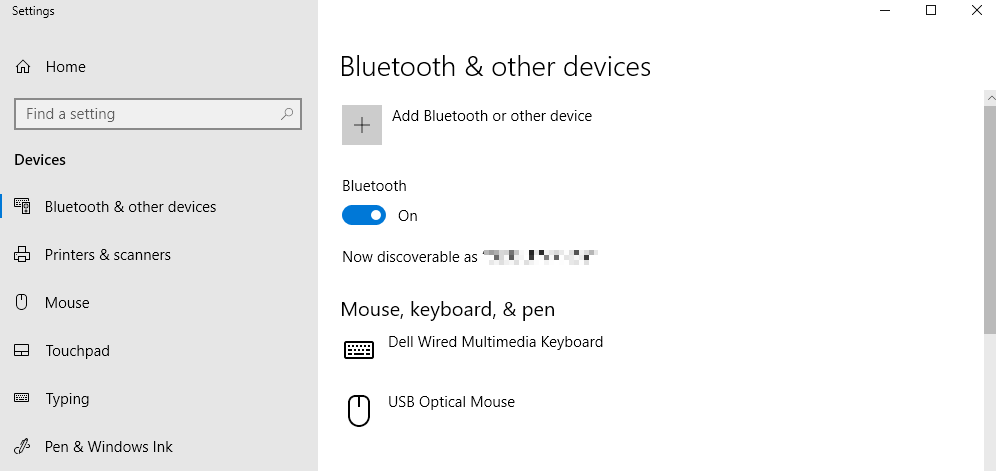
PR3 Csim

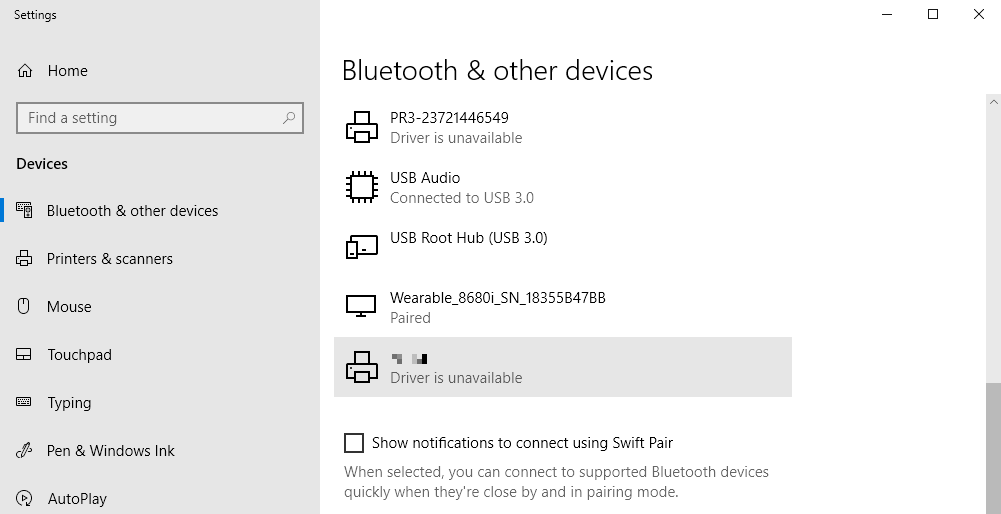
Get version information

# Windows 10

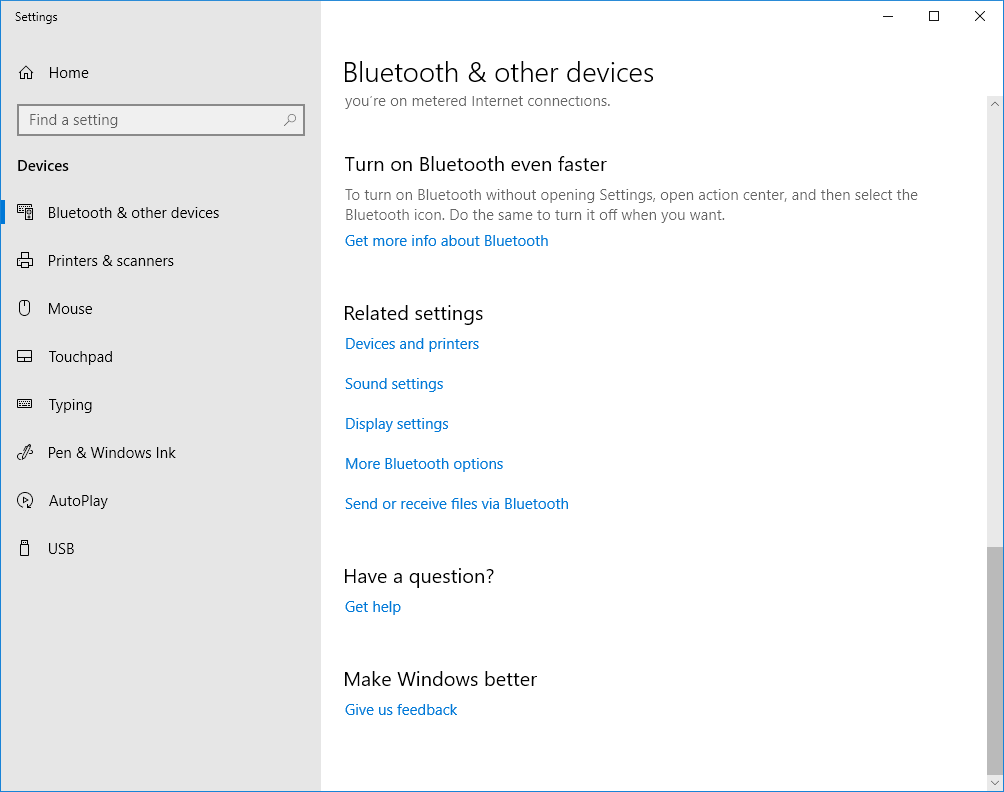
Within control panel “Bluetooth & other Devices” connect to the PR3:



The PR3 is connected:

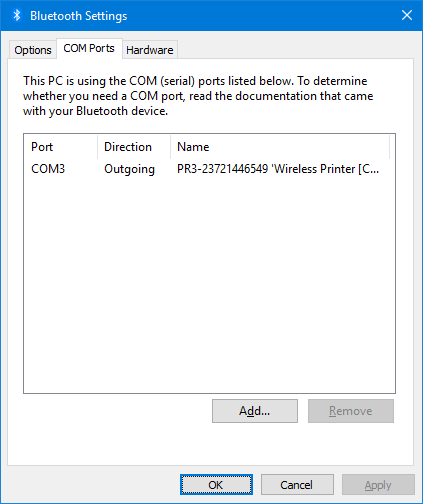


Locate the COM port of the connection or assign an outgoing COM port



Click More Bluetooth options

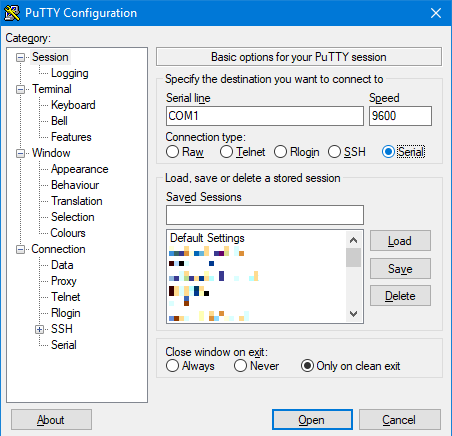
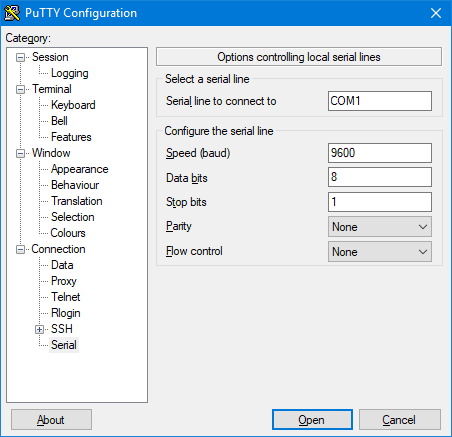
Ensure you have an outgoing COM port assigned to the PR3



# Connect from a terminal app to the PR3

Example using puTTY

Start puTTY and set the connection to serial with no handshake and enter the correct COM port, for example COM3.

Click [Open] to connect to the printer:

Without no local echo, you will not see the characters you enter. Now enter

!U  
VERSION  
PRINT

Or, simply:

!U1  
VERSION

Press Enter key at end of each line. Now puTTY should show:



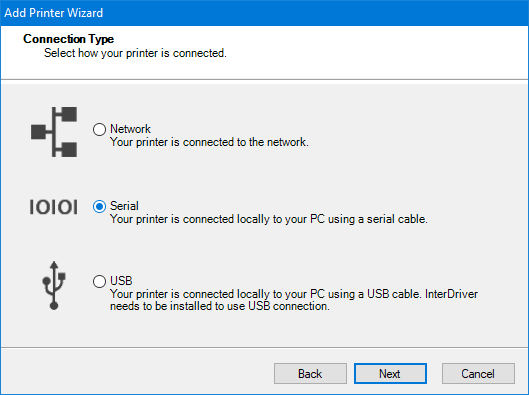
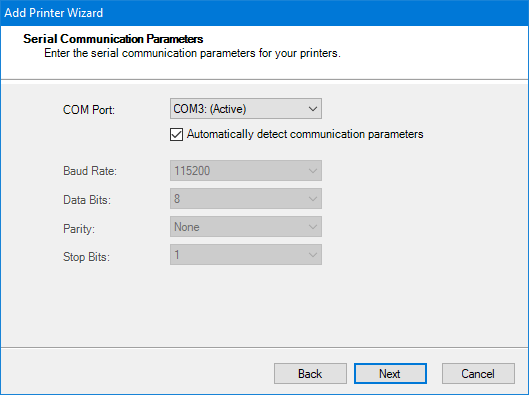
Which is the Csim version as “Csim 1.7.10”.

If the printer prints the text or just responses nothing, ensure it is set to Csim emulation.

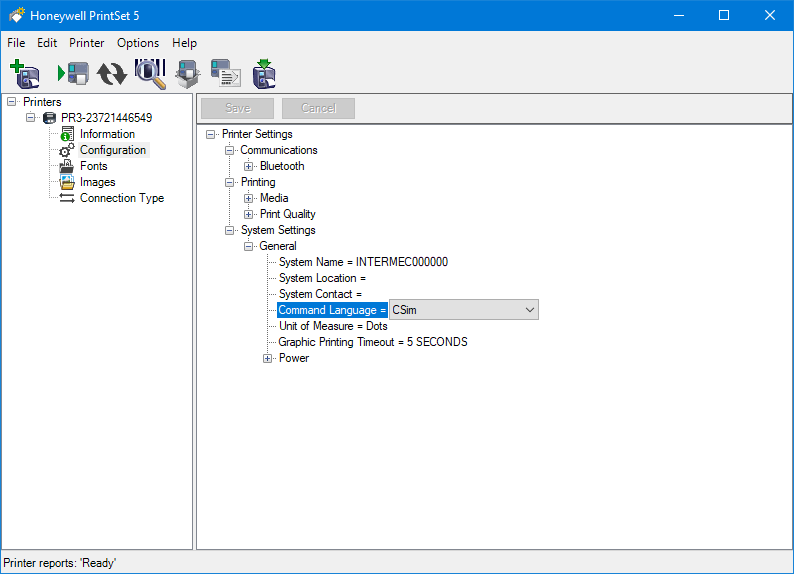
# Set PR3 emulation to Csim

Download, install and start PrintsSet 5:

If not done, add the PR3. The printer should be found automatically on the connected Bluetooth virtual port (COM3).

Click Configuration and ensure “Command Language” is “Csim” or change this to “Csim” and let the printer reboot:



# Get Csim version from Android code using a Bluetooth connection

The “honeywell-android-printing-sdk” does not provide a back channel API to receive printer repsonses within code. You need to write an application that opens a bluetooth socket to the printer and provides the Input and Output Stream for the connection.

### Code sample:

## void connectAndSend(final String sMacAddr, final byte[] sendBytes){ BluetoothDevice device; try { // Get local Bluetooth adapter mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter(); device = mBluetoothAdapter.getRemoteDevice(sMacAddr); mainActivity.addLog("BT device found"); }catch (Exception e){ Toast.makeText(m\_context,"Invalid BT MAC address", Toast.LENGTH\_LONG); mainActivity.addLog("Failed to get BT device by Mac address!"); device=null; } byte[] buf=new byte[200]; int maxTry=10; int iTry=0; if (device != null) { addLog("connecting to " + sMacAddr); mainActivity.addLog("connecting to " + sMacAddr + " ..."); try { addLog("createInsecureRfcommSocketToServiceRecord"); mSocket = device.createInsecureRfcommSocketToServiceRecord(UUID\_SPP); //tmp = device.createRfcommSocketToServiceRecord(UUID\_SPP); mSocket.connect(); mainActivity.addLog("BT socket connected"); // Get the BluetoothSocket input and output streams try { mInStream = mSocket.getInputStream(); mOutStream = mSocket.getOutputStream(); mOutStream.write(sendBytes); mOutStream.flush(); Log.d(TAG, "write done"); mainActivity.addLog("CSim version requested"); String sIn = ""; do { int iCnt = mInStream.read(buf); String s = new String(buf, 0, iCnt); Log.d(TAG, "received: " + s); sIn += s; if (sIn.endsWith("\r\n")) { Log.d(TAG, "receive complete: "+ sIn); mainActivity.addLog("Received: "+ sIn); break; } iTry++; Thread.sleep(500); }while(iTry<maxTry); mInStream.close(); mOutStream.close(); mSocket.close(); mainActivity.addLog("BT socket closed"); } catch (IOException e) { Log.e(TAG, "temp sockets not created", e); mainActivity.addLog("Failed to connect to BT socket!"); } } catch (IOException e) { Log.e(TAG, "create() failed", e); mainActivity.addLog("socket create failed!"); } catch (InterruptedException e) { Log.e(TAG, "create() failed", e); mainActivity.addLog("socket stream interrupted!"); } // This is a blocking call and will only return on a // successful connection or an exception } else { addLog("unknown remote device!"); mainActivity.addLog("unknown remote device!"); } }

## Notes

* you need to wait for the answer ending with \r\n.
* do not send other data before the answer has been received and the printer is ready.