Proceedings of the 2008 Winter Simulation Conference
S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler eds.

OFFLINE COMMISSIONING OF A PLC-BASED CONTROL SYSTEM USING ARENA

Jeffery S. Smith
Younchol Cho

Dept. of Industrial & Systems Engineering
Auburn University
Auburn, AL 36849, USA

ABSTRACT

In this paper, we address a generalized method of mapping a control system simulation model to the PLC emulator being tested using model variables and PLC tags under the offline commissioning environment. For this research we created an example system similar to a high speed packaging system described in a previous WSC paper. Implementation experience using Rockwell Software applications is provided.

1 INTRODUCTION

High speed filling and packaging line systems require high stability at specified speed and well-balanced control program under complicated control conditions to ensure high system through put. These complex programs developed to control these systems run on the programmable logic controller (PLC). Mueller (2001) describe a simulation-based technology called "PolySim", designed to improve the quality of these systems. The example system described uses the AutoMOD simulation package.

Traditionally, the final approval testing or commissioning of a control system is performed just prior to the startup phase of the controlled system. This is an expensive, risky and error-prone way of developing control systems due to the limitation of testing caused by the high level of safety issues to operators and damages to machines in the testing site (Auinger 1999; Schludermann et al. 2000; Versteegt et al. 2002). However, the use of simulation allows one to develop these systems in a safe way with less time and costs.

In this paper we create a high speed packaging (HSP) system similar to the one described by Mueller (2001). The purpose of this paper is to present a general method to map control system model to controller for testing before commissioning. We use several Rockwell Software applications to accomplish the goal of this paper.

The organization of this paper is as follows. In section 2 we discuss simulation-based commissioning. Section 3 describes a general concept for mapping control system model to controller using Rockwell software. In the basis of the concept described in section 3, section 4 introduces the detail procedure with HSP example system. Finally, we will make conclusions in section 5.

2 SIMULATION-BASED COMMISSIONING

Simulation has been widely used for shop floor control system design, implementation, control and testing. In particular, much of the efforts for the testing of control systems could be shifted to the development phase of control system, since simulation has been used for it. Therefore, the mechanical function of the control system is only tested during the commissioning phase after the plant is built (Miles and Siddeley 1989).

Many authors have methods for control system testing. They suggest to use a combination of reality and simulation to test control systems as a general solution. Figure 1 presents the methods based on these combinations.

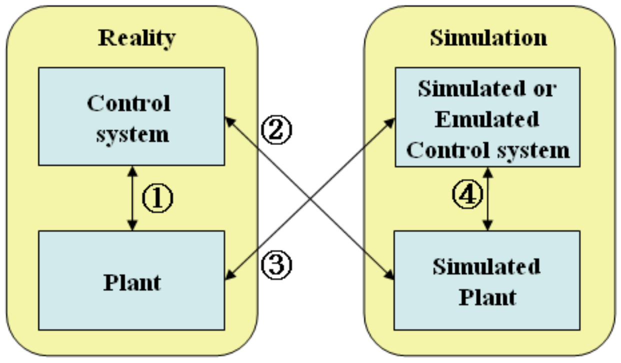


Figure 1: Four possible approaches for commissioning

Four possible approaches can be distinguished based on the possible combinations between reality and simulation (Versteegt et al. 2002; Auinger et al. 1999).

- Traditional commissioning (①): Both systems in reality.
- Soft-commissioning (②): real control system and simulated process system.
- Reality-in-the-loop (RIL) (③): simulated control system and real process system.
- Offline commissioning (④): Both systems simulated.

In today's manufacturing industry the use of autonomous devices using embedded software is common and is growing continuously. In order to handle these many autonomous subordinate control devices the systematic control systems are required. Eskin et al. (2003) present that these systematic control systems controlling the sub-devices should be highly scalable and reliable, and also be able to be commissioned on time with less costs. The commissioning method corresponding to these problems is to use simulation technology.

Under traditional testing method a PLC controller is directly connected to the physical hardware. The biggest disadvantages of this direct connection to hardware involve the risk associated with testing on the physical hardware and the cost associated with the system downtime. In the soft-commissioning approach, the PLC controller is connected to simulation instead of hardware under, resulting in a safer and less expensive testing environment. Figure 2 presents the difference between traditional way and soft-commissioning / offline commissioning approach according to simulation use for testing control systems.

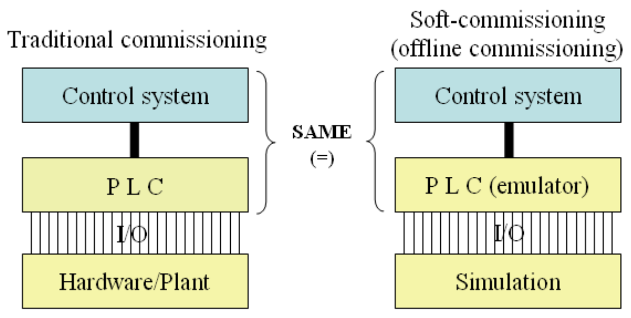


Figure 2: The difference between traditional and soft-commissioning

Consequently, by using these soft-commissioning and offline commissioning methods we can test control systems in a very effective and flexible way, more than anything else many experiments which can not be performed with real system because of safety and costs issues can be conducted easily under safe, better controlled conditions. (Schiess 2002; Versteegt et al. 2002; Pfeiffer et al. 2003).

3 MAPPING CONTROL SYSTEM MODEL TO CONTROLLER BEING TESTED

In this section we will focus on addressing the general concept of mapping control system model to PLC controller, which is performed based on the offline commissioning approach.

Four different Rockwell Software applications are used for our demonstration. These include: 1) ArenaRT simulation package which models the system; 2) RSLogix5000 Loader which is used for control logic programming; 3) RSLogix Emulate 5000 which emulates PLC controller; and 4) RSLinx which is used to interface between Rockwell Software applications.

We assume that the optimized process model is already prepared using Arena simulation package. Therefore, we will address the process from development control system model. The steps are listed as follows:

- Development of the control system model and control logic
- Configuration of emulator
- Connection between controller and emulator
- Download control logic to emulator
- Connection between simulator and controller

Mapping is here to connect control system model and controller being tested using variables and tags defined within both systems. I/O variables are used for communicating with tags of the controller and internal variables are used within the simulation model to support the simulation logic.

Controller tags are used a method for assigning and indirectly referencing the I/O memory locations in the controller. Particularly, the physical addresses are not required with RSLogix5000 Loader. Instead they are replaced with tags which are a pure text based addressing scheme. As a result, tags and variables are crucial means to enable the exchange of information between the PLC controller and control system model.

Figure 3 describes the overall process of mapping simulation model variables defined in the control system model to tags that map to I/O points in the controller.

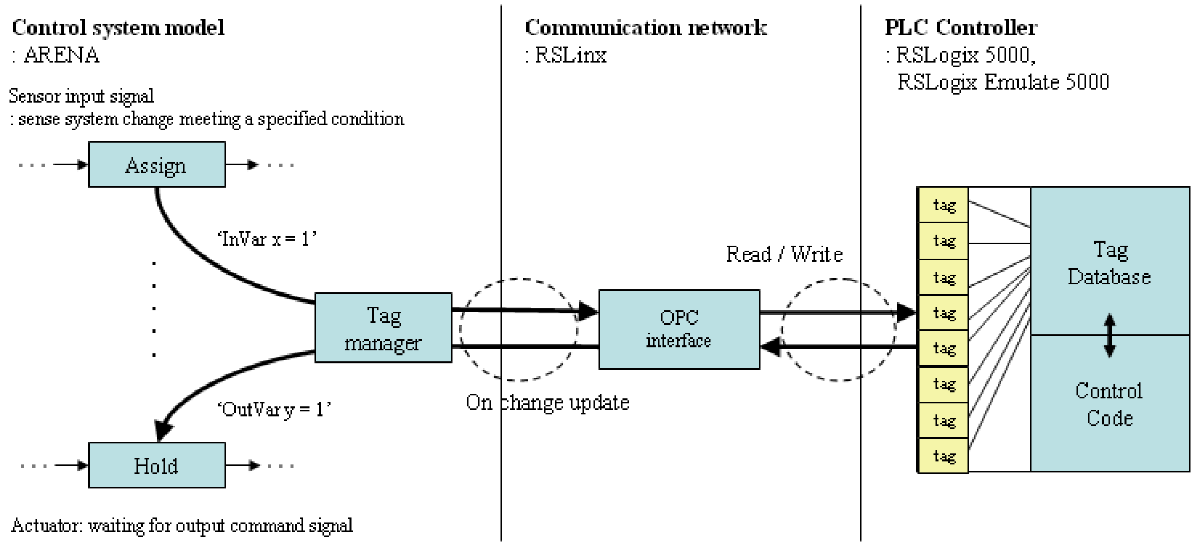


Figure 3: The overall process of mapping Arena variables to PLC tags

In order to configure the I/O points (variables) for control system model the following primary steps are proposed:

- Classify physical resources
- Configure and define the I/O points needed for each control level or each physical resource
- Construction of control system model based on the I/O points defined

The basic control model involves using sensors to detect components of the system state and actuators for affecting control. So, at the most basic level, the PLC is responsible for monitoring the sensors (inputs) and setting the actuators (output) to affect control of the system. For example, consider a simple conveyor. Typical inputs that the conveyor controller needs to monitor include:

- Total number of entities on the conveyor
- The number of accumulated entities at the specified point on the conveyor
- Conveyor failure or stop, etc.

The I/O points should be able to represent the appropriate system status required for effective control of systems and must be defined with a unique name. Also, the I/O points communicate based on the binary system, thus they can have only '0' or '1' value. Here value '1' means activation. For instance, define 'number of containers accumulated on the conveyor' from the Figure 4 as a variable 'x' for input point. In the real system, the Accumulation sensor configured on the conveyor checks continuously for the presences of accumulated containers. It notifies the control system by sending a input signal of 'Accumulation sensor=1' when the specified condition is met. When the simulation model replaces the physical conveyor, this accumulation is modeled using the Arena variable corresponding to the conveyor accumulation. So, when the accumulation in the simulation reaches the specified point, the variable mapped to the corresponding I/O tag is set to a value of '1' (InVar x in Figure 3). As such, the PLC does not know (or care) whether it is "talking" to the real sensor or the simulation. Similarly, if the control system needs to take some action in response to the accumulation (e.g., slow down process 1), the PLC sets an output value. If the PLC is connected to the real system, this tag is connected to process 1 and it slows the process. If the PLC is connected to the simulation model, the tag is connected to a simulation variable that slows down the simulated process (OutVar y in Figure 4).

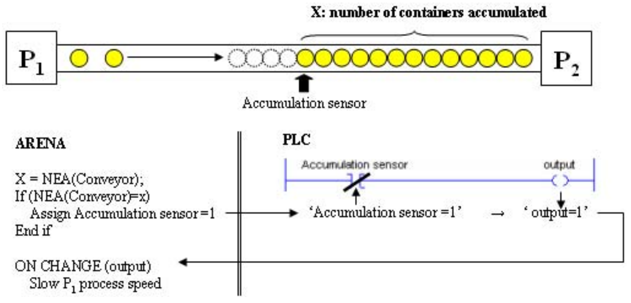


Figure 4: On change update by I/O signals

After identifying the physical resources and the corresponding I/O points, the next step is to construct control system model based on the I/O points identified. To split the optimized simulation model into control logic and control system model we use the HOLD and ASSIGN modules of Arena. The ASSIGN module is used to send an input signal or update the system status. The HOLD module is used to receive a responding output signal from controller to activate a equipment or machine. Note that although we described the construction of control system model in terms of Arena, the general methodology is applicable to virtually any discrete-event simulation package.

For our example, the control system logic is programmed by RSLogix5000, which uses the ladder diagram language for PLC program. The ladder diagram language is basically a symbolic set of instructions used to create the controller program. These ladder instruction symbols are arranged to obtain the desired control (Petruzella 1989).

The control logic is written in conjunction with the I/O configuration in the control system model by ladder logic diagrams in this paper. The following steps guide the basic process for programming a control logic with RSLogix5000:

- Create a project
- Configure the controller
- Create and configure I/O
- Enter tags
- Writing control logic
- Download the control logic to controller

Before performing the last step of downloading control logic to controller, the emulator should be configured and connected to RSLogix5000 in advance. RSLogix Emulator 5000 is a software simulator for RSLogix5000 controller. The purpose of this software is to mimic the function of a PLC without the actual hardware and do debugging in more advanced manner. That is the emulator allows us to experiment with and debug the control logic programmed on RSLogix5000 in a safe and controlled environment without investing in physical controllers and I/O modules.

After setting up the emulator, it should be connected to RSLogix5000 controller using the RSLinx. RSLinx Classic is a comprehensive factory communications solution for the Microsoft Windows operating systems and it provides Allen-Bradley programmable controller access to a wide variety of Rockwell Software and Allen-Bradley applications.

The purpose of RSLinx is to provide connectivity for client applications using OPC or multiple DDE data formats. Traditional communication tools are required for its own interface package to access data from a device whenever it needed. However, the emergence of OPC removed this inconvenience and it was a major step towards the connecting of control systems from different vendors and enabling the development of software that could easily access data from a wide range of devices without any manufacturer-specific code from the third party developers. DDE is also a standard inter-application communication protocol built into Microsoft Windows operating systems and supported by many applications that run under Windows. DDE takes data from one application and gives it to another application. It allows windows programs that support DDE to exchange data between themselves. Here, the emulator as a client application can access to RSLogix 5000 controller by RSLinx. Once finished all processes described above, the control logic can be downloaded to the emulator for testing and debugging.

Up to now, we have addressed all the processes required before connecting simulator to PLC controller. Finally, if we connect all I/O variables to the matched tags in the controller by the tag manager, connecting the control system model to controller is completed. ArenaRT has tag manager that is used to connect variables defined in the control system model to tags defined in the controller. In the next section the steps will be presented in detail with applying this concept to High Speed Packaging (HSP) system testing.

4 HIGH SPEED PACKAGING SYSTEM DESIGN AND TESTING

4.1 System Description

There are two processors and one connecting conveyor physically configured in this system (as shown in Figure 5). The empty containers (e.g. bottles) from exterior facility to HSP (High Speed Packaging) system are filled with some material contents at P_1, and transported to the next processor through conveyor, then labeled with product instruction at the P_2 (Labeler) just before leaving this system.

Process failures at P_1 and P_2 can cause blocking and starvation of the individual processes. Figure 5 shows the system working in the nominal case (both processors working nominally).

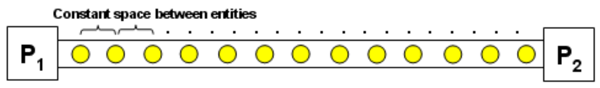


Figure 5: Nominal case

The space between entities being transported on the conveyor is constant and the space length is a function of the conveyor speed and the processing speeds of the machines. As mentioned before, the stability is very susceptible issue in this system. Therefore, under nominal situation P_1, P_2 process rate and conveyor speed can be adjusted to maintain constant amount of containers on the conveyor. In other words if the number of containers is less than the specified number, P_1 process rate and conveyor speed will be increased. Conversely, if the number is larger than the specified one, P_2 process rate will be increased.

If P2 fails, containers will start to accumulate containers at the front of P2 (as shown in Figure 6). In addition, if the P_2 failure lasts until when the accumulation arrive at the left end of conveyor, which means conveyor is full of entities without any empty space, P_1 will be blocked. Once P_1 gets blocked, it is immediately turned off. Restarting P1 after blocking and shutting down requires some delay time for system retuning when P_2 is completely repaired and begins to work. Therefore, much efforts are required to avoid or minimize this blocking. One way as an alternative is to set a *cutting point* on the conveyor to delay the accumulation. When the accumulation reaches the cutting point, the system slows P_1 in order to slow the accumulation and reduce the chance of blocking.

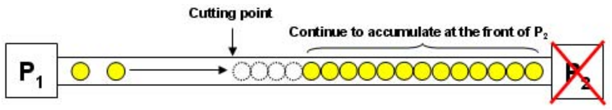


Figure 6: P_1 blocking case

In contrast to the previous P1 blocking case, P2 starvation case as seen in the Figure 7 is that P1 is failed while P2 working properly with normal process rate.

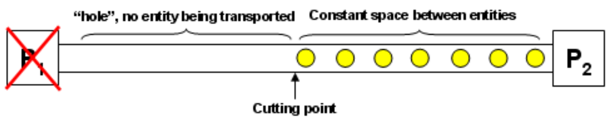

"hole", no entity being transported
Constant space between entities
P1
P2
Cutting point

Figure 7: P_2 starvation case

The number of containers moving on the conveyor under this case starts to decrease because of the failure of P_1. If P_1 is not repaired before all containers on the conveyor are processed, P_2 will be starved. After P_2 waits for a container for specified time, then it is also turned off like P_1 if any container does not come to P_2 during that time. It also requires some delay time to restart later as P_1 does. So, in order to avoid or minimize the chance another cutting point is employed to slow the P_2 process rate down. When the "hole" reaches the cutting point, P_2 process rate is decreased.

4.2 Problem Description

Basically the following data are given:

- P_1 and P_2 process rate (normal, slow, fast).
- TTF, TTR of P_1 and P_2.
- ttr (time to restart from P_1 blocking or P_2 starvation).
- Waiting time: the time period that P_2 can wait for a container before getting starved.

Based on these given information, Following parameters should be decided appropriately:

- CP1 and CP2 (Cutting Point): The process rate of P_1 and P_2 at the each specified cutting point on the conveyor are reduced to reduce the chances for P_1 blocking or P_2 starvation.
- Conveyor speed: The moving speed of container on the conveyor. There are two type of conveyor speed; normal and high. High conveyor speed is used to avoid P_2 starvation.
- Conveyor length: Conveyor capacity. It can hold and transfer more entities as long as it has.

However, for simple problem demonstration we assume that all parameters' value are specified beforehand and CP2 is fixed at the middle point of conveyor. Thus, we are interested in only CP1. Basic input data (time unit is minute) are given as Table 1.

Table 1: Input data

P	TTF	TTR	ttr	Wait time
P_1	expo(150)	2+expo(10)	tria(15,20,25)	
P_2	expo(25)	1+expo(0.9)	tria(3,4,5)	0.05
	Process rate (normal/slow/fast)		Conveyor	
P_1	200/10/240		Speed (normal/high)	300/350
P_2	200/10/240		Length	600 cells

4.3 HSP Control System Commissioning

Ruppel and Wysor (1997) define the fidelity as the degree of similarity between a model and the system being modeled. Although the static analysis usually use an highly abstracted model of system state that loses some information, it gives a chance to estimate a system performance roughly and enable one to test the effect of changes in system parameter quickly.

First, we perform static analysis with available data, TTF, TTR, normal process rate, conveyor length and normal speed, using the Microsoft Excel. As mentioned before, the result from static analysis gives only rough estimation about system performance according to the parameter changes. As seen in Table 2 we can get the boundary information related to max system through put.

Table 2: Static analysis result

P_S	Repair time	Working time	Actual service rate
P_1	74.07	925.93	185.19
P_2	70.63	929.37	185.87

The max system through put should be 185.19 per minute since this is a serial line.

Dynamic simulation model developed using Arena package represents the system in more detail than static model and uses all given data for analysis. So with the dynamic analysis results we can make more practical decisions for designing a optimized process. For model validation after programming we run the model with the same data used for static analysis without any dynamic parameters like block and starvation time, etc. The results should be same if programming is done correctly.

When completing the model coding and validation, we conduct many experiments to find the best CP1. According to the experiment result chart in the Figure 8 the CP1 should be decide based on the range between 220 and 540.

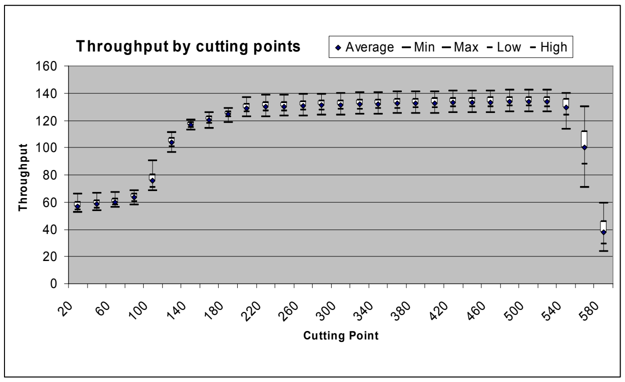


Figure 8: Flow rate chart by cutting point

But, this decision has been made based on the aspect of system through put. Actually, there could be many other considerations and aspects to find best CP in the system, and it could be different depending on the cases. However, we do not deal with this issues here since that is beyond the paper objective.

Once we get the optimized process model, the development of control system model is started with configuring I/O points process.

Three equipment are identified in the HSP system, which are P_1, P_2, and conveyor. Four input and three output points are required for P_1, P_2 as seen in the Table 3. For the input points 'Failed' means the breakdown of processor, 'DeT' finishing the process delay time, and 'com' complete process in the assigned processor. 'Inactive' of P_1 represents the blocking state caused by full conveyor with containers and 'Inactive' of P_2 indicates exhaust state caused by breakdown of P_1. At first as given three different speed of process rates three output points are required to control those process rates according to systems status. Whereas, conveyor does not require the input point for failure state since we assumed that no breakdown of conveyor occurs in this system. Instead, conveyor have been defined with other many input points representing the complicated traffic situation on the conveyor. MoreTH means the number of containers being moved is lager than half of the conveyor length, conversely LessTH smaller. CP1_Arrival and CP2_Arrival represent arrival to each cutting point. NoCon_CheckBox is defined to indicate if container exists in the specified region to decide the P_2 state. If no container exist in that area, P_2 state is changed to exhaust. Finally, the PE_1 points out that the conveyor is full of containers. As the result of PE_1 input P_1 get blocked.

Table 3: Equipment level I/O points

Eq	Input point	Output point
P_1	Filer_Failed Filer_Inactive Filer_DeT Filer_com	Filer_ProcessRateSlow_exe Filer_ProcessRateNorm_exe Filer_ProcessRateUp_exe
P_2	Labeler_Failed Labeler _Inactive Labeler _DeT Labeler _com	Labeler _ProcessRateSlow_exe Labeler _ProcessRateNorm_exe Labeler _ProcessRateUp_exe
Conveyor	MoreTH LessTH CP1_Arrival CP2_Arrival NoCon_CheckBox PE_1	ConvSpeedNorm_exe ConvSpeedUp_exe

The I/O points described above are belong to the equipment control level in the control hierarchy. However, for cooperation between equipment or workstations the higher level of I/O points are required. There are a few I/O variables defined for that in this system. As seen in the Table 4 below Filer_CS and Labler_CS are used to signal of container's arrival for process in the corresponding workstation, Filer_OK and Labeler_OK are defined to indicate if those processor are available. These workstation level I/O points are controlled by workstation level controller.

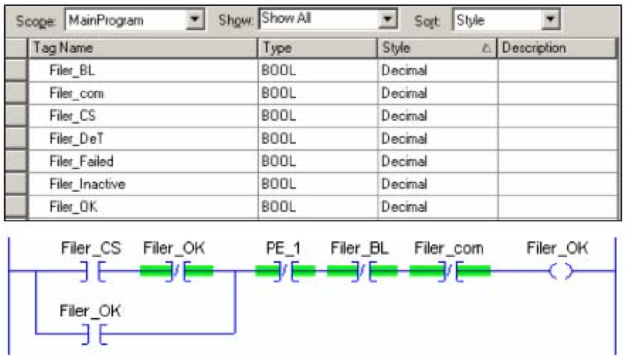
Table 4: The higher level I/O points

Level	Input point	Output point
Workstation	Filer_CS Labeler_CS	Filer_OK Labeler_OK
Shop	CheckOK	

CheckOK in the HSP system is considered as a shop level input point since it is referenced by workstation controllers in the shop. Many of processes from different workstations require the input signal of 'CheckOK==1' in advance before going forward. Up to this point we have configured I/O points to be required. Finally, we should note that all I/O points must be defined as a variable in the Arena package.

Next step is to remove the control logic from simulation model based on these I/O points using Arena's hold and assign module. The assign module is used to send input signal to controller. For example one entity comes into the assign module to send the signal 'CP1_Arrival==1' when the precondition if accumulated containers reach the cutting point is met. Then, directly moves to hold module to wait for the output signal 'Filer_ProcessRateSlow_exe==1' to control the process rate of P_1 properly. This should be done with other I/O points in the same way. The development of control system model is accomplished when all splitting work and verification are properly finished.

After finishing the development of control system model, we start to program the control logic using RSLogix5000. First, we need to write all the tags, which should be used the same name as control system model to avoid any confusion. Then, based on the tags we program the control logic. Figure 9 shows the programmed tags and P_1 station control logic. Tags are used in the ladder diagram as instructions to program the control logic. The type of tags are selected by 'BOOL', since these tags work based on the binary system. Refer to the guide manual for detail use.

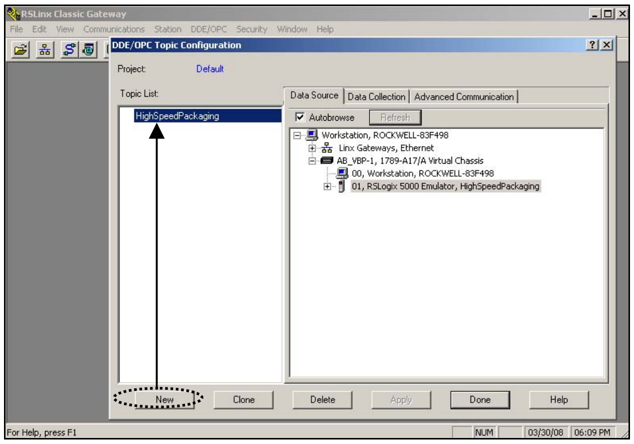


Tag Name	Type	Style	Description
Filer_BL	BOOL	Decimal	
Filer_com	BOOL	Decimal	
Filer_CS	BOOL	Decimal	
Filer_DeT	BOOL	Decimal	
Filer_Failed	BOOL	Decimal	
Filer_Inactive	BOOL	Decimal	
Filer_OK	BOOL	Decimal	

Figure 9: Tags and control logic

The tag 'Filer_BL' in the Figure 10 has not been defined in the control system model as a I/O points. This is the same as the internal variable of control system model. Now, we are ready to connect the control system model to controller for commissioning.

First, the emulator should be configured and coupled with RSLogix5000 using RSLinx before downloading the control logic from RSLogix5000. Refer to guide manual for detail process we will address only the coupling process here. To connect emulator to RSLogix5000 choose "Topic configuration" from the DDE/OPC menu of RSLinx and then DDE/OPC Topic configuration diagram appears as seen in the Figure 10. Then, click the "New" button and name the topic "HighSpeedPackaging". Finally, in the right pane expand the AB_VBP-1, 1789-A17/A Virtual Chassis, click the RSLogix5000 Emulator and click "Done" button to connect this new OPC topic to the Emulator.



Project: Default

Topic List: HighSpeedPackaging

[✓] Autobrowse

Workstation, ROCKWELL-83F498
Linx Gateways, Ethernet
AB_VBP-1, 1789-A17/A Virtual Chassis
00, Workstation, ROCKWELL-83F498
01, RSLogix 5000 Emulator, HighSpeedPackaging

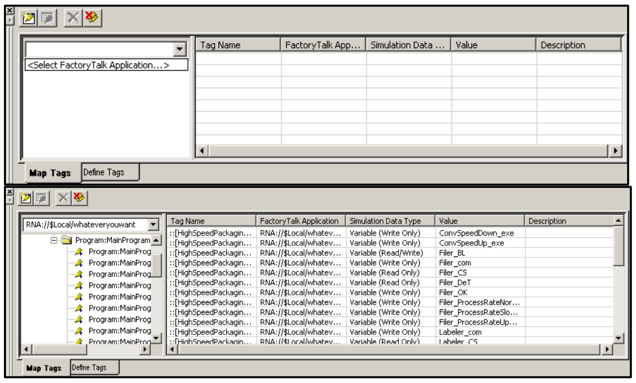
Figure 10: Connecting between controller and emulator

Then, we can download the control logic to the emulator. From the Communications menu of RSLogix5000 controller choose "Who Active". Then expand AB_VBP-1 and choose the RSLogix 5000 Emulator in slot 1. Finally, click "Download" button and confirm that the "HighSpeedPackaging" project is downloaded into the slot 1.

Before starting to mapping process we have to check that RSLogix Emulate 5000 Chassis Monitor and RSLogix 5000 Loader should be in Run or Program Mode. If both program do not activated, it is impossible for control system model to be connected to PLC controller.

Tag manager window appear at the bottom part of Arena by choosing "Tag manager" from the menu. First, click the button in the list box of left pane and select the text "<Select FactoryTalk Application…>" to pops up the Open FactoryTalk Application window.

When the "whateveryouwant" from the FactoryTalk Application window is selected, all the variables defined in the control system model appear in the left pane of upper part tag manager in the Figure 11.



<Select FactoryTalk Application…>

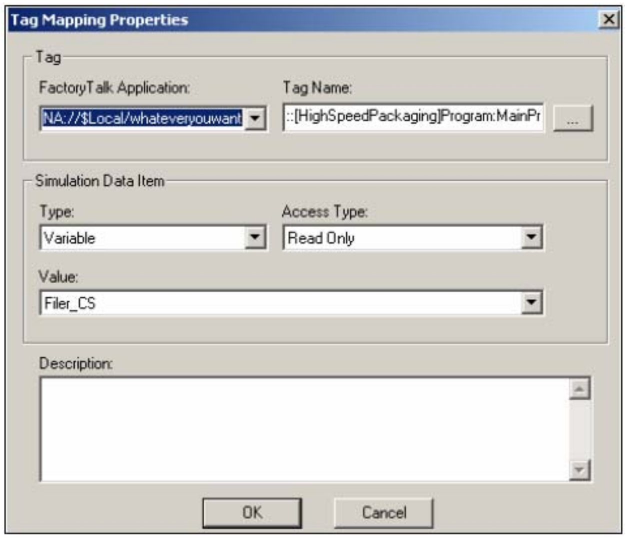
RNA://$Local/whateveryouwant

Program:MainProgram
Program:MainProg
Program:MainProg
Program:MainProg
Program:MainProg
Program:MainProg
Program:MainProg
Program:MainProg
Program:MainProg
Program:MainProg

Tag Name	FactoryTalk Application	Simulation Data Type	Value	Description
::[HighSpeedPackagin…	RNA://$Local/whatev…	Variable (Write Only)	ConvSpeedDown_exe	
::[HighSpeedPackagin…	RNA://$Local/whatev…	Variable (Write Only)	ConvSpeedUp_exe	
::[HighSpeedPackagin…	RNA://$Local/whatev…	Variable (Read/Write)	Filer_BL	
::[HighSpeedPackagin…	RNA://$Local/whatev…	Variable (Write Only)	Filer_com	
::[HighSpeedPackagin…	RNA://$Local/whatev…	Variable (Read Only)	Filer_CS	
::[HighSpeedPackagin…	RNA://$Local/whatev…	Variable (Read Only)	Filer_DeT	
::[HighSpeedPackagin…	RNA://$Local/whatev…	Variable (Write Only)	Filer_OK	
::[HighSpeedPackagin…	RNA://$Local/whatev…	Variable (Write Only)	Filer_ProcessRateNor…	
::[HighSpeedPackagin…	RNA://$Local/whatev…	Variable (Write Only)	Filer_ProcessRateSlo…	
::[HighSpeedPackagin…	RNA://$Local/whatev…	Variable (Write Only)	Filer_ProcessRateUp…	
::[HighSpeedPackagin…	RNA://$Local/whatev…	Variable (Write Only)	Labeler_com	
::[HighSpeedPackagin…	RNA://$Local/whatev…	Variable (Read Only)	Labeler_CS	

Figure 11: Connecting control system model to controller

Each I/O point can be connected to responding tag in the controller one by one. The left pane of tag manager represents all the tags stored in the controller. What we have to connect those is just click the tag to be connected and then tag mapping properties diagram appears in the Figure 12. we used 'Filer_CS' tag for demonstration.


Tag Mapping Properties

Tag

FactoryTalk Application: NA://$Local/whateveryouwant

Tag Name: ::[HighSpeedPackaging]Program:MainPr

Simulation Data Item

Type: Variable

Access Type: Read Only

Value: Filer_CS

Description:

Figure 12: Matching variables and tags

We have to select the variable from Type field which should be connected to 'Filer_CS' tag. Thus, Filer_CS variable is put in the value field. There are three types access in the access type field: Read Only, Write Only and Read/Write. Read Only is chosen when the tag is used as a Input and Write Only can be selected when the tag is used as a Output in the ladder diagram of control logic. However if the tag represents both Input and Output, Read/Write type must be selected. The upper part of Tag manager window will be changed to lower part Tag manager window in the Figure 14 when we finish all this coupling work.

Now, we completed the connecting the simulation to PLC controller. As the final step for commissioning HSP control system we need to check if the commissioning model react properly to control system as real system does. If the model does not work properly, we should go back to previous step to debug the error, otherwise the commissioning work is finished.

5 CONCLUSIONS

We used the HSP example system to present a basic concept of mapping a control system model to a controller for testing the control system. In addition, we have presented a specific implementation using several Rockwell Software applications (including the Arena simulation package and the Logix family of PLC tools). For the example, we have focused on the configuration of I/O points, connecting it to tags defined in the control system for communication between the control system and Arena model of the controlled system. While this paper focuses on using Arena and the Logix tools, the methodology can easily be applied using other simulation and control tools.

REFERENCES

Auinger, F., M. Vorderwinkler, and G. Buchtela. 1999. Interface driven domain-independent modeling architecture for "soft-commissioning" and "reality in the loop. In *Proceedings of the 1999 Winter Simulation Conference,* ed. P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, 798-805: Institute of Electrical and Electronics Engineers, Inc.

Dougall, D. J. 1998. Applications and benefits of real-time I/O simulation for PLC and PC control systems. *ISA Transactions* 36 (4):305-311.

Eskin, A., and A. Frank. 2003. Offline Simulation of a Managed System for Testing a Developed Management system. In *Proceeding of the 2003 IEEE International Conference,* 154-163: Software-Science, Technology & Engineering (SwSTE'03).

Miles, T. I., and H. Siddeley. 1989.Using Discrete-event Computer Simulation to Test Control Systems. In *proceedings of the 1989 Winter Simulation Conference*, ed. E. A. MacNair, K.J. Musselman, and P. Heidelberger, 848-858.

Mueller, G. 2001. Using emulation to reduce commissioning costs on a high speed bottling line. In *Proceedings of the 2001 Winter Simulation Conference,* ed. B. A. Peters, J. S. Smith, D. J. Medeiros, and M. W. Rohrer, 1461-1462.

Petruzella F. D. 1989. *Programmable Logic Controllers*. New York: McGRAW-HILL, Inc.

Pfeiffer, A., B. Kádár., and L. Monostori. 2003. Evaluating and improving production control systems by using emulation. In *Proceedings of the 2003 International Association of Science and Technology for Development,* 261-267. Marbella, Spain.

Rockwell Software Inc.. 2000. RSLinx classic Getting Results Guide. Available via < http://literature.rockwellautomation.com/idc/groups/literature/documents/gr/linx-gr001_-en-e.pdf> [accessed March 17, 2008].

Rockwell Software Inc.. 2000. RSLogix Emulate 5000 Getting Results Guide. Available via < http://literature.rockwellautomation.com/idc/groups/literature/documents/gr/tstent-gr001_-en-p.pdf> [accessed March 17, 2008].

Rockwell Software Inc.. 2000. RSLogix 5000 Loader Getting Results Guide. Available via <http://literature.rockwellautomation.com/idc/groups/literature/documents/gr/lg5000-gr001_-en-p.pdf> [accessed March 17, 2008].

Ruppel, F., and W. Wysor. 1997. Guidelines for simulation-based control system testing. SEMATECH.

Schiess, C. 2001. Emulation: debug it in the lab-not on the floor. In *Proceedings of the 2001 Winter Simulation Conference,* ed. B. A. Peters, J. S. Smith, D. J. Medeiros, and M. W. Rohrer, 1463-1465.

Schludermann, H., T. Kirchmair, and M. Vorderwinkler. 2000. Soft-commissioning: hardware-in-the-loop-based verification of controller software. In *Proceedings of the 2000 Winter Simulation Conference,* ed. J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, 893-899. Orlando, Florida.

Versteegt, C., and A. Verbraeck. 2002. The Extended Use of Simulation in Evaluating Real-Time Control Systems of AGVs and Automated Material Handling Systems. In *Proceedings of the 2002 Winter Simulation Conference*, 1659-1666, Piscataway, NJ.

Vorderwinkler, M., T. Eder, R. Steringer, and M. Schleicher. 1999. An Architecture for Soft-Commissioning, Verifying control software by linking discrete event simulators to real world control systems. In *Proceedings of the 1999European Simulation Multi conference*, 191-198. Warsaw, Poland.

AUTHOR BIOGRAPHIES

Jeffrey S. Smith is a professor in the Industrial & Systems Engineering department at Auburn University. Prior to joining Auburn, Dr. Smith was an associate professor in the Industrial Engineering Department at Texas A&M University. He received the B.S. in Industrial Engineering from Auburn University in 1986 and the M.S. and Ph.D. degrees in Industrial Engineering from Penn State University in 1990 and 1992, respectively. In addition to his academic positions, Dr. Smith has held industrial positions at Electronic Data Systems and Philip Morris U.S.A. Dr. Smith is an active member of IIE and INFORMS. His email and web addresses are <jsmith@auburn.edu> and <http://sim.auburn.edu/~jsmith>.

Younchol Cho is a doctoral student in the department of industrial and system engineering at Auburn University. He graduated from ROK Naval Academy and received his master degree from Korea National Defense University in Seoul Korea. His email address is <choy-oun@auburn.edu>.