A summary of the master's thesis Real-time Control of Industrial Robot Cell with PowerLink

Lloyd Tran and David Barbulovic
Department of Automatic Control, Lund University, Sweden

Abstract— **The thesis investigated on how one can build a PowerLink network and Control Nodes that act as e.g., sensors, actuators, etc., together with B&R:s products. How to establish different PowerLink networks was also investigated such as a PLC connected to a Raspberry Pi2 and Pi3 with the image processing software, OpenCV, running in the sensor node with configurations.**

I. INTRODUCTION

To control automation equipment, including workcell coordination and configuration of e.g., industrial robots, conveyor belts, external axes and IO-units, one often uses a Programmable Logic Controller (PLC). This often requires that the user has dedicated configuration programs from the manufacturer.

If the setup of sensors, IO-units or actuators are to be changed, one must program and learn about the PLC through the supplied program from the manufacturer, which may be time consuming, as it requires the user to have the program and the right operating system in order to use it. It would be more resourceful if the user could change parameters through scripting without requiring to use a specific operating system. To implement this, openPowerLink[2] with openConfigurator[4] was used.

Library implementation may also be a limitation as it is often a bit more complex to implement, e.g. OpenCV[5], firmware-update, etc, in the dedicated software for a PLC. The considered robot workcell was to be controlled by a PLC supplied from the Austrian industrial automation company, B&R.

PowerLink is an open source-based deterministic real-time Ethernet protocol. It is a alternative which in contrast to the old field-bus system CAN, run on standard Ethernet equipment and does not require different CAN buses for different tasks.

A. Goal

The goal for the master thesis[1] was to implement a PowerLink real-time communication between a PC and a PowerLink node. The PowerLink node should also be able to work with Automation Studio and be a working device together with, e.g., an industrial robot. The PowerLink node can be, e.g., a Raspberry Pi2 or BeagleBone hardware with IO-units, sensors or actuators. Another PowerLink node will be an ACOPOS 1045 from B&R[3], as the task for the thesis work also concerns the servo-drive and to investigate if and how it can be directly controlled from a PC or other computer units through PowerLink. The final approach will be to setup a PowerLink communication between the Managing Node and ACOPOS 1045, e.g., where a camera connected to a CN acts as a sensor to aid in the task of picking up an object with help of a FlexPicker, where its motor is controlled by Acopos 1045 servodrive.

II. THEORY

PowerLink is a open-source, deterministic Ethernet based protocol that consists of two types of nodes. The first one is the Managing node, MN, which can be seen as the master on the network and there is only one of them in a PowerLink network. The second is the Control Node, CN, which is the slave on the network. PowerLink is deterministic due to the PowerLink Cycle Time. During this time, the MN tells the CN:s that data must be sent during the time interval. If the time is not enough, then the PowerLink network would not work. But if the Cycle Time is too big, then the determinism is lost and unnecessary delay is introduced. The minimum Cycle Time can be calculated, which needs to be done if determinism is sought.

III. METHOD

The methodology for reaching the thesis goal of creating PowerLink nodes, consisted mainly of acquiring knowledge on PowerLink, B&R:s PLC, Raspberry Pi2, OpenCV[5], which is a image processing software and installing prerequisite files that are required in the PowerLink implementation. First a small PowerLink network was established between two PC:s, in order to see how PowerLink works. Here the network traffic capture program, Wireshark, was used to observe how the data was exchanged in the network. The PowerLink stack was downloaded and then built with Cmake and then executed through the command prompt.

The next phase was to investigate how the MN can send data to the CN. Here the setup was a windows PC as MN and a B&R X20BC0083, which is a device with digital and analog I/O:s. In order to do this, OpenConfigurator was used, which can be seen as a PowerLink network creation toolkit. It should be mentioned that in order to use OpenConfigurator, an XML Description file, XDD-file, for both the MN and CN has to be written. The XDD-file describes what kind of device the node is. It should also be mentioned that without a XDD-file for a device, it is not possible to control it. In the XDD-file, a description is made such as which data to be send or received and what kind of datatype, INTEGER8, REAL32, etc. Thankfully, these files for both of the nodes can be downloaded from OpenPowerLink's website and manipulated to suite the desired PowerLink network data. For the B&R X20BC0083, the XDD-file was downloaded from the manufacture's website. The files generated from OpenConfigurator were then implemented in the PowerLink stack of the MN and then executed. Through coding the desired I/O:s were set, which also was able to be observed from indicator LEDs at each digital output.

Investigation was also done regarding how CN sends data to MN, where the setup was a B&R PLC as MN and Raspberry Pi2 as CN. By using Automation Studio and changing the contents of the XDD-file, it is possible to implement the CN through Automation Studio. Although the Objdicts.h file, which is part of PowerLink files, also has to be changed in the CN. The Objdicts.h resembles a XDD-file and these needs to be the same or the PLC and CN cannot communicate with each other.

When enough knowledge was gained regarding of how one controls a PowerLink node, then the next part began of implementing a PowerLink control node for Acopos 1045 in order to control the FlexPickers motors. However due to that the XDD-file was not available for Acopos 1045 controlling the FlexPicker was not possible as intended for this master thesis.

Instead focus was shifted to creating the other PowerLink node. In order to create a camera node, the image processing software OpenCV was implemented on a Raspberry Pi2, Raspberry Pi3 and on an Ubuntu stationary computer. Here the camera PowerLink node acted as a CN and a B&R PLC as MN.

The camera control node could detect an object in the room and send back information of the position of the object in pixel coordinates. Threading was also introduced, as the Raspberry Pi2 and Pi3 have limited computing power and to ensure that the PowerLink communication would not suffer.

IV. Result

A. Creating and Controlling Control Nodes

The result showed that creating and maintaining a PowerLink requires a lot of knowledge of the theory behind it. Setting up the network can be cumbersome for newcomers, as alot of how to program is not demonstrated in the document for PowerLink. Directly controlling ACOPOS 1045 was not successful as the XDD-file for was not available. Although this thesis goal was not achieved, the direct control of B&R:s X20BC1483 showed that it is possible, as long the XDD-file for the device is provided by the manufacturer. With the XDD-file user can also alter the file to desired configuration like add or remove I/O variable and run it on Raspberry or Beaglebone.

B. The OpenCV Software

When OpenCV was executed on the Raspberry Pi2, measurements showed that the computation time was approximately ten times slower than when running on a PC which proved that the Raspberry Pi2 was not suitable for high performance system. Implementation on a Raspberry Pi3 was also made, which improved the performance, but not enough to be satisfactory. Instead an Ubuntu stationary computer was used and it worked satisfactorily. The computation time for the OpenCV part of the code to calculate the pixel coordinate had a median time of 0.138 seconds, while the time for sending the information from PowerLink to the PLC had the median time of 0.00016 seconds. This shows that the PowerLink node has good real-time performance and is suitable to be integrated in an automatic control device, e.g., determining position for robots or for objects in their workspace.

V. Conclusions

Creating new I/O:s and datatypes proved to be cumbersome as no direct documentation showed how to create these. The PowerLink stack provides a lot of files, which makes it easy to get lost in the jungle of information. OpenPowerLink's webpage provides demo of how to build a PC to PC PowerLink connection, though there is no mentioning on how for example, to send some data from one PC to another.

Learning how to use and creating a PowerLink network was a handful of work, as the demo for implementing a Raspberry Pi2 CN did not work as planned. This was because the files for some reasons could not be installed properly, which required a lot of time before this problem could be solved. Even using OpenConfigurator was a problem, here is a lot of information about it, even tutorials and videos, but no documentation showed which files should be replaced in the PowerLink stack. To know which files to replaced was known through testing.

This thesis also showed the vulnerablity of creating a PowerLink node, if the XDD-file for a device is not provided from the manufacturer. It also showed that it is possible to build a stable PowerLink node, which allows the user to send data from a a sensor node to a PLC.

Acknowledgment

We want to thank B&R and the Department of Automatic Control for helping us with supplies of hardware and knowledge. A special thanks to our supervisor Anders Robertsson for help and advice during the thesis period.

References

[1] L. Tran & D. Barbulovic, Real-time Control of Industrial Robot Cell with PowerLink. Master thesis, Lund University, Department of Automatic Control, http://www.control.lth.se/Publications.html, TFRT-6048, 2018

[2] Ethernet PowerLink Homepage, http://openpowerlink.sourceforge.net/web/. [Online; accessed 20-Dec-2017]

[3] B&R Homepage, https://www.br-automation.com/en/perfection-in-automation/.[Online; accessed 19-Dec-2017]

[4] Kalycito, User Manual for Ethernet POWERLINK openCONFIGURATOR eclipse plugin, Version: 2.1.1. https://goo.gl/jORhdR.[Online; accessed 21-Dec-2017]

[5] OpenCV, About OpenCV, http://opencv.org/about.html. [Online; accessed 19-Dec-2017]