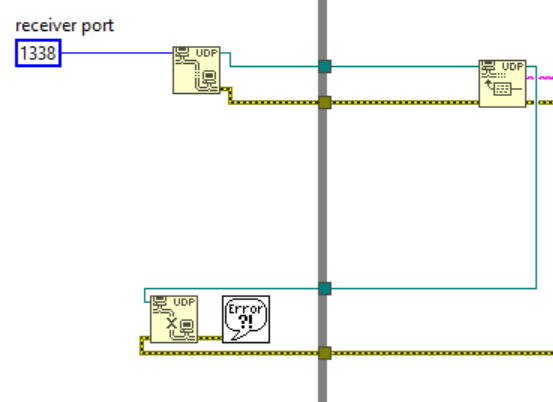
**Getting started guide: Component 3 – Robotic Hand**

Getting familiar with the LabVIEW software

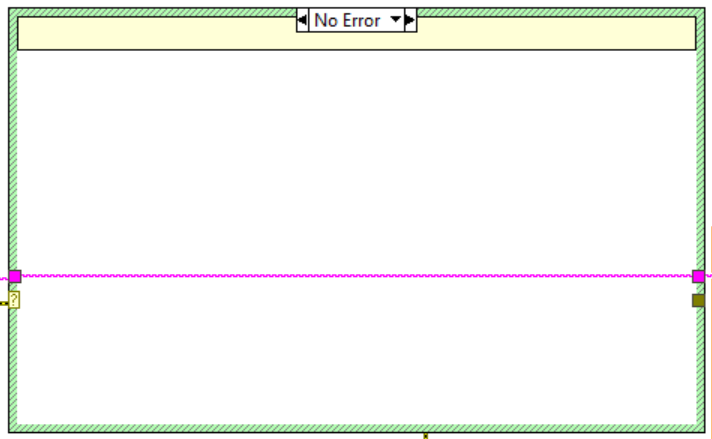
1. Main.vi

This is the complete LabView VI to receive data from Component 2, handle that data and send it to the RoboRIO.

* UDP – Receiving data (block diagram)

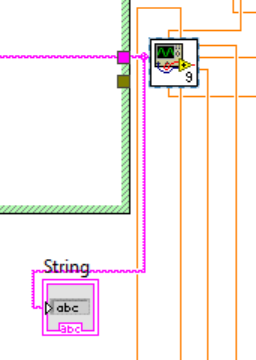


First receiving UDP messages.

En bild som visar text

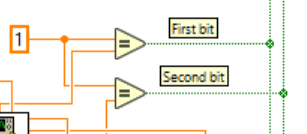
Automatiskt genererad beskrivning

Error handling of these messages. Error if no message available.



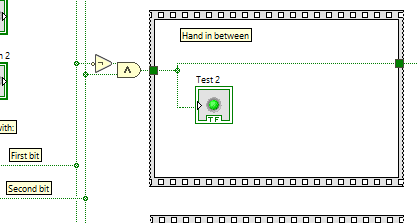
After receiving data, conversions to double are made.

* Handling data part 1 (block diagram)



After receiving the first two bits, they are converted from Double (orange) to Boolean (green) as the figure above shows.

* Handle data part 2 (block diagram)



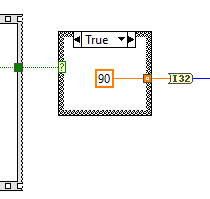
The figure above shows what would happen with command “Hand in between”, i.e. with first two bits equal “01” (see table below). The “Test 2” indicator confirms that this is the case.

The two bits of data received from Component 2 via UDP decides the following:

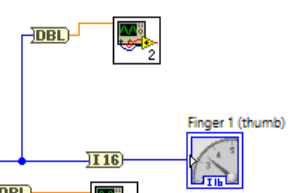
|  |  |  |
| --- | --- | --- |
| First bit | Second bit | Command |
| 0 | 0 | “Hand open” |
| 0 | 1 | “Hand in between” |
| 1 | 0 | “Hand closed” |
| 1 | 1 | “Individual fingers” |

Where “Hand open” means 0 degrees of flexion, “Hand in between” about 90 degrees and “Hand closed” about 180 degrees. This is for all fingers. Then there is “Individual fingers” that allows the user to move one or more fingers at a time to states “open” or “closed” with the last 5 bits in a message (a message has 2+5=7 bits).

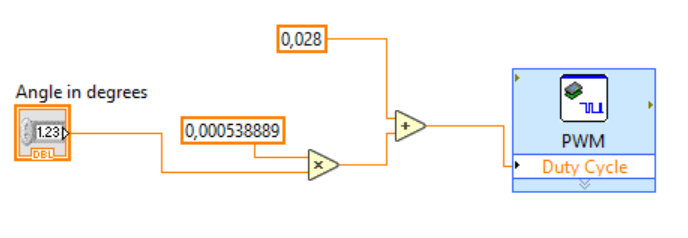
* Handle data part 3 (block diagram)



The received command then specifies how fingers should move, as the figure above shows. Note that this is not the final angle but describes about what angle represents a state. 90 represents “in between”.



The data about each finger is multiplied and added with constants to translate the angle to a PWM number that will eventually move the servo(s) the desired angle. These calculations are made in subVIs for each finger, the figure above is displaying that subVI for thumb + indicator for graphical representation/verification of the thumb movement. This is then sent to RoboRIO in that subVI, as shown in figure below.



1. All hand commands template.vi

There is also an “All hand commands template.vi”. This VI demonstrates how the system (Main.vi) works in theory. Instead of receiving messages from Component 2 and displaying these with the RoboRIO, there are buttons and indicators.

Quick Startup Guide

1. Install LabVIEW 2016 32-bit with roboRIO toolkit and drivers from the DVDs included with the roboRIO
2. Install Python 3.x
3. Plugin the roboRIO: USB, external power and servos. Thumb is PWM0, index finger is PWM1… Be careful and make sure that the servos are not plugged in the wrong way, there is no reverse polarity protection. Black is ground. Plug in the servos before connecting the roboRIO to the computer or external power.
4. Download LabVIEW and Python files from <https://github.com/Emanuel-Bjurhager/robot_hand/>
5. Edit the file called ”UDP relay.py”. Make sure the roboRIO IP variable “IP” is correct  
   Text

   Description automatically generated with low confidence
6. In the file “UDP relay.py” change the variable “UDP\_IP” to your computer IP address (you can find your IP using the command “ipconfig” in CMD) and save the fileA picture containing text

   Description automatically generated
7. Run the Python code using the CMD “ python ‘.\UDP relay.py’ ”
8. Open the LabVIEW project by double clicking the file called “Neuroteknik projekt.lvproj”.
9. In the LabVIEW project, open and run the filed called Main.vi
10. The hand should now move when commands are sent to your computers IP and port 1337. If it does not work, try disabling Windows Firewall.