TAMPERE UNIVERSITY OF TECHNOLOGY

Discrete Automation System Design

Assignment 01

Group member:

Farid Khosravi 267964

Mehdi Mahmoodpour 267958

Palash Halder 267962

ASE – 9426 Winter 2017

Contents

Topic: Implementing IEC 61499 for developing distributed automation systems

Objective:

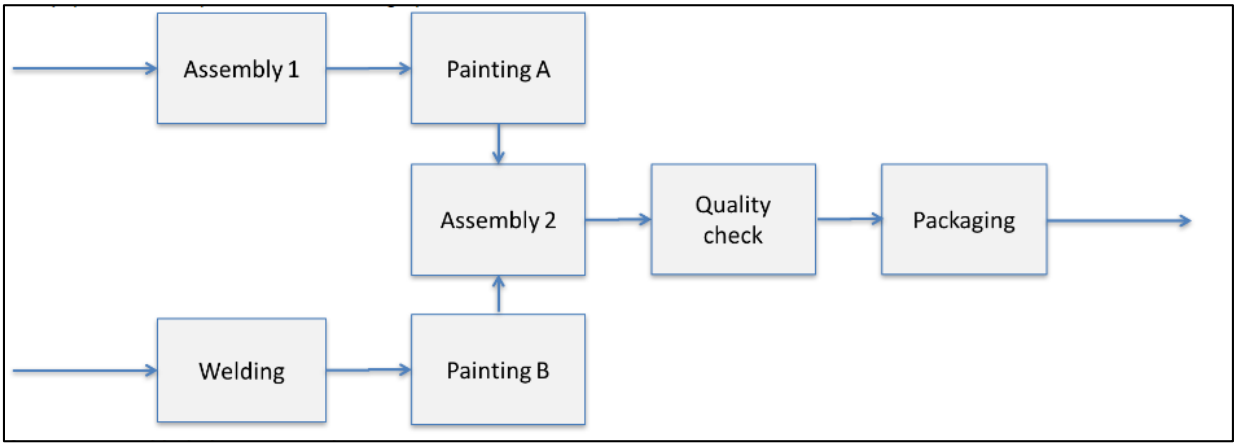
- Design of distributed systems following IEC 61499 standard
- Understanding of event-driven systems
- Implementing basic and composite function blocks
- Knowing and using basic SIFB (e.g. PUBLISH, SUBSCRIBE...)
- Ability to apply the MVC pattern for system design and implementation.
- Know the basic blocks for the various libraries - hmi, control, model, view.

Design:

Concept:

The design of application is based on the following:

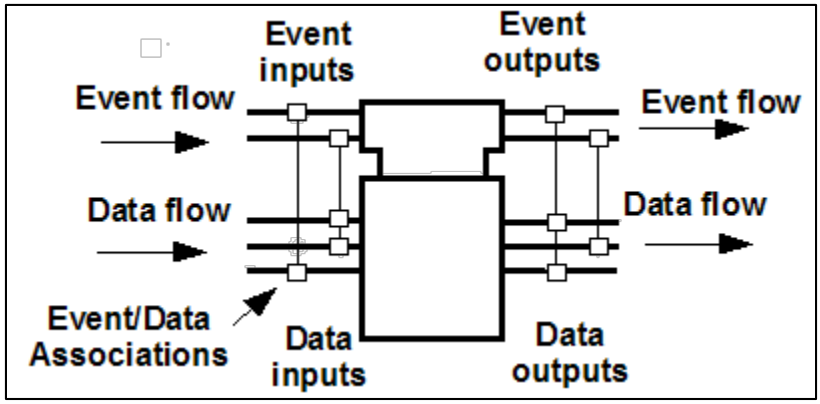
- On running the program the HMI and a view of the industrial system is shown.
- The user must insert the requirements such as the number of products (4 pallets each) and set the conveyor speed.
- The system diagram is as shown below:



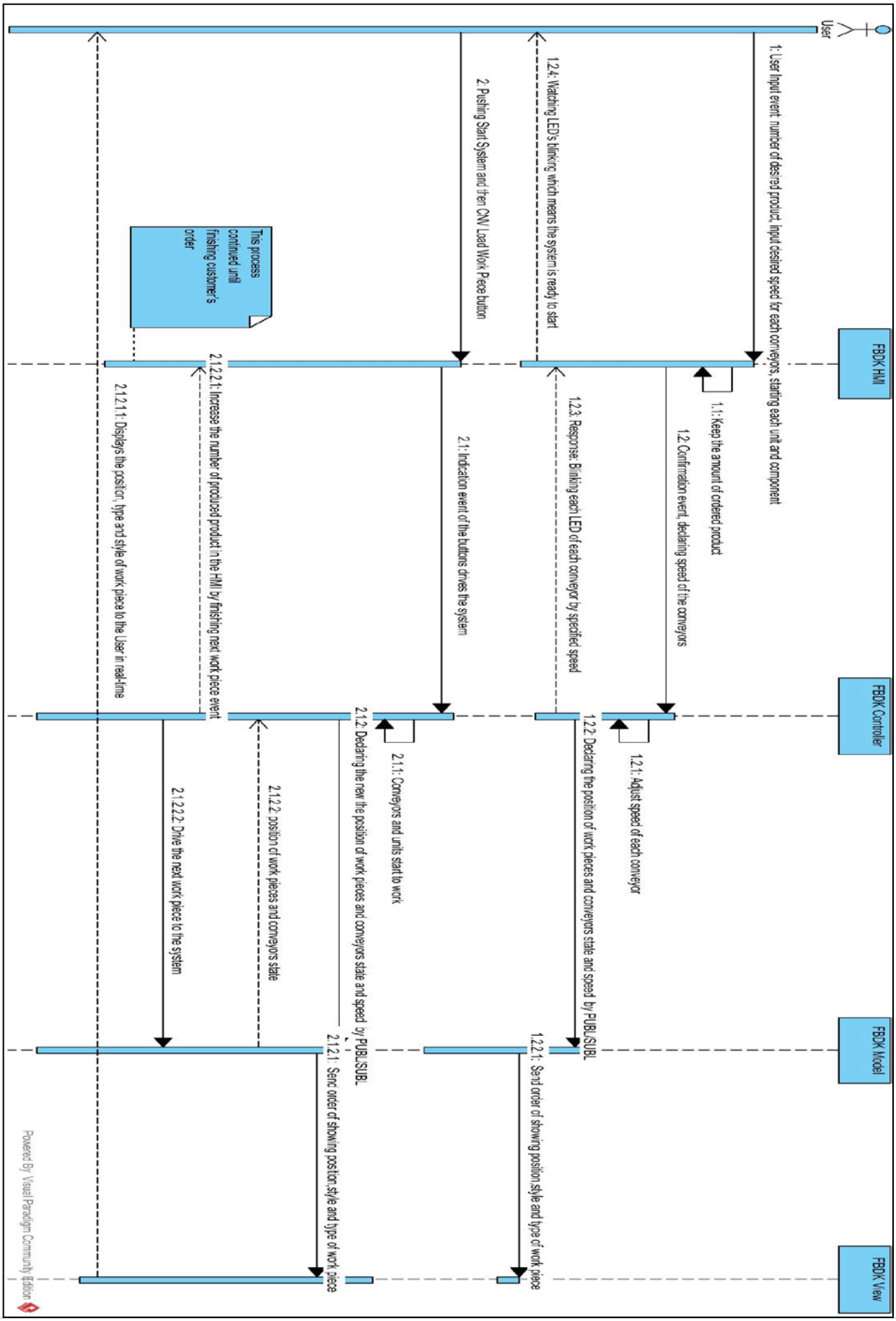
Here, the complete system is divided 7 sub-system which relate to conveyors. The pallet is inserted into the system via Assembly 1 and Welding.

Implementation:

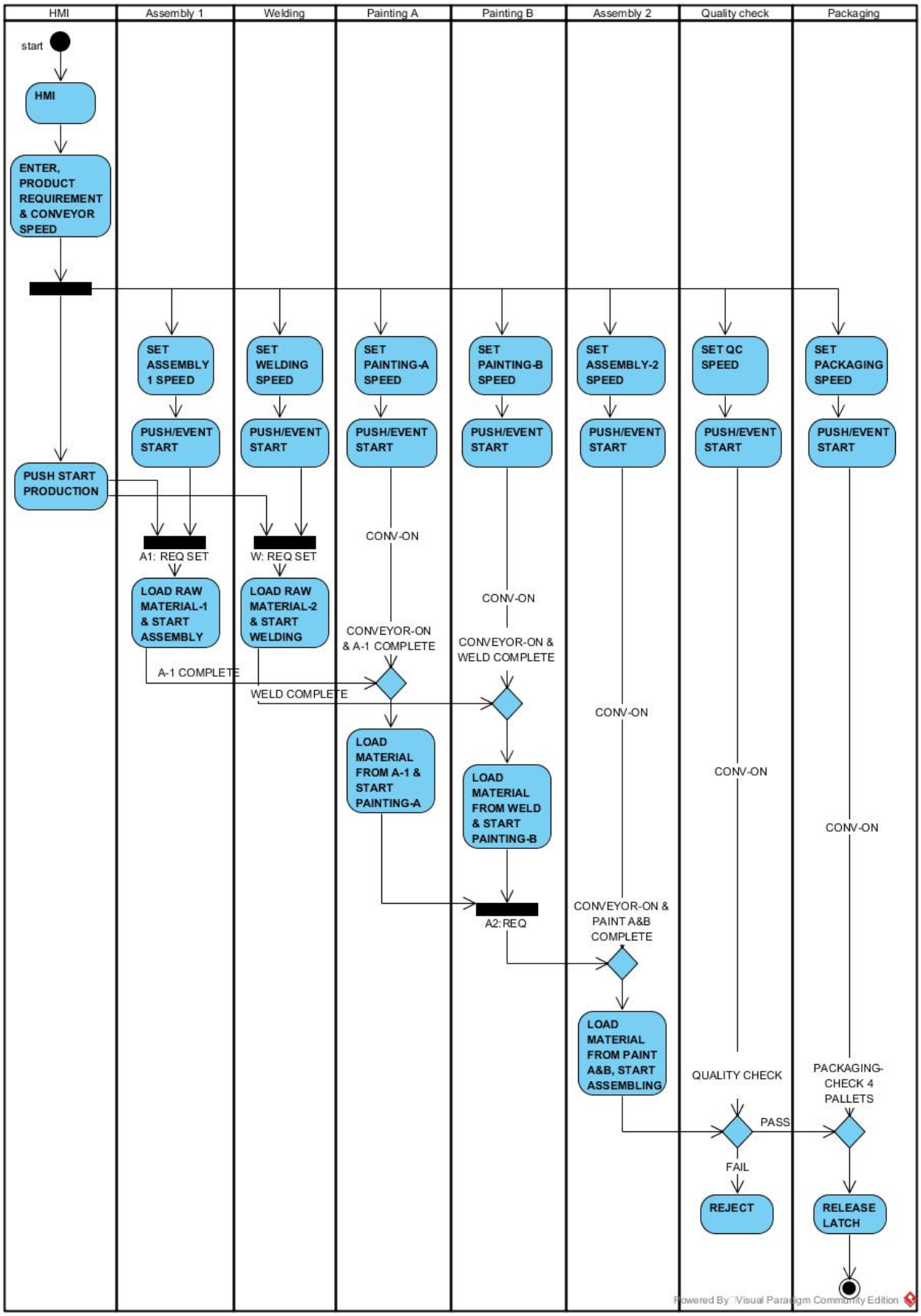
- *Design of distributed systems following IEC 61499 standard*
  - Implementation of IEC 61499 requires implementation of event driven system.
  - Dividing the complete system into 7-individual sub-system.
  - A network of interconnected function blocks defines the sub-system.
  - The next response by the system is dependent on the events generated.
- *Understanding of event-driven systems*
  - IEC 61499 function blocks contain event inputs and outputs in addition to data inputs and outputs.
  - Information about events is passed from one function block to another by means of event variables.
  - Events serve for synchronization and interactions among the execution control mechanisms in interconnected networks of function blocks. A block sending data must emit an event and pass it to the receiving block to ensure the data are read.
  - There is association between events and data input/output, it literally means that the values of the variables associated with an event (and only these!) will be updated when the event occurs.



The sequence diagram depicting the event driven approach of the system is shown in the next page.

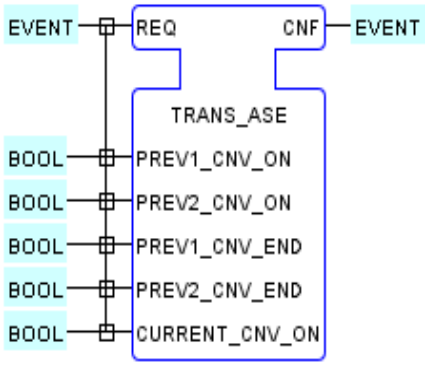

User
FBDK HMI
FBDK Controller
FBDK Model
FBDK View
1: User Input event: number of desired product, input desired speed for each conveyors, starting each unit and component
1.1: Keep the amount of ordered product
1.2 Confirmation event, declaring speed of the conveyors
1.2.1: Adjust speed of each conveyor
1.2.2: Declaring the position of work pieces and conveyors state and speed by PUBLISUBL
1.2.2.1: Send order of showing position,style and type of work piece
1.2.3: Response: Blinking each LED of each conveyor by specified speed
1.2.4: Watching LED's blinking which means the system is ready to start
2: Pushing Start System and then CNV Load Work Piece button
2.1: Indication event of the buttons drives the system
2.1.1: Conveyors and units start to work
2.1.2: Declaring the new the position of work pieces and conveyors state and speed by PUBLISUBL
2.1.2.1: Send order of showing position,style and type of work piece
2.1.2.2: position of work pieces and conveyors state
2.1.2.2.1: Increase the number of produced product in the HMI by finishing next work piece event
2.1.2.2.2: Drive the next work piece to the system
2.1.2.1.1: Displays the position, type and style of work piece to the User in real-time
This process continued until Finishing customer's order
Powered By Visual Paradigm Community Edition

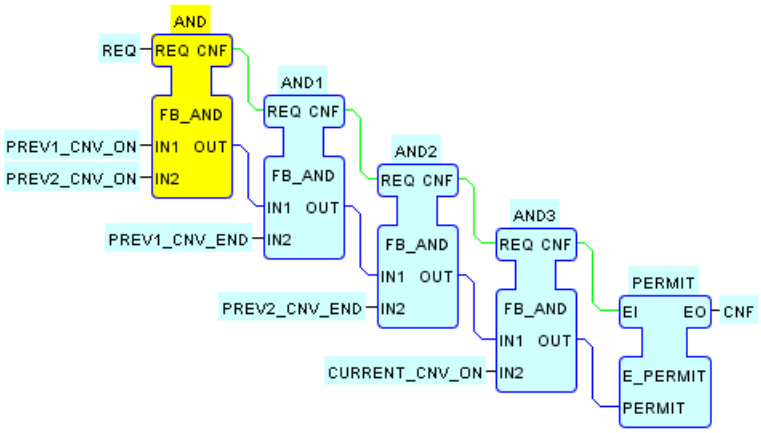
The activity diagram is shown below:



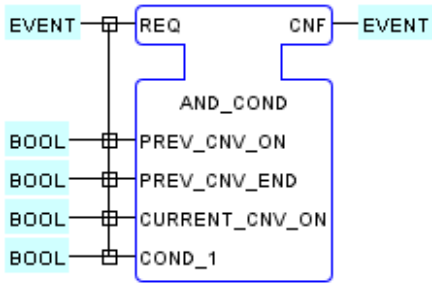
- *Implementing basic and composite function blocks:*

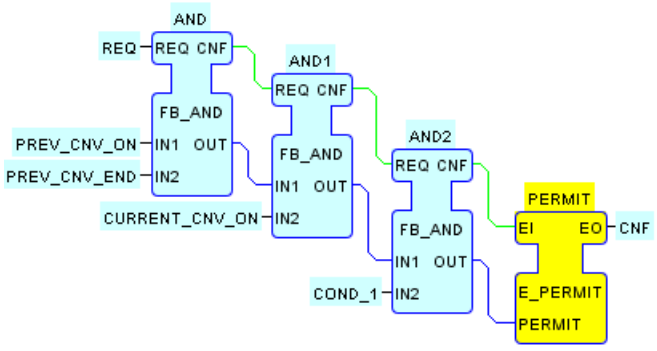
In this assignment, Basic and composite function blocks were used to implement the program. The advantage of using these blocks is to making the program lightweight, fast and organized. Composite blocks are utilized to integrate different blocks with different functionalities into one block to accomplish desired action. Composite and basic function blocks that have been used in this project are shown in the following.



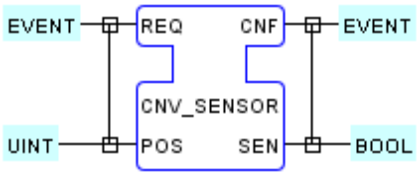


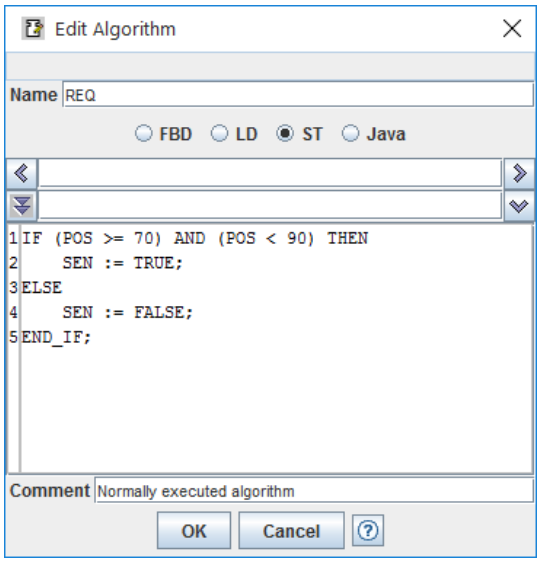
The first composite block was used to emit event for loading last conveyor of assembly and welding work stations. Five conditions were considered to get confirmation event. Previous conveyors and current conveyor should be on and also pallet should be reached at the end of previous conveyors. If all Boolean conditions are **True** then **CNF** event, will be emitted to load next conveyor.





The next composite FB, was utilized to check the conditions of loading conveyor No.6. For loading, previous and current conveyors should be on and pallet should meet the end of previous CNV. Moreover, the another condition that should be taken into consideration is that the part has been not selected by user to be rejected.

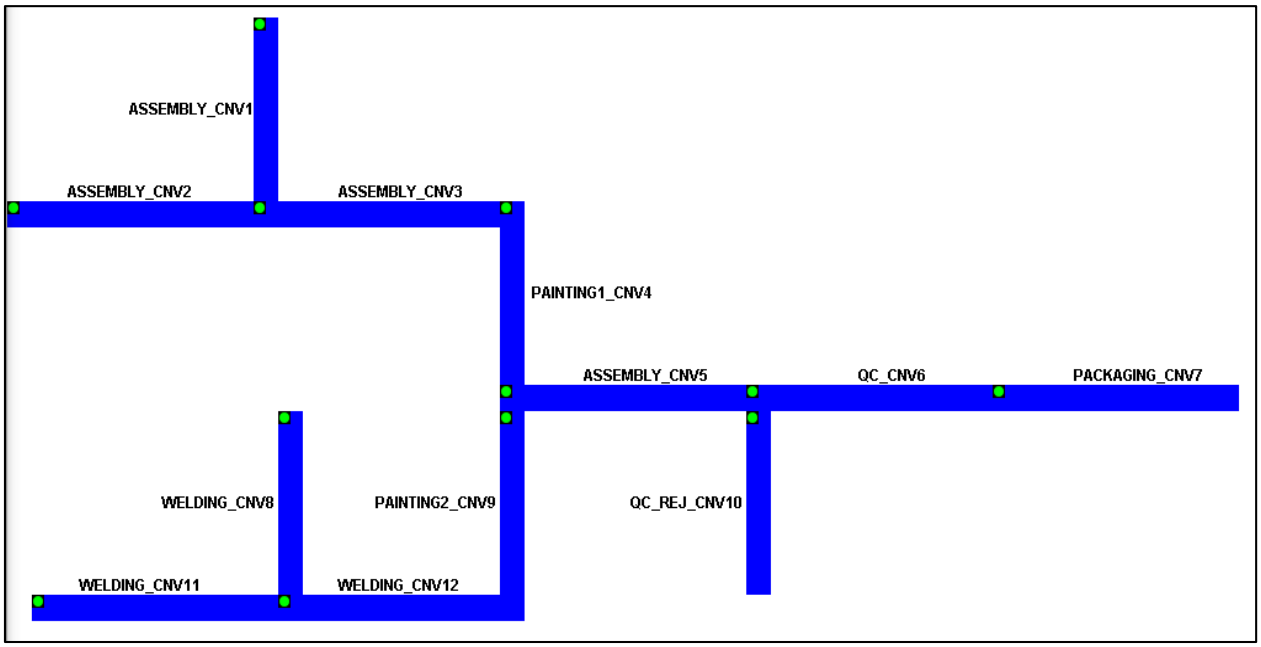




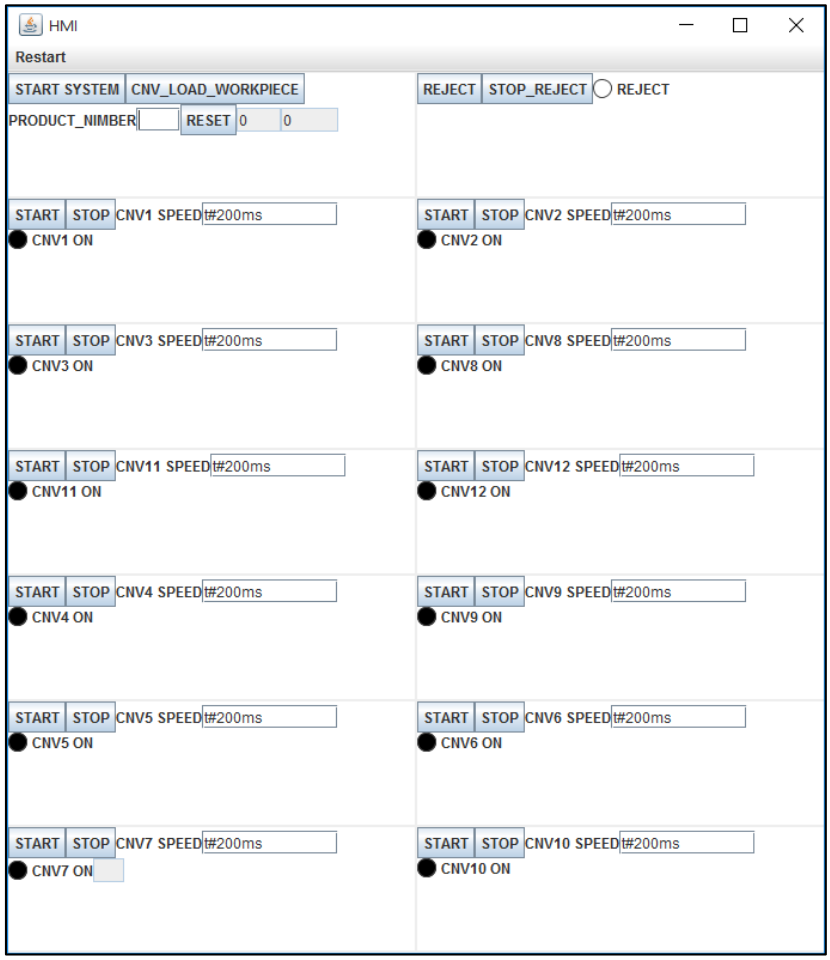
The functionality of above basic function block is to turn on the next conveyor. The position of previous CNV is published and when the position of pallet is between 70 and 90 percent of length of conveyor , then the Boolean output will be set to True. This Boolean value was used to command the next conveyor to be on automatically.

- *Basic SIFB (e.g. PUBLISH, SUBSCRIBE...)*
  - The complete system was divided into individual sub-system and each performed jobs independently.
  - The interaction depends only on the event generated by a system publishing its events, and is followed by the system subscribing to the same.
  - In this assignment, the system is divided into 3-major “devices” namely-HMI, MODEL & VIEW.
  - Each device has “Resources”. These resources are implemented using function block and the interaction of Resources is dependent on the Publisher-Subscriber model.
  - Also, the interaction between devices is dependent of Publisher-Subscriber model.
  - Here, publisher type PUBL_0 and PUBL_1 and subscriber type SUBL_0 and SUBL_1 was used. Where PUBL_0 and SUBL_0 are used only for event subscription and PUBL_1 and SUBL_1 can send one data element with the event.
  - There are other Function blocks which can send more data bit depending on requirement.
  - The publisher-subscriber functionality can be defined by using the same name for PUBL and SUBL blocks or by defining an ID. We have used same name method.

- *Ability to apply the MVC pattern for system design and implementation.*
  - The MVC design, aims at providing an integral solution, which increases the reusability of function block applications.
  - Here, the MVC pattern in implemented as following:
    a. **Model**- The Model device is serving as the model layer, the input from the HMI are the input for Model. The physical is modelled at this layer. All the field events & data such as assembly-1 ended, Painting-B ended are output of this layer. These events & data are forwarded to the controller to generate the next operation to be performed.
    During, start of the system, both data and events are sent from the HMI to controller for processing and generate the initial state of the system.
    The Model device sends events & data to the View device to represent the dynamic state of the system.
    b. **Controller**- the controller layer receives data & events from the model device and the HMI. During initialization, the parametrization at the HMI like "motor speed" is set by the user. The HMI sends these events & data to the controller for setting up the initial parameters.
    The controller sets its clock and generates event & data according the logic defined by programmers of the system. On receiving events and data the model responds accordingly as explained above. The model performs the job and again sends back the events & data back to controller to decide the next operation.
    c. **View**- the View device is View layer; the dynamic status of process is sent with help of events and data from the model layer to view. View layer is necessary to represent the system in a human readable form. With the help of view layer one can predict the progress of process and decide the next steps accordingly.The view of our system is shown below.



d. **HMI**-this layer is in addition to MVC concept, but it is an integral layer as all the user input such as events or data to set the system. The HMI events and data are sent to the controller to initialize the system. Also, some parameter data such as number of products produced are shown at the HMI. All the START and STOP buttons are at HMI to provide scope of manual intervention when required. The HMI of our system is illustrated in the following figure.



- Knowing the basic blocks for the various libraries:
  Here, is a list of some of the important folders and function blocks used.
  - HMI-some of the important blocks used:
    a. IN_EVENT-generate event on user pushing button.
    b. IN_ANY_LBL-get user input like conveyor speed by typing inside a text box.
    c. OUT_ANY-show user the number of pallets/products produced.
    d. OUT_EVENT-show conveyor running status by blinking green LED.
  - MVA-the function blocks from this folder were used to compute results based on user data:
    a. CNV_MDL-this was used to set status of conveyor. The velocity of conveyor, type of disc to be used, start position etc. were defined in this block.

b. MECH_VIEW-this block was used to view the conveyor status.

- NET-this is responsible for transfer of data and events, thus interconnecting the function blocks, PUBL & SUBL blocks were used as explained before.
- EVENTS- responsible for generation of events, division of events:
    a. E_CTU- counter for counting the number of products produced.
    b. E_CYCLE-generating events at regular interval.
    c. E_SPLIT-splitting events so that two function blocks can be requested at the same time.

GROUP MEMBERS' INVOLVEMENT:

- Palash Halder: Design and implementation of HMI layer and recording video.
- Farid Khosravi: Design and implementation of Model layer and writing report
- Mehdi Mahmoodpour: Design and implementation of Control and View layer and writing report