GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

AUTOMATIZACIÓN DE LA MINIFÁBRICA ICAI USANDO SU GEMELO DIGITAL

Autor: Gonzalo Dorao Sánchez-Campos

Director: José Antonio Rodríguez Mondéjar

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Automatización de la minifábrica ICAI usando su gemelo digital

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el curso académico 2020/2021 es de mi autoría, original e inédito y no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

Fdo.: Gonzalo Dorao Sánchez-Campos Fecha: 07/05/2021

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: José Antonio Rodríguez Mondéjar Fecha: 21/06/2021

Firmado por RODRIGUEZ MONDEJAR JOSE ANTONIO - 29057313X el día 05/07/2021 con un certificado emitido por AC FNMT Usuarios

GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

AUTOMATIZACIÓN DE LA MINIFÁBRICA ICAI USANDO SU GEMELO DIGITAL

Autor: Gonzalo Dorao Sánchez-Campos

Director: José Antonio Rodríguez Mondéjar

Madrid

AUTOMATIZACIÓN DE LA MINIFÁBRICA ICAI USANDO SU GEMELO DIGITAL

Autor: Dorao Sánchez-Campos, Gonzalo.
Director: Rodríguez Mondéjar, José Antonio.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

Se ha creado un gemelo digital de la minifábrica de ICAI en RobotStudio para minimizar los ajustes necesarios al pasar la programación a la minifábrica física. Además, se ha realizado una automatización de un pick&place de piezas de Lego como demostración de su funcionamiento.

Palabras clave: Automatización, minifábrica, digitalización, gemelo digital, pick&place.

1. Introducción

Incorporar brazos robóticos en cadenas de montaje o fabricación en una industria implica un gran aumento del rendimiento, la eficiencia y mejora de los tiempos de procesado. Cualquier tarea repetitiva que pueda ser realizada por un brazo robótico en lugar de por un humano será completada en una fracción del tiempo y reduciendo gastos en recursos humanos. Sin embargo, estas plantas robotizadas plantean un inconveniente que afecta a la flexibilidad de la producción. En caso de querer cambiar la programación de la planta y de los robots, es necesario parar los procesos y volver a configurar la estación para que siga las nuevas rutas y lea los nuevos programas. Es por esto por lo que durante los últimos años muchas plantas han estado optando por programar en entornos virtuales, probando los nuevos programas en gemelos digitales de la cadena de procesado para no perturbar el ritmo habitual de la planta. De esta manera, una vez terminado y depurado el nuevo programa, se puede parar la planta durante los escasos minutos necesarios para subir el nuevo programa al sistema y continuar con la producción.

2. Definición del proyecto

El objetivo de este proyecto es crear un gemelo digital de la minifábrica ICAI usando la herramienta RobotStudio de tal manera que se reduzcan al mínimo los ajustes a realizar cuando se traslade la programación a la minifábrica física. El objetivo es automatizar la minifábrica para que realice un pick&place de piezas de Lego suministradas desde un cargador o desde la cinta transportadora. Se ha completado el gemelo digital de la minifábrica y se ha realizado una automatización del pick&place como demostración de su funcionamiento.

3. Descripción del modelo

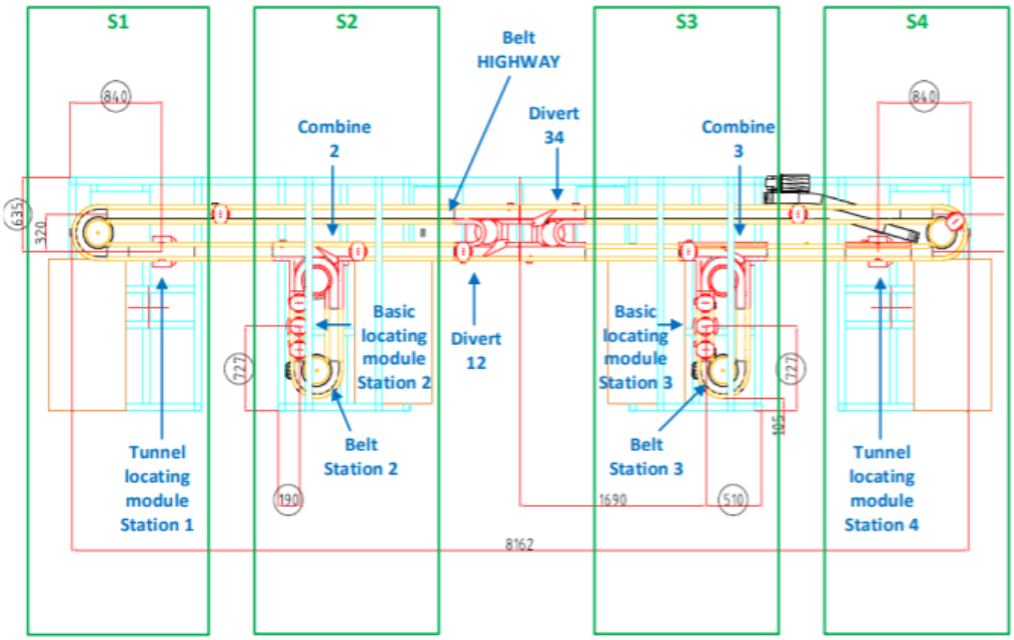

Figura 1 - Plano de la minifábrica de ICAI

La minifábrica (Figura 1) está dividida en cuatro subestaciones (S1-S4) por las que pasan los palés sobre los que se trabaja. En cada una de ellas hay un robot IRB 120 apoyado en una mesa (S1 y S4) o fijado del revés a una estructura metálica (S2 y S3).

Para asegurar que el palé no se mueve durante la operación del robot, cada subestación tiene un mecanismo de localización. En el caso de las estaciones S1 y S4 el mecanismo de localización bloquea un palé y lo eleva, permitiendo que pasen palés por debajo mientras el robot trabaja. Este mecanismo se puede ver en la Figura 2. Los localizadores de las estaciones S2 y S3 se pueden ver en la Figura 3 y simplemente tienen una función de bloqueo. Todos los mecanismos de localización cuentan además con un retenedor previo y otro posterior para controlar el flujo de palés, así como un sensor inductivo para detectar la llegada de los mismos.

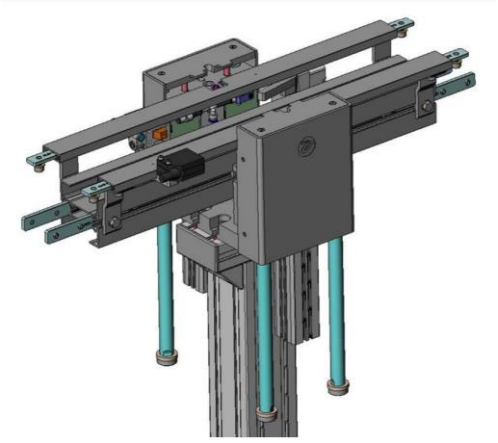

Figura 2 - Posicionador de túnel XBUL 11 T

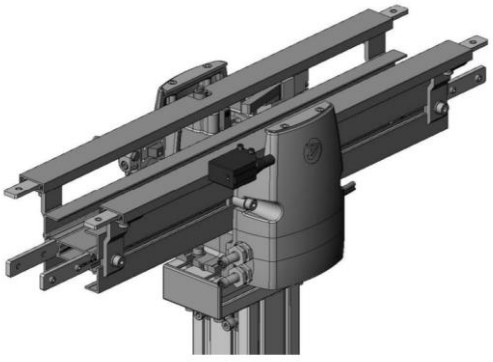

Figura 3 - Posicionador de túnel XBUL 11

Además de estos, también hay varios mecanismos que sirven para controlar la dirección del flujo de palés. Los Combine 2 y 3 se encargan de aceptar los palés que se acercan a las subestaciones S2 y S3 o de expulsar los que ya están en ellas. El Divert 12 se encarga de que los palés circulen solo por las subestaciones S1 y S2 o de dejarles vía libre para que recorran las cuatro. El Divert 34 hace lo mismo con S3 y S4. Todos estos mecanismos cuentan con retenedores previos y posteriores, así como sensores inductivos, al igual que ocurre con los mecanismos de localización. Además, antes de entrar en S1 hay un retenedor general llamado GSTOP que controla el flujo global de los palés.

La estación funciona con un controlador para cada robot y dos PLC para las señales de entrada y salida de la planta. Estos PLC a su vez están conectados a tres remotas de entradas y salidas digitales Siemens ET200SP que se encargan de distribuir las señales por la estación. En el gemelo digital, sin embargo, todas las señales se controlan con el controlador del robot de S1 por motivos de simplicidad. Se podrá programar la configuración real de las señales como futuro trabajo, pero el funcionamiento es el mismo.

No se realiza una simulación física, pero la lógica de la estación frena los palés cuando detecta colisiones y reanuda la marcha cuando se despeja el camino, igual que pasaría en la planta real.

4. Resultados

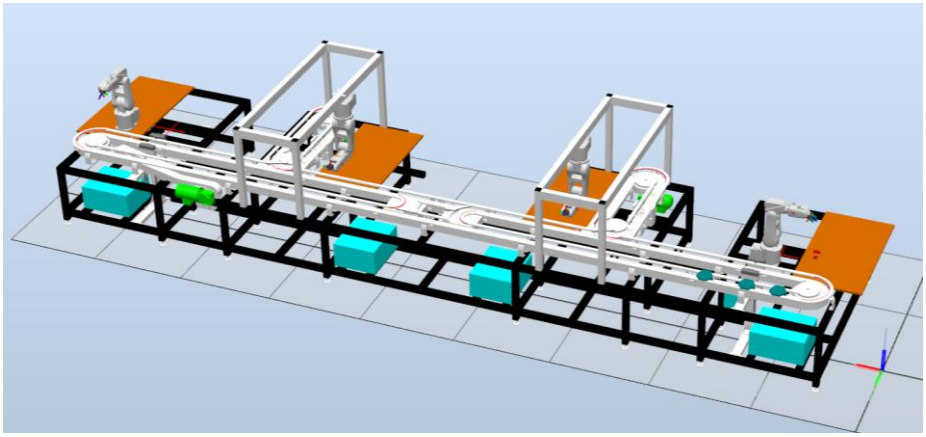

Figura 4 - Gemelo Digital

Se ha conseguido desarrollar un gemelo digital (Figura 4) funcional en el que se puede programar la lógica de la estación mediante código RAPID. Se puede trabajar con tantos palés como quepan físicamente en la estación y se puede cambiar la programación de los cuatro robots IRB120 y de la estación mediante código RAPID o con las herramientas visuales de la herramienta RobotStudio.

5. Conclusiones

Gracias al gemelo digital, cualquier usuario con una licencia de RobotStudio será capaz de escribir un programa en RAPID para modificar la lógica de la minifábrica.

6. Referencias

[1] Conrad Leiva. “Demystifying the Digital Thread and Digital Twin Concepts”. Industry Week 2016. August, 2016. https://www.industryweek.com/technology-and-iiot/systems-integration/article/22007865/demystifying-the-digital-thread-and-digital-twin-concepts.

[2] Kennedy, Kara. “What is Digital Twins (+Impact on Business Modernization)”, Learn Hub 2018. January 2018. http://learn.g2.com/trends/digital-twins.

[3] Pettey, Christy. “Prepare for the impact of Digital Twins”. Smarter With Gartner 2017. September 2017. https://www.gartner.com/smarterwithgartner/prepare-for-the-impact-of-digital-twins/

[4] “RobotStudio: La herramienta de programación offline más utilizada del mundo en robótica”. ABB 2020. https://new.abb.com/products/robotics/es/robotstudio

ICAI'S MINIFACTORY AUTOMATIZATION USING ITS DIGITAL TWIN

Author: Dorao Sánchez-Campos, Gonzalo.
Supervisor: Rodríguez Mondéjar, José Antonio.
Collaborating Entity: ICAI – Universidad Pontificia Comillas.

ABSTRACT

The aim of this project is to create a digital twin for ICAI's mini factory using the software RobotStudio to minimize the necessary adjustments for the deployment of the code to the physical mini factory. A basic automatization of the factory was designed to illustrate how the Digital Twin performs a pick&place operation with Lego bricks.

Keywords: Automatization, minifactory, digitalization, digital twin, pick&place.

1. Introduction

The introduction of robotic arms in assembly lines implies an improvement in performance, efficiency, and processing times. Any repetitive task that can be done by a robot will be completed by it in a fraction of the time it would take a human to do the same task while also reducing personnel costs. However, these robotic stations lack flexibility. In order to change the station's program, all the processes must be stopped, and the station must be reconfigured to follow the new instructions. Therefore, during the last years many stations have been programmed in virtual environments, testing the new routines on digital twins to avoid slowing down the station's normal production flow. This allows programmers to write the code, test it in a virtual environment and finally stop the station for just a few minutes while the final code is being deployed on the physical station.

2. Project summary

The aim of this project is to create a Digital Twin for ICAI's minifactory using RobotStudio to minimize the necessary adjustments when the program is deployed to the physical minifactory. The objective is to automate the factory to perform a pick&place operation using Lego bricks fed from a feeder or from the conveyor. Apart from completing the Digital Twin, an automatization was designed to illustrate how the Digital Twin performs the pick&place operation.

3. Model description

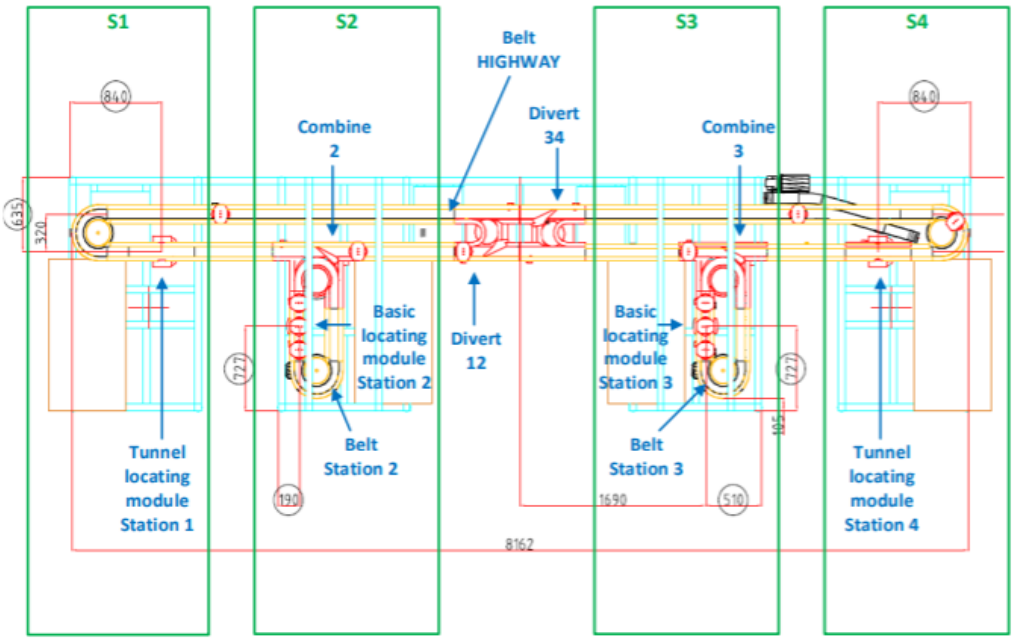

Figure 5 - ICAI's minifactory schematics

The minifactory (Figure 5) is divided into 4 substations (S1-S4) through which the pallets will flow and be worked on by the robots. In each of the substations there is a IRB120 robot, which are fixed on a table (S1 and S4) or upside down attached to a metallic structure (S2 and S3).

To ensure that the pallet is not moving during the robot's operation, each substation has a locking mechanism. For stations S1 and S4 this mechanism locks the pallet and lifts it, allowing other pallets to pass underneath it while the robot is working. This mechanism can be seen in Figure 6. The locking mechanisms in S2 and S3 can be seen in Figure 7 and only have a locking function. All locking mechanisms have a stopper and a pre-stopper to control the pallet flow, as well as an inductive sensor to detect the arrival of said pallets.

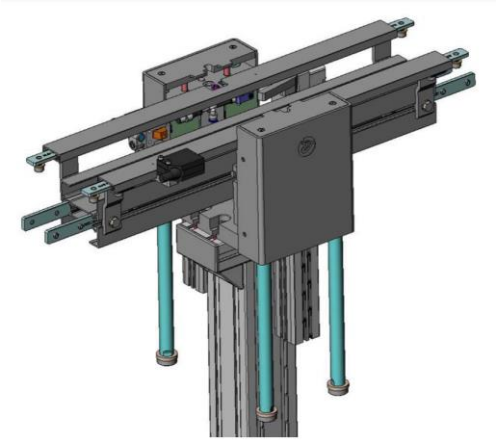

Figure 6 - Tunnel locating module XBUL 11 T

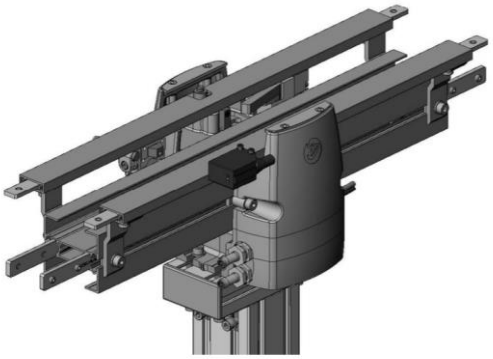

Figure 7 - Basic locating module XBUL 11

Apart from these mechanisms, there are several others that control the pallet flow direction. The Combine 2 and 3 accept or reject the pallets that approach substations S2 and S3 or the ones that are already in them. The Divert 12 makes the pallets flow only through S1 and S2 or through the whole station. Divert 34 does the same with S3 and S4. As with the locking mechanisms, these also have stoppers, pre-stoppers, and inductive sensors. Furthermore, just before S1 there is a stopper (GSTOP) that controls the global pallet flow.

The station has one controller for each robot, as well as two PLC for the inputs and outputs of the station. However, in the Digital Twin all signals are controlled from the S1 controller for simplicity reasons. The real signal configuration can be programmed in the future, but the performance will be the same.

There is no physical simulation, but the station's logic stops the pallets when they collide with objects or with each other and makes them move again when the path is clear, just as it would happen in the actual physical station.

4. Results

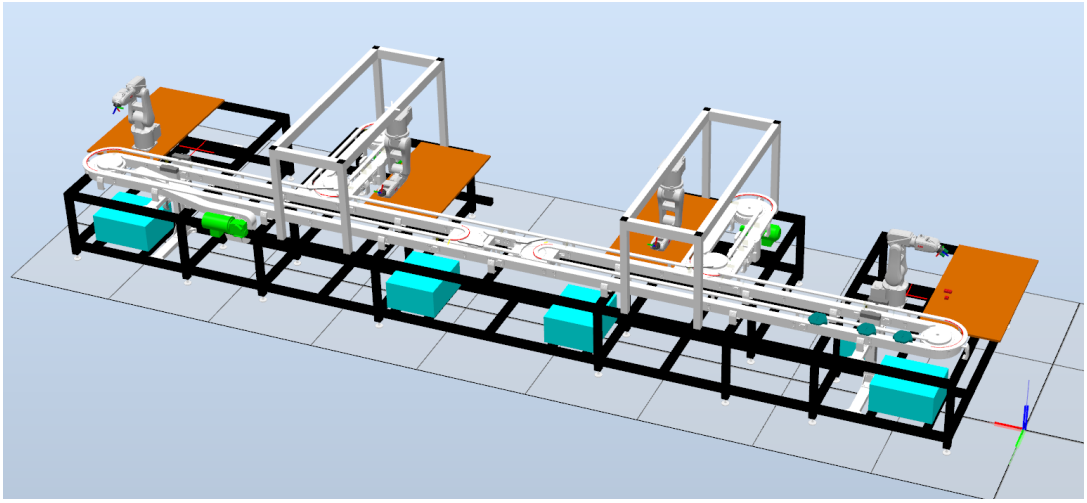

Figure 8 - Digital Twin

A functional Digital Twin (Figure 8) was developed. Its logic can be programmed using RAPID code. It can work with as many pallets can fit in the station and the robots can be reprogrammed using either RAPID code or RobotStudio's graphic tools.

5. Conclusions

This Digital Twin will allow anyone with a RobotStudio license to write a RAPID code to reprogram the minifactory's logic.

6. References

[1] Conrad Leiva. “Demystifying the Digital Thread and Digital Twin Concepts”. Industry Week 2016. August, 2016. https://www.industryweek.com/technology-and-iiot/systems-integration/article/22007865/demystifying-the-digital-thread-and-digital-twin-concepts.

[2] Kennedy, Kara. “What is Digital Twins (+Impact on Business Modernization)”, Learn Hub 2018. January 2018. http://learn.g2.com/trends/digital-twins.

[3] Pettey, Christy. “Prepare for the impact of Digital Twins”. Smarter With Gartner 2017. September 2017. https://www.gartner.com/smarterwithgartner/prepare-for-the-impact-of-digital-twins/

[4] “RobotStudio: La herramienta de programación offline más utilizada del mundo en robótica”. ABB 2020. https://new.abb.com/products/robotics/es/robotstudio

Índice de la memoria

Índice de figuras

Capítulo 1. INTRODUCCIÓN

Desde que existe la actividad industrial, siempre se ha tendido a mejorar los procesos de producción para aumentar su rendimiento y la calidad de la producción a la vez que se abaratan costes y se acortan los tiempos de procesado y mantenimiento.

La primera Revolución Industrial sustituyó el trabajo humano físico por el trabajo mecánico de máquinas más eficientes. Estas máquinas han ido adquiriendo una calidad, precisión y rapidez crecientes exponencialmente con el paso de las décadas. Si bien este fenómeno se va a seguir produciendo, la revolución ante la que nos encontramos en la actualidad es la de la Industria 4.0, que no pretende cambiar tanto las máquinas que se utilizan en los procesos sino la organización mediante la que se dirigen.

Esta Revolución Industrial apuesta por el Internet de las Cosas (IoT, según las siglas en inglés de *Internet of Things*), la mano de obra robótica, la coordinación de tareas mediante inteligencia artificial y el mantenimiento predictivo mediante algoritmos de predicción de averías. Todos estos cambios supondrán una reducción del número de personas trabajando manualmente en una fábrica a la vez que se aumenta la producción y la eficiencia, lo cual bajará sustancialmente los costes de producción.

La pérdida de trabajos por culpa de esta revolución es un tema de debate que queda pendiente para otro texto, pero como breve reflexión se podría recordar la situación posterior a las anteriores Revoluciones Industriales, en las que los empleos de millones de personas se vieron drásticamente alterados. Aun así, la economía se acabó estabilizando sin tasas de paro significativamente mayores. Sin embargo, el problema que viene a solucionar este Proyecto de Fin de Grado es otro muy distinto.

En un taller o una fábrica con personal humano, un director puede dar nuevas instrucciones a los trabajadores para que cambien sus rutinas con cierta facilidad, siempre que no se requiera un nuevo entrenamiento de la mano de obra. En el caso de plantas robotizadas, desgraciadamente no es así. El modelo que se ha empleado durante los últimos años ha consistido en parar la producción, reprogramar los robots, probar el nuevo programa y finalmente retomar la producción con el programa actualizado. Estos procesos pueden tener la producción parada durante semanas o incluso meses en el peor caso, motivo por el cual se ha creado el concepto de Gemelo Digital.

Un Gemelo Digital es una copia virtual de la fábrica que se quiere reprogramar. En él se puede cambiar el programa de los robots y de la planta en general, simular la producción y corregir errores sin necesidad de detener la producción en la fábrica física real. Una vez diseñado y depurado el programa, este se puede aplicar a la planta en cuestión de minutos o segundos, reduciendo al mínimo las pérdidas económicas por la parada de la fábrica.

La minifábrica de ICAI no es una excepción, y también es deseable tener un Gemelo Digital en el que alumnos y profesores puedan trabajar sin necesidad de interrumpir la actividad que esté realizando la minifábrica real. En este caso no existe el problema de las pérdidas económicas, sino que el propósito del diseño de este Gemelo Digital es que varias personas puedan trabajar en el mismo modelo de minifábrica al mismo tiempo sin interferirse unas a otras, ya que cada una tendría su propio Gemelo Digital en su ordenador. Por supuesto, para probar el código en la planta real tendrán que turnarse, pues en la minifábrica física solo puede ejecutarse un código al mismo tiempo.

Para que la simulación sea lo más fiel posible a la realidad, el modelo tiene que ajustarse con precisión a la minifábrica física. Es por esto por lo que este Gemelo Digital se ha diseñado con extremo cuidado y tratando de representar la realidad de la forma más veraz posible en el tiempo limitado del que se ha dispuesto. Esto se ha traducido en una simulación rigurosa a cambio de ciertas simplificaciones en la lógica de la estación que serán comentadas en los próximos puntos. Todas estas simplificaciones podrán refinarse en el futuro para que la minifábrica digital sea idéntica a la real tanto física como lógicamente, aunque su estado actual es funcionalmente idéntico al real.

Capítulo 2. DESCRIPCIÓN DE LAS TECNOLOGÍAS

2.1 ROBOTSTUDIO

RobotStudio es una herramienta de programación y simulación de ABB que permite programar estaciones robotizadas desde un ordenador sin interrumpir sus tareas. Las simulaciones se realizan con el mismo software que utilizan los robots de ABB (ABB VirtualControler), con lo que se consiguen simulaciones muy realistas.

2.2 IRB 120

El IRB 120 (Figura 9) es un brazo robótico con 6 grados de libertad y velocidad máxima de 3m/s. Es capaz de transportar 3kg (incluyendo la herramienta que se le acople) y tiene un alcance máximo de 0,58m.

Figura 9 - IRB 120

Su área de trabajo (Figura 10) es extensa, aunque tiene zonas muertas cercanas a la base y en la parte trasera de la máquina.

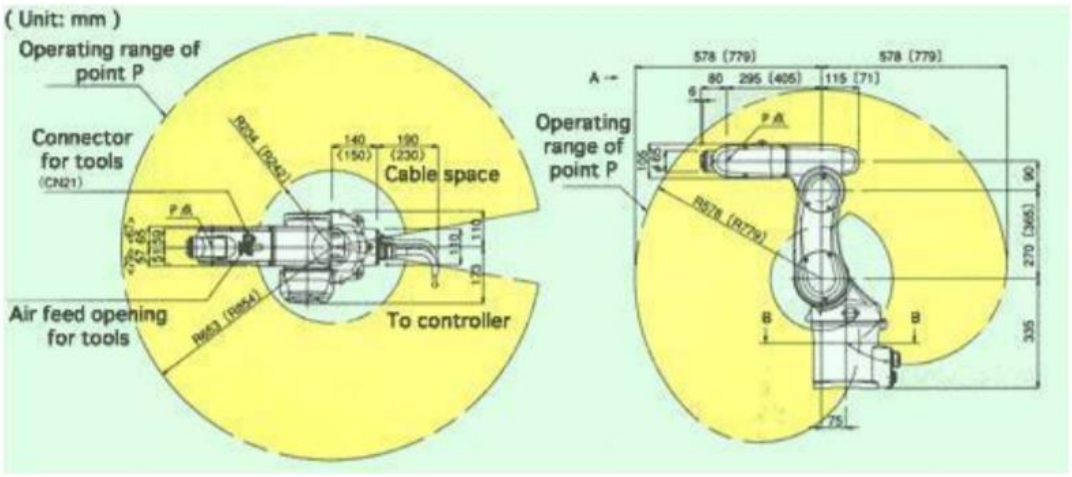

Figura 10 - Área de trabajo de IRB 120

2.3 CONTROLADOR LÓGICO PROGRAMABLE (PLC)

Un PLC (Controlador Lógico Programable o *Programmable Logic Controller*) son ordenadores cuyo objetivo es controlar las señales utilizadas en diversos procesos de ingeniería y manufacturación.

En el caso de la minifábrica de ICAI se utilizan dos PLC Siemens CPU 314C-2 PN/DP (Figura 11); uno para las subestaciones S1 y S2 y otro para las S3 y S4. Sirven para controlar las válvulas de los mecanismos de la planta.

Figura 11 - PLC Siemens CPU 314C-2 PN/DP

Capítulo 3. ESTADO DE LA CUESTIÓN

3.1 EJEMPLOS DE DIGITAL TWIN O GEMELO DIGITAL

Los gemelos digitales se están usando en todo el mundo en máquinas de todo tipo. Sirven para la planificación de proyectos de fabricación industrial, aeronáuticos, de tecnología orientada al sector sanitario o incluso al modelado de redes de distribución de agua o diversas obras civiles.

Entre las empresas más importantes que utilizan este tipo de modelos con regularidad se encuentran Maserati, Airbus, Siemens o Daimler entre muchos otros. Tras el gasto inicial de crear este Gemelo Digital, los beneficios amortizan la inversión en la gran mayoría de los casos. La versatilidad, comodidad y reprogramabilidad de los Gemelos Digitales hacen de ellos la herramienta perfecta en las fases de preparación de cualquier proyecto de ingeniería o que en general requiera labores de diseño, programación y simulación.

3.2 MODELO 3D DE LA MINIFÁBRICA

Al comenzar el Proyecto de Fin de Grado se contaba con un modelo 3D de la minifábrica de ICAI en formato .dwg modificable en CAD (Figura 12). Este modelo es bastante fiel a la realidad ya que es el que se había utilizado en el diseño y montaje de la minifábrica física.

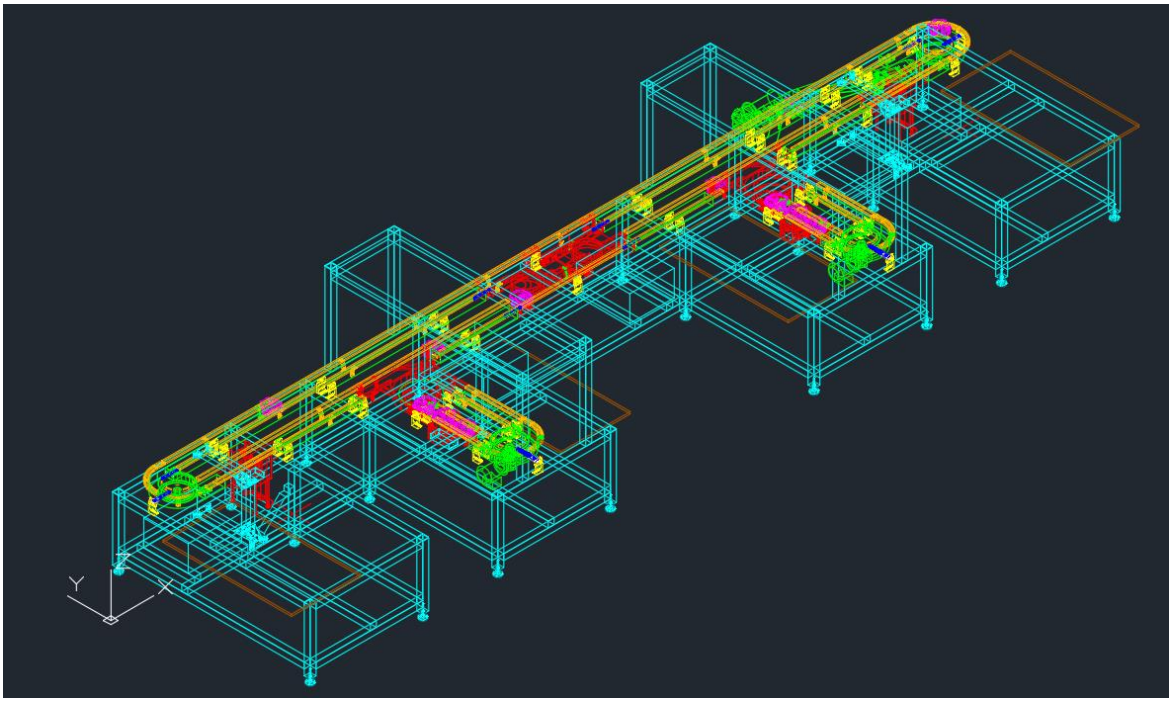

Figura 12 - Modelo 3D de la minifábrica

Sin embargo, al exportar la geometría a RobotStudio se producían varios errores menores de colisión de geometrías y partes de raíles borradas o mal orientadas. El motivo de estos errores se desconoce, pero se ha arreglado mediante el editor de geometrías incluido en RobotStudio.

El mayor problema fue que al importarla se comportaba como un único objeto sólido, por lo que no se disponía de los modelados de las piezas individuales de los mecanismos que haría falta utilizar. La solución a este problema se explica en el apartado 4.3.1.

Capítulo 4. DEFINICIÓN DEL TRABAJO

4.1 JUSTIFICACIÓN

El objetivo de este trabajo es diseñar un Gemelo Digital para facilitar la programación de la minifábrica de ICAI a profesores, alumnos de máster o quien lo precise. RobotStudio permite realizar la programación de manera virtual tanto *online* como *offline* y da la opción de usar controladores virtuales o reales según se requiera. Trabajar con un Gemelo Digital de la minifábrica en RobotStudio es la manera más eficiente de aprender a programar los robots de una fábrica real. Además, este método permite que varias personas estén trabajando en la misma planta sin interferirse entre sí.

4.2 OBJETIVOS

Para que el Gemelo Digital resulte satisfactorio han de cumplirse los siguientes objetivos.

4.2.1 CORRECCIÓN DEL MODELO 3D

El modelo 3D de la minifábrica debe ser exactamente igual que el modelo físico. Para ello habrá que posicionar todas las piezas desviadas en su lugar correspondiente, eliminar las geometrías sobrantes y añadir las geometrías necesarias para terminar la composición. Estos fallos se concentran alrededor de la curva de la subestación S4 y en los raíles de toda la estación.

4.2.2 MODELADO DE LOS MECANISMOS

Los mecanismos deben comportarse como lo harían en la minifábrica física. Deben recibir las mismas señales y responder a ellas de la misma manera, incluyendo recorridos y tiempo de ejecución. Estos mecanismos son las guías que modifican la trayectoria de los palés (2 Divert y 2 Combine), retenedores (9 previos y 4 posteriores), localizadores (2 de túnel y 2 simples), y las herramientas con las que agarrar las piezas de Lego (4 pinzas).

4.2.3 DISEÑO DE LA LÓGICA DE LA ESTACIÓN

La lógica de la estación física requiere multitud de señales, aunque su funcionamiento no es complicado. Los palés siguen el recorrido siguiendo el camino que marcan las guías y se detienen cuando se encuentran bloqueados por un localizador, un retenedor u otro palé detenido que bloquea el paso. Los mecanismos están controlados por válvulas neumáticas de doble efecto que reciben dos señales binarias. En caso de recibir 00 o 11 se mantienen en su estado anterior. Al recibir 01 se pondrán en su Estado 1 si es que no estaban ya en él y al recibir 10 harán los mismo con el Estado 2.

El problema de la lógica de la estación en el Gemelo Digital radica en que no existe una simulación física real, ya que RobotStudio solo permite simulaciones físicas en cintas transportadoras que consistan en una línea recta. Debido a este motivo se debe crear una lógica que dicte cuando los palés deben avanzar en línea recta, frenarse, realizar curvas o cambiar su altura según la parte del recorrido en la que se encuentre, los obstáculos que haya en el camino, el estado de las señales y la posición de los mecanismos. De esta manera, aunque no se esté simulando la física de los objetos, se estará imitando su comportamiento real.

4.2.4 CREACIÓN DE UN PROGRAMA DE DEMOSTRACIÓN DE PICK&PLACE

Para demostrar cómo funciona el sistema en su conjunto se pretende hacer un pequeño programa de demostración que muestre la minifábrica en funcionamiento. Ya que la minifábrica está diseñada para el Pick&place de piezas de Lego, se simulará como los robots cogen piezas con sus pinzas y las colocan en los palés. El programa se realizará de forma modular, es decir, se creará una función a la que se le especifique el tipo de pieza que se quiere usar y su posición final y el controlador del robot calculará las rutas necesarias. De esta manera se podrán programar construcciones complejas sin necesidad de definir puntos adicionales en RobotStudio ni programar trayectorias con dichos puntos.

4.3 METODOLOGÍA

4.3.1 DISEÑO DE GEOMETRÍAS NECESARIAS

Las geometrías se obtendrán utilizando las operaciones de modificación de geometría de RobotStudio. Se necesitarán tres operaciones fundamentalmente:

1. Intersección. Se cubrirán las partes del objeto 3D que representen un mecanismo con un cubo u otra geometría simple creada desde RobotStudio. Al utilizar la herramienta intersección se creará un nuevo elemento independiente del objeto 3D anterior en el lugar de la intersección entre el objeto 3D de la planta y el objeto 3D auxiliar. Este elemento será el mecanismo con todas sus piezas juntas en un único objeto 3D. Las piezas del mecanismo se separarán entre sí siguiendo el mismo procedimiento hasta que todas las piezas estén separadas, momento en el cual se podrá generar el mecanismo y permitir su movimiento. Es posible que esta segunda operación requiera figuras tridimensionales auxiliares más complejas que un simple cubo debido a las geometrías más complicadas de las piezas.
2. Extrusión. Necesaria para crear las piezas auxiliares y los raíles que falten en la estación.
3. Dibujar línea. Será necesario dibujar rectas y curvas como referencias para otras geometrías y para guiar las curvas que han de realizar los palés. Sobre el comportamiento de los palés sobre las curvas se hablará con más detalle en el apartado 5.2. Las guías y referencias con líneas serán necesarias para que el modelo de la planta sea matemáticamente exacto al del diseño físico y no una simple aproximación.

4.3.2 MECANISMOS EN ROBOTSTUDIO

RobotStudio tiene una herramienta llamada Crear Mecanismos que cumple con este propósito. Para los mecanismos con rotación como las guías o los retenedores posteriores se usan al menos dos piezas y se especifican un eje de giro y un ángulo mínimo y máximo. Para los mecanismos lineales como los elevadores o los retenedores previos se seleccionan las

piezas, se eligen ejes lineales y se especifican las coordenadas mínima y máxima del movimiento.

4.3.3 DISEÑO DE SMARTCOMPONENTS DE MANERA MODULAR

SmartComponents es una herramienta que ofrece RobotStudio para programar la lógica de la estación. Cada SmartComponent tiene una serie de entradas y salidas digitales y propiedades que pueden ser números, valores booleanos u objetos de la estación entre otros.

Estos SmartComponents se pueden anidar unos dentro de otros y estar conectados entre sí. De esta manera se hacen módulos lógicos que se pueden utilizar en diversas partes del programa. En este proyecto se utilizará un SmartComponent general dentro del cual habrá otros SmartComponent. El general será el encargado de intercambiar señales con el controlador virtual de los IRB120, mientras que los SmartComponents interiores se encargarán de procesar la información recibida por el general para controlar la simulación y devolver al SmartComponent general las señales que deben volver al controlador, como es el caso de los sensores inductivos repartidos por toda la planta o la posición de ciertos mecanismos como los elevadores.

El SmartComponent interno más importante será el de Palé. Servirá para controlar el movimiento del palé, tanto tramos rectos como curvos además de colisiones, paradas, cambios de velocidad y puestas en marcha. Será un componente clonable, ya que la estación real trabaja con multitud de palés simultáneamente.

4.3.4 CREACIÓN DE RUTAS Y LÓGICA PARA LA DEMOSTRACIÓN

Para la demostración tendrá que escribirse un código en RAPID, que es el lenguaje que utilizan las controladoras de ABB para dirigir las señales del robot y de la planta. Este código coordinará los movimientos del robot y hará que ponga piezas de Lego en los palés cuando estos estén posicionados correctamente. Para ello el controlador leerá las señales del SmartComponent general y devolverá señales para que la simulación se ejecute de manera correcta.

Como resumen esquemático del funcionamiento de la demostración, el código RAPID se ejecuta en el controlador, que a su vez se comunica con el SmartComponent general, que es el que se encarga de los comportamientos de los objetos en la simulación.

La lógica del robot estará en una tarea llamada TROB_1 y la lógica de la subestación S1 se encontrará en la tarea S1. No se programará la lógica de otros robots y subestaciones ya que para ilustrar el funcionamiento general de la minifábrica y el pick&place de palés solo hace falta que los palés sean manipulados en una subestación. Ambas tareas se ejecutarán en paralelo en el controlador del robot de S1.

4.4 PLANIFICACIÓN Y ESTIMACIÓN ECONÓMICA

4.4.1 PLANIFICACIÓN TEMPORAL

Por fases.

Fase 1: Diseño de mecanismos. Hasta finales de abril.

Fase 2: Diseño de trayectorias y movimiento de palés. Hasta mediados de mayo.

Fase 3: Programación de la lógica de la estación. Hasta mediados de junio.

Fase 4: Creación de la demostración. Hasta finales de junio.

4.4.2 ESTIMACIÓN ECONÓMICA

La ventaja de los Gemelos Digitales son su ahorro económico. En el caso de este Proyecto de Fin de Grado se ha utilizado una licencia académica de RobotStudio proporcionada por la Universidad Pontificia de Comillas, por lo que no se ha incurrido en gastos adicionales.

4.5 ALINEACIÓN CON LOS OBJETIVOS DE DESARROLLO SOSTENIBLE

Este proyecto, como todos los que se encargan de la evolución de la industria, está alineado con varios Objetivos de Desarrollo Sostenible (ODS), pues una mayor eficiencia en los procesos industriales trae consigo numerosas repercusiones positivas

4.5.1 SALUD Y BIENESTAR

Desde que en 1983 se inventó el primer robot cirujano *Arthrobot*, la cirugía robótica ha ido avanzando cada vez más rápido con el paso de los años. Las operaciones complicadas en las que hace falta una precisión y rapidez considerables son procesos que un robot bien programado y calibrado puede realizar con más éxito que cualquier humano. El problema actual es que programar y calibrar estos robots es costoso, ya que deben operar sobre el cuerpo humano. Solo se puede probar estos robots en cadáveres donados a la medicina, pero gracias a las simulaciones en entornos virtuales se pueden hacer las pruebas y retoques que se desee con las trayectorias del robot antes de su uso en cuerpos humanos reales

4.5.2 INDUSTRIA, INNOVACIÓN E INFRAESTRUCTURA

Este modo de programación de robots es ideal para el crecimiento del sector industria, ya que permite tener lista la programación de los robots antes de que lleguen a las fábricas además de facilitar su reprogramación. Este avance es necesario para tener plantas que puedan mejorar con el tiempo sin tener que detener el proceso de producción por ello. En vez de parar la fábrica entera para recalibrar y reprogramar un robot, basta con actualizar su código durante un proceso mucho más breve.

4.5.3 TRABAJO DECENTE Y CRECIMIENTO ECONÓMICO

Este objetivo se encuentra directamente ligado al objetivo anterior, ya que una mejora de la industria tiene un impacto importante sobre el crecimiento económico. Además, los robots pueden desempeñar ciertas tareas para liberar a las personas de ellas o incluso cooperar con los trabajadores para facilitarles su función, como en el caso de los cobots. Esto permite que las condiciones de trabajo en la industria y otros sectores mejoren.

Capítulo 5. MODELO DESARROLLADO

5.1 ANÁLISIS DE LA PLANTA

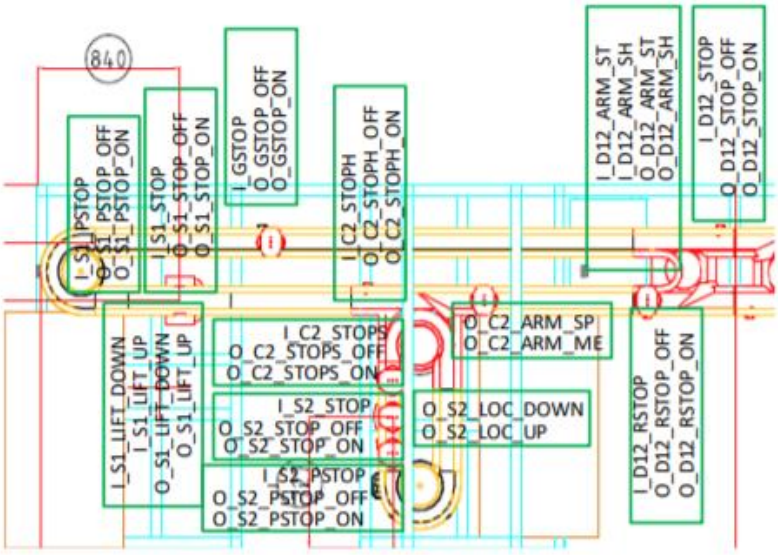

Figura 13 - Señales en S1 y S2

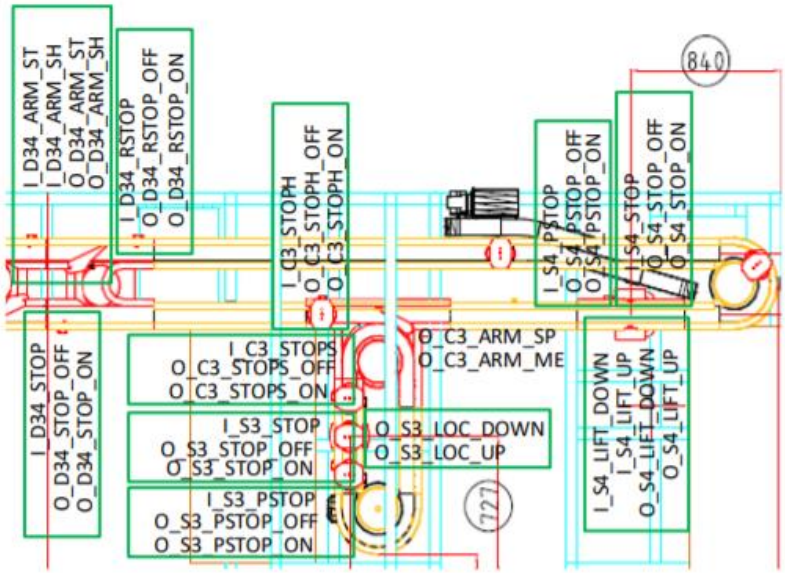

Figura 14 - Señales en S3 y S4

En Figura 13 y Figura 14 se puede ver la distribución de las señales de toda la estación. Las señales que empiezan por I son las salidas de los sensores de la planta y están conectados con la entrada (Input) de la controladora de la planta. Las señales que empiezan por O son las salidas (Output) de la controladora, y controlan las válvulas de los mecanismos. Las señales O van en pares ya que las válvulas son de doble acción y cada una de ellas necesita dos señales.

Hay un sensor en cada retenedor, ya sea previo o posterior, así como en los mecanismos localizadores elevadores para saber en qué posición se encuentran.

Como se explicó en 4.2.2, la estación cuenta con 2 Divert, 2 Combine, 9 retenedores previos, 4 retenedores posteriores, 2 localizadores de túnel, 2 localizadores simples y 4 robots con sus 4 pinzas. Suman un total de 42 señales para la planta y 4 para las pinzas, además de las señales que controlan los brazos robóticos. Todas estas señales proceden de las cuatro controladoras de los IRB120, de los dos PLC de los subconjuntos S1+S2 y S3+S4 y de tres remotas de entradas y salidas digitales Siemens ET200SP que se conectan con los PLC. En el Gemelo Digital todas las señales procederán de la controladora del robot de S1 para simplificar el modelo, pero el funcionamiento será exactamente el mismo.

5.2 TRAYECTORIAS

Para que los palés puedan seguir la trayectoria deseada se deben definir de antemano las trayectorias que han de seguir, ya que no se realiza una simulación física, sino que los palés siguen un programa.

Los palés tienen 3 tipos de movimiento en la estación: avance, curva y elevación/descenso. La lógica de estos movimientos se explicará en más detalle en 5.4.1. Para avance y elevación/descenso no hará falta especificar las rutas de antemano, ya que son trayectorias rectas para las que RobotStudio tiene un SmartComponent predeterminado. Las curvas se pueden ver en Figura 15 y Figura 16, y sirven para que los palés continúen dentro de la estación al pasar por S1 (Curva 1), para tomar el circuito S1+S2 en lugar de la estación

completa (Curva 2), para entrar en S2 (Curva 3), para salir de S2 (Curva 4) y para continuar en S2 (Curvas 5 y 6). Debido a la simetría de la estación, en el subconjunto S3+S4 hay el mismo número de curvas con las mismas funciones. Son un total de 12 curvas. Las curvas 1, 2, 5, 6 y sus simétricas se han realizado como una semicircunferencia, mientras que 3 y 4 y sus simétricas son arcos de elipses. Estas elipses no siguen matemáticamente el mismo camino que seguiría el palé en la planta física, pero son una aproximación suficientemente exacta de la trayectoria del palé sin necesidad de entrar en curvas complejas.

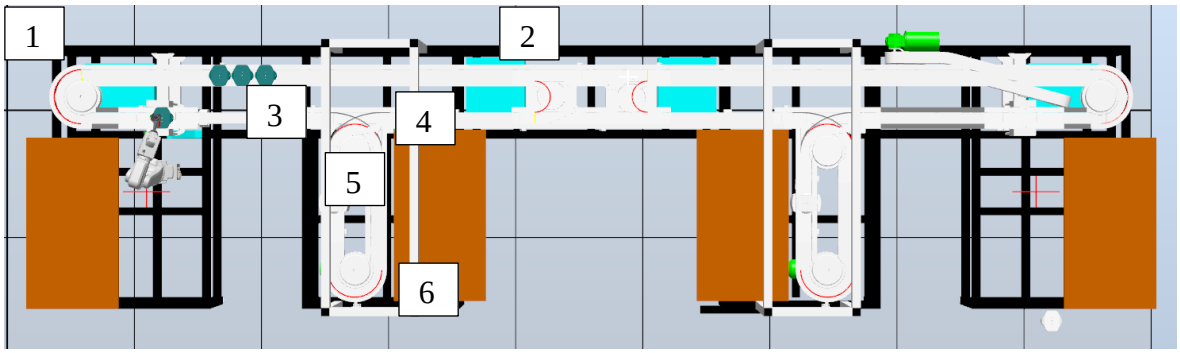

Figura 15 - Vista en planta de la estación

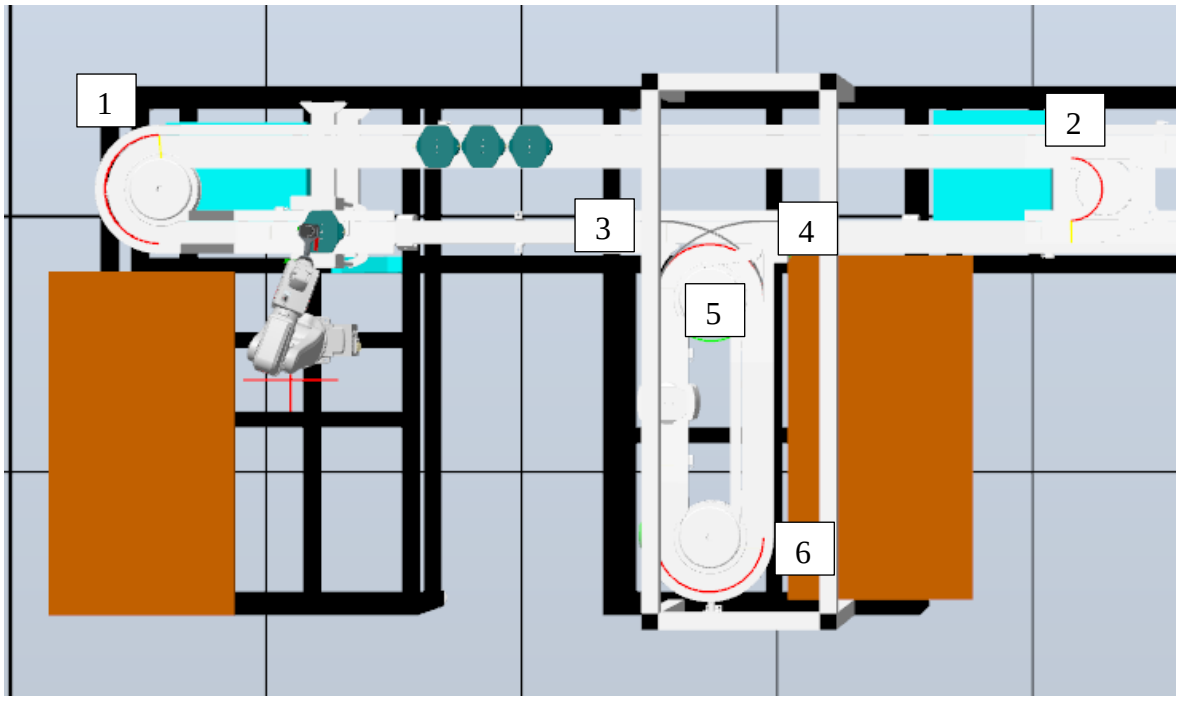

Figura 16 - Detalle de curvas de S12

Cada curva tiene un sensor justo al inicio de la trayectoria. Estos sensores no se corresponden con sensores reales de la minifábrica, sino que su función es detectar cuándo un palé está a punto de tomar la curva para mandar la señal correspondiente al SmartComponent General (Figura 17) y que este se encargue de que el palé correcto siga la trayectoria deseada.

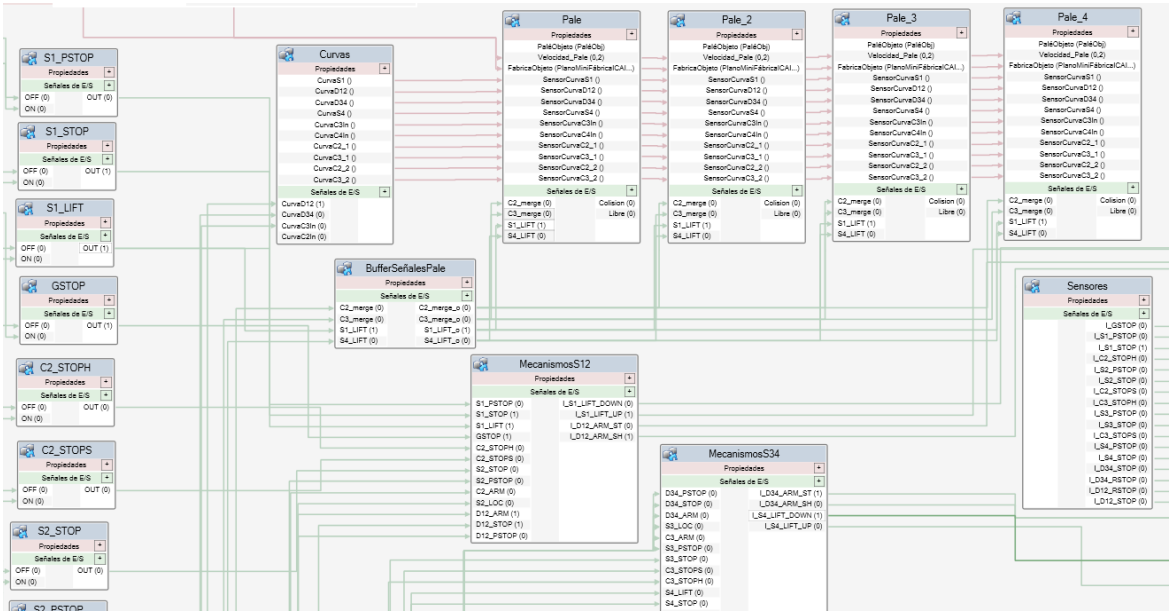

Figura 17 - SmartComponent General. Vista interior.

5.3 MECANISMOS

Para conseguir mecanismos funcionales a partir del modelo 3D proporcionado, se utilizaron las herramientas comentadas en 4.3.1 para separar todas las piezas de los mecanismos entre sí para luego ensamblarlas en su posiciones correspondientes de nuevo. En total hay 6 mecanismos y estas son sus especificaciones:

5.3.1 RETENEDOR PREVIO

Especificaciones: Translación horizontal 12mm.

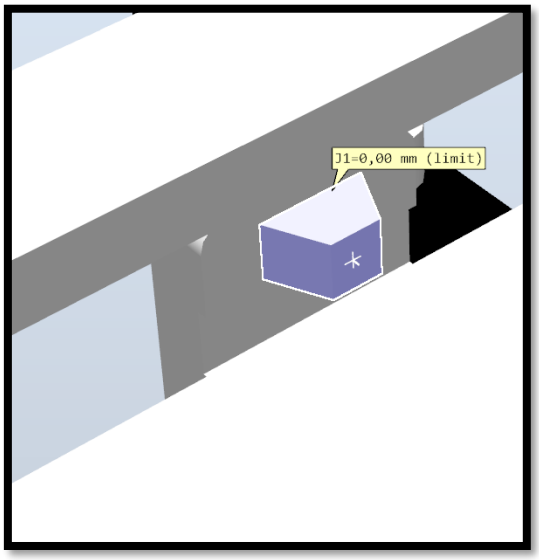

Figura 18 - Retenedor previo (bloqueando)

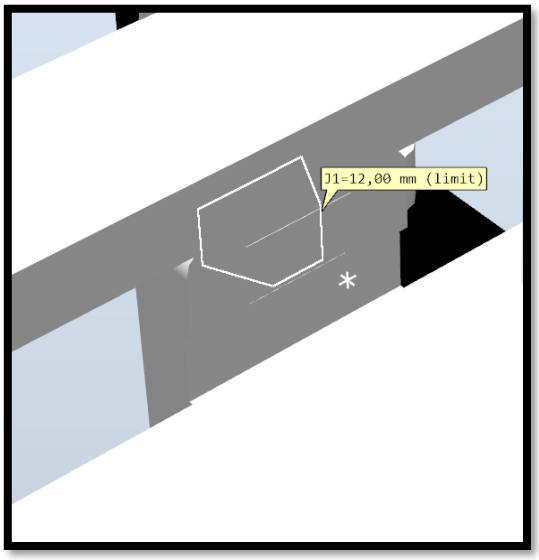

Figura 19 - Retenedor previo (libre)

5.3.2 RETENEDOR POSTERIOR

Especificaciones: Rotación 90º.

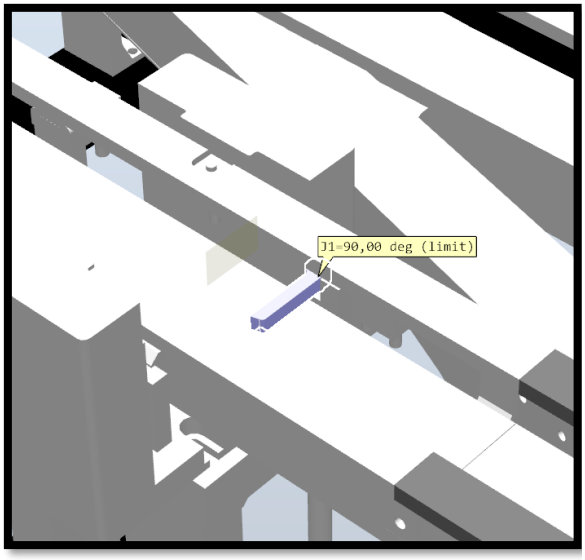

Figura 20 - Retenedor posterior (bloqueando)

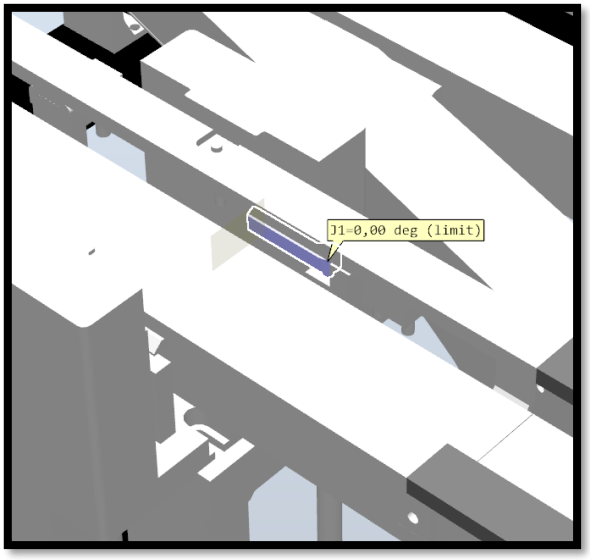

Figura 21 - Retenedor posterior (libre)

5.3.3 COMBINE

Especificaciones: Rotación 50º

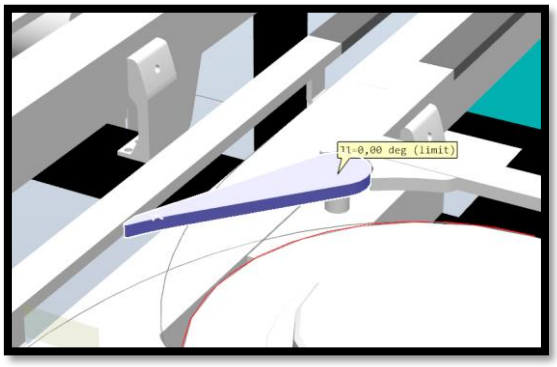

Figura 22 - Combine (aceptando)

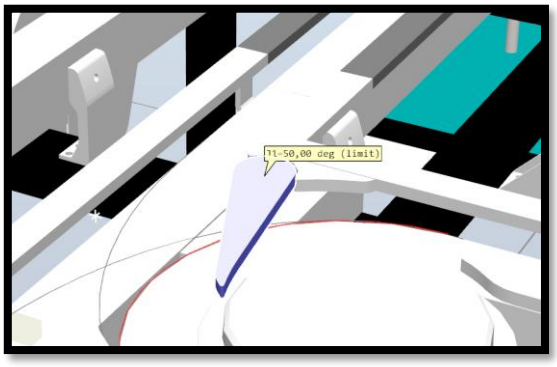

Figura 23 – Combine (rechazando)

5.3.4 DIVERT

Especificaciones: Rotación 25º.

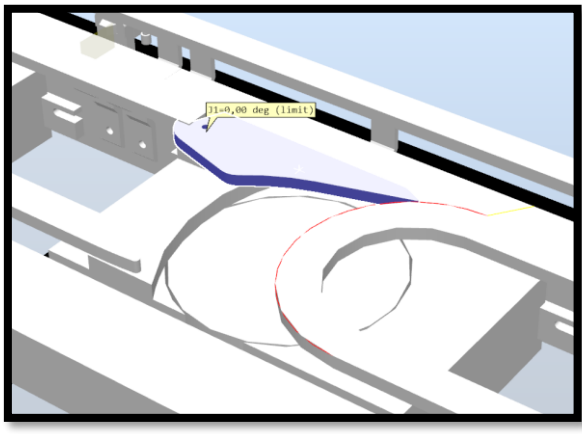

Figura 24 – Divert (recto)

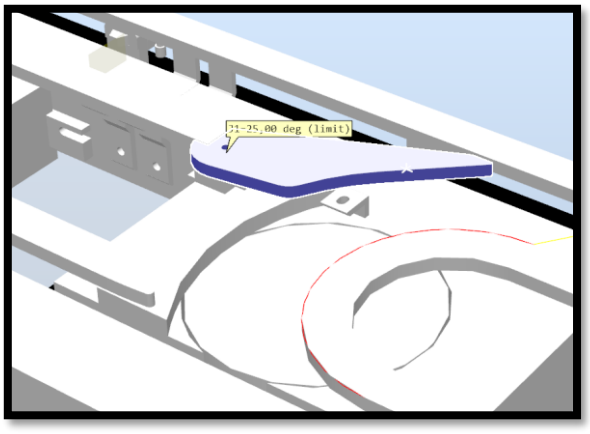

Figura 25 - Divert (cambio)

5.3.5 LOCALIZADOR TÚNEL (XBUL 11 T)

Especificaciones: Translación vertical 245mm.

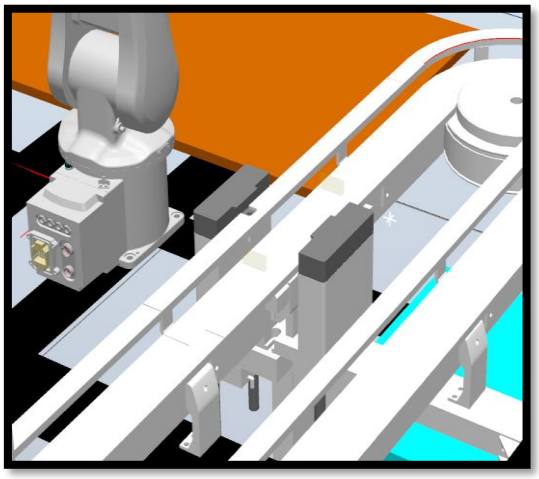

Figura 26 – XBUL 11 T (abajo)

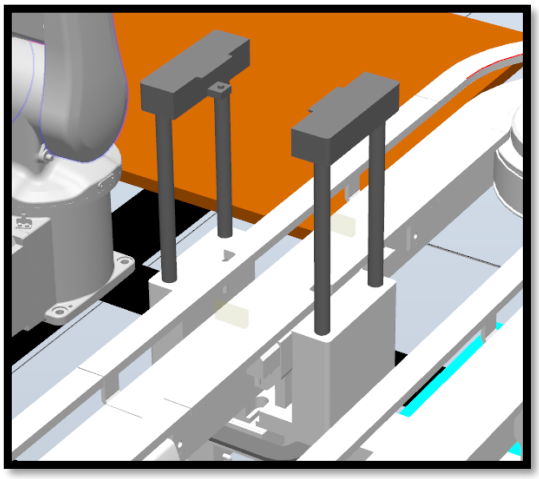

Figura 27 – XBUL 11 T (arriba)

5.3.6 PINZA

Especificaciones: Translación horizontal de 20mm por cada extremo de la pinza.

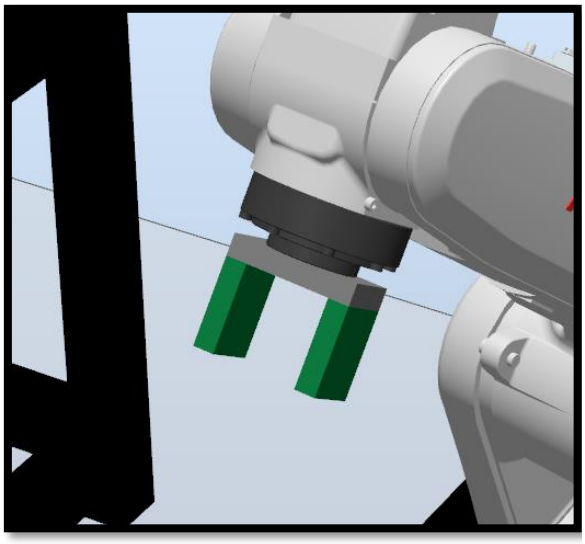

Figura 28 – Pinza (abierta)

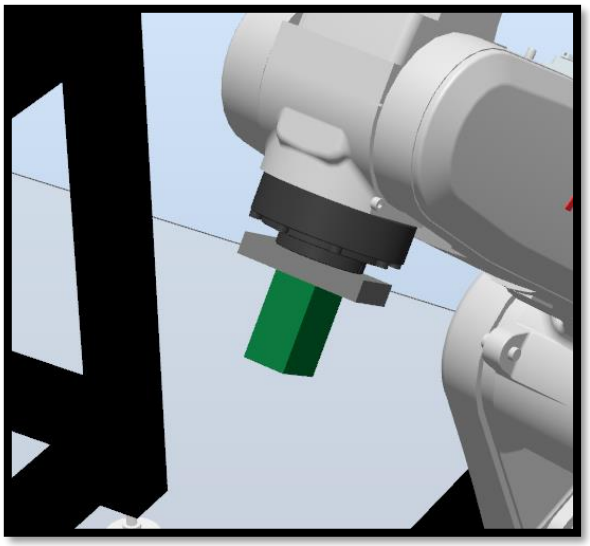

Figura 29 – Pinza (cerrada)

No se considera el localizador simple un mecanismo ya que no tiene un movimiento perceptible en la simulación, puesto que simplemente bloquea el movimiento de los palés. Se trata como a un objeto estático.

5.4 LÓGICA

Como se ha mencionado anteriormente, la lógica de la estación se ha programado mediante módulos SmartComponents. A continuación, se explica la función de cada uno de ellos.

5.4.1 PALÉS

Este es el SmartComponent (Figura 30 y Figura 31) más complejo, ya que contiene todo el movimiento de cada palé individualmente. Controla las colisiones entre palés y con el entorno, el avance rectilíneo, subidas y bajadas y las trayectorias curvas.

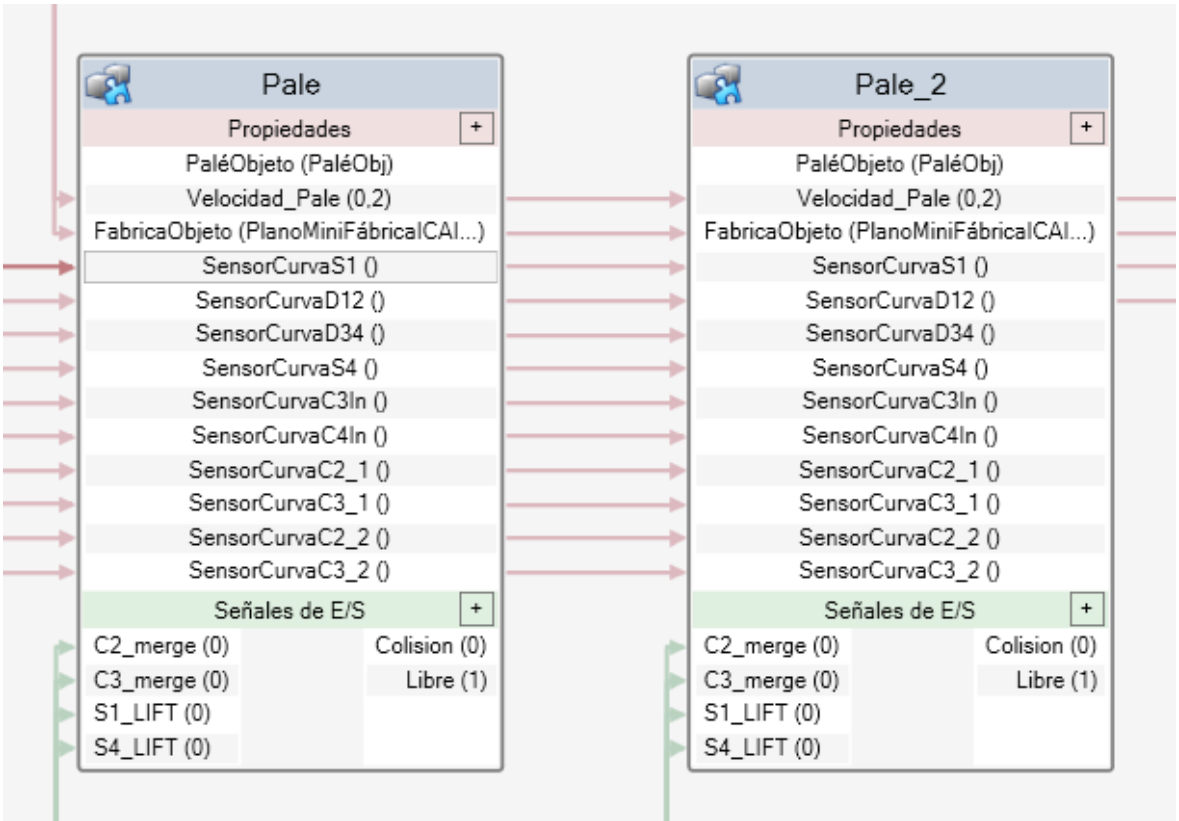

Figura 30 - Dos SmartComponent Palé

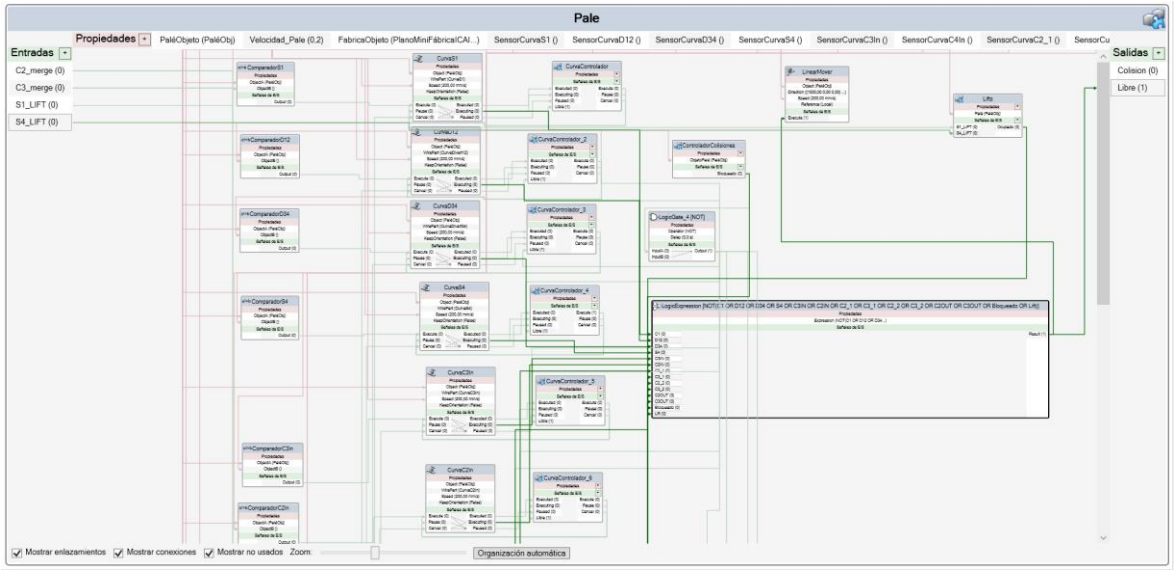

Figura 31 - SmartComponent Palé (vista interior)

Tiene 4 entradas y 2 salidas. De las cuatro entradas, C2_merge y C3_merge le aportan la información necesaria para determinar qué camino tomar cuando están en S2 o S3 y pasan cerca del combine. Tiene la opción de seguir en la subestación o de salir de ella y converger con el flujo principal de palés, que es lo que hacen cuando estas señales están a nivel alto. Las otras dos señales, S1_LIFT y S4_LIFT dan la información de las órdenes que reciben los mecanismos XBUL 11 T. En caso de encontrarse encima de uno de ellos, seguirán su movimiento. Las dos salidas del SmartComponent tienen tan solo la función de ayudar con el *debug* del programa. Además, el SmartComponent cuenta con varias propiedades que también cuentan como entradas a efectos prácticos y que se encargan de recibir la información de los sensores de las curvas (Figura 32 y Figura 33).

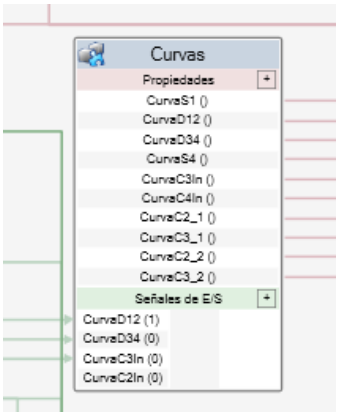

Figura 32 - SmartComponent Curvas. Vista exterior.

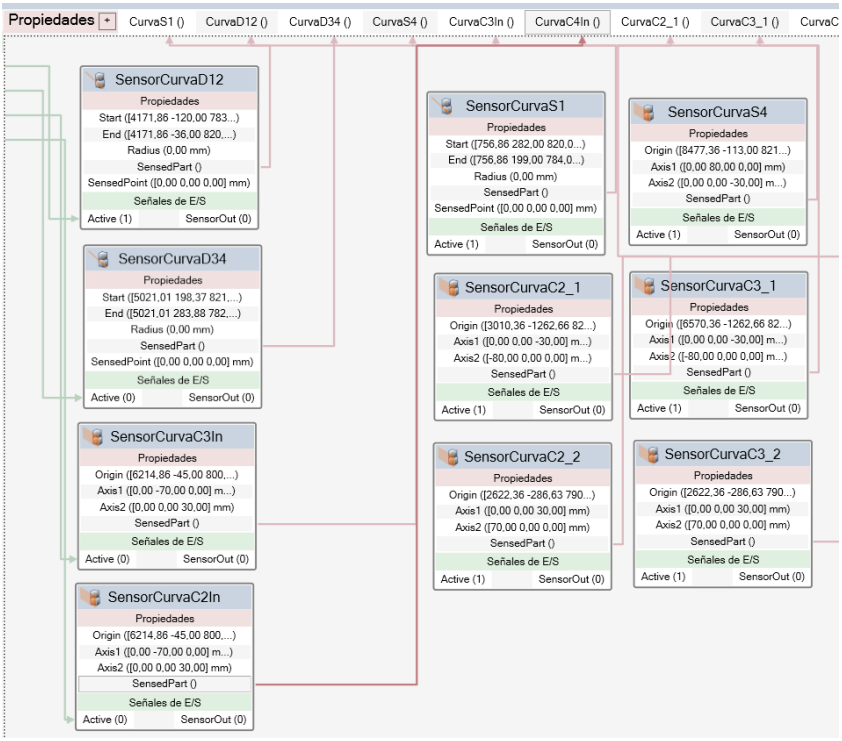

Figura 33 - SmartComponent Curvas. Vista interior.

El SmartComponent Palé está compuesto por cuatro unidades básicas: control de curvas, movimiento de avance (Figura 34), movimiento de elevadores (Figura 35) y control de colisiones (Figura 36).

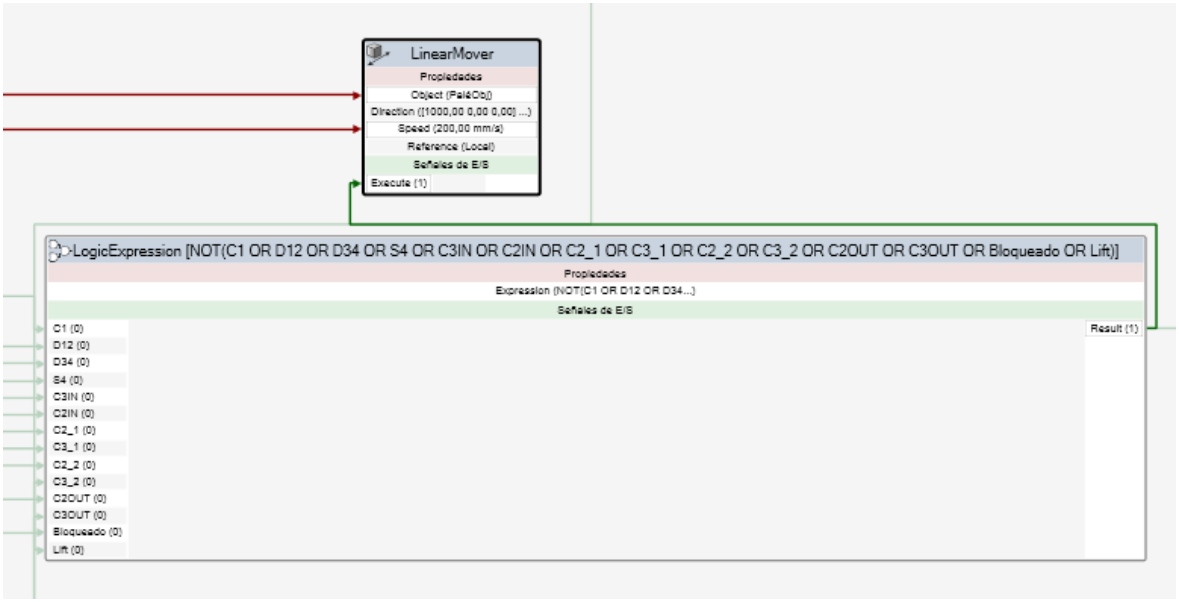

Figura 34 - Lógica avance (SmartComponent Palé)

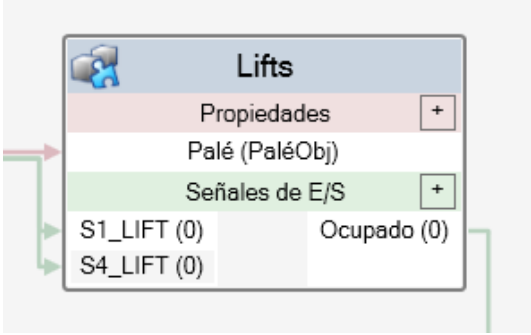

Figura 35 - SmartComponent Lifts (vista exterior)

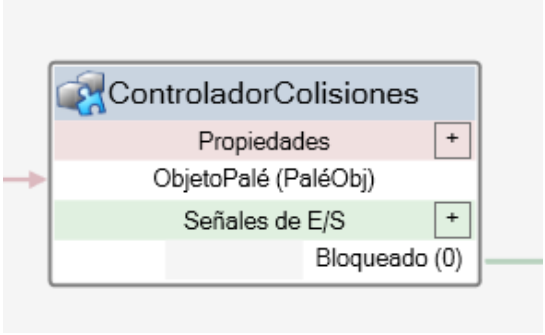

Figura 36 - SmartComponent Colisiones

5.4.1.1 Control de curvas

Las lógica de las curvas simples (Figura 37) está formada por un comparador, un componente de seguimiento de curvas y un controlador. El comparador recibe información constante sobre el sensor colocado a la entrada de la curva. En caso de que el objeto recibido coincida con el palé en el que se encuentra el comparador, ejecuta la acción de seguimiento de curva. El controlador de curvas (Figura 38) recibe 4 señales. Las tres primeras son información sobre el estado del seguimiento de la curva (terminada, en proceso, pausada o no empezada) mientras que la última señal procede del control de colisiones del palé y contiene la información de si el palé puede avanzar o no. Sus tres salidas están conectadas

al seguidor de curva y permiten pausar y reanudar la marcha de un palé en medio de una curva en caso de colisión.

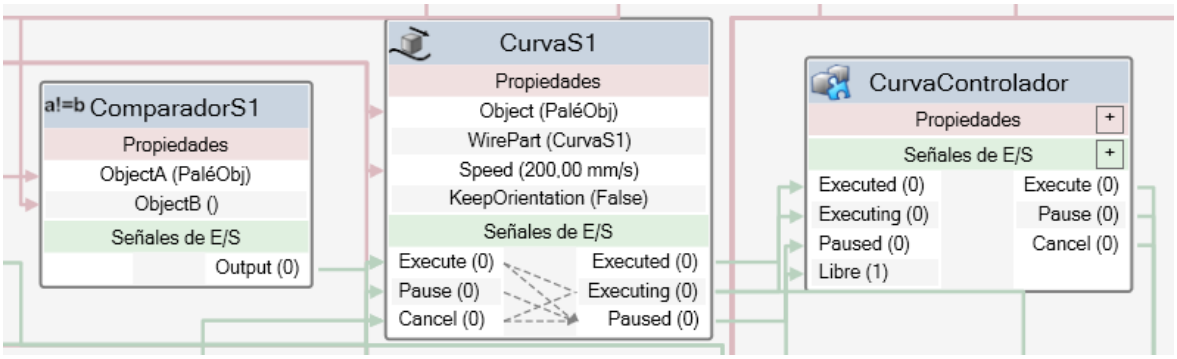

Figura 37 - Lógica de curvas simples

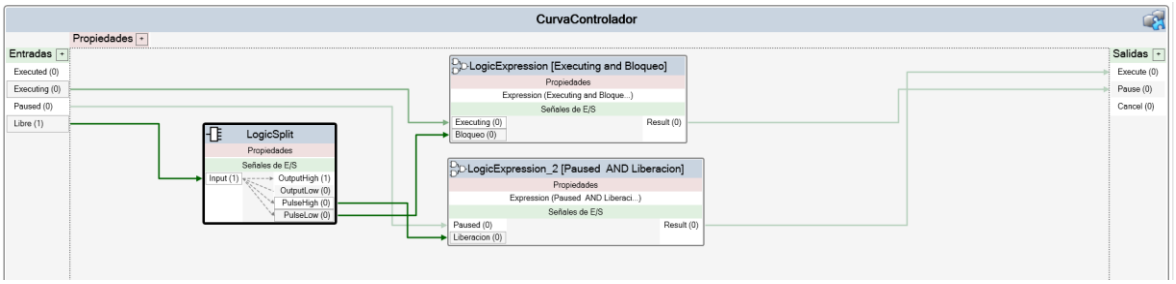

Figura 38 - Controlador de curvas (vista interior)

Las curvas complejas (Figura 39) se tratan de un modo similar a las simples, solo que al detectar un palé se ejecuta una curva u otra dependiendo de la posición de los mecanismos de guía Combine y Divert.

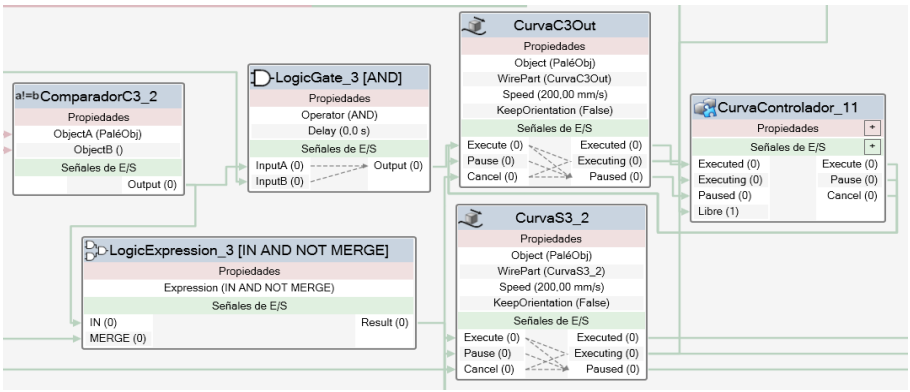

Figura 39 - Lógica de curvas complejas

5.4.1.2 Control de movimiento de avance

La lógica de movimiento de avance es sencilla (Figura 40). Consta de una expresión lógica que indica si el palé puede continuar la marcha o no. Activará la señal "bloqueado" siempre que detecte una colisión, se encuentre bloqueado en un mecanismo o esté ejecutando una curva, ya que no puede continuar avanzando linealmente durante la trayectoria curva.

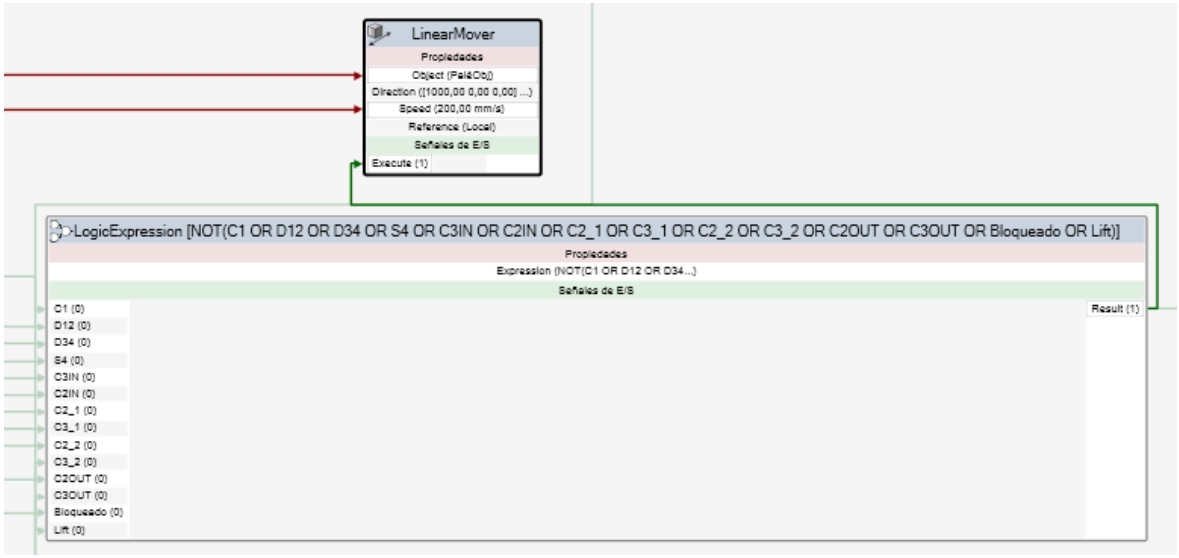

Figura 40 - Lógica del movimiento de avance

5.4.1.3 Control de movimiento vertical

La lógica detrás del movimiento vertical asociado a los elevadores es común para el SmartComponent "Lifts" de S1 y el de S4, aunque en Figura 41 solo se muestra la parte de S1 para ahorrar espacio y hacer la imagen más clara. La acción de subir el palé se ejecuta cuando se detecta que está en posición gracias a la colisión con el retenedor de S1 y a la vez hay un flanco de subida en la señal de S1_LIFT. La información sobre la posición del lift se guarda como una variable binaria en el SRLatch. Al subir se aplica el Set y se actualiza la posición a "arriba". La lógica para bajar es la misma. Para S4 la lógica se repite.

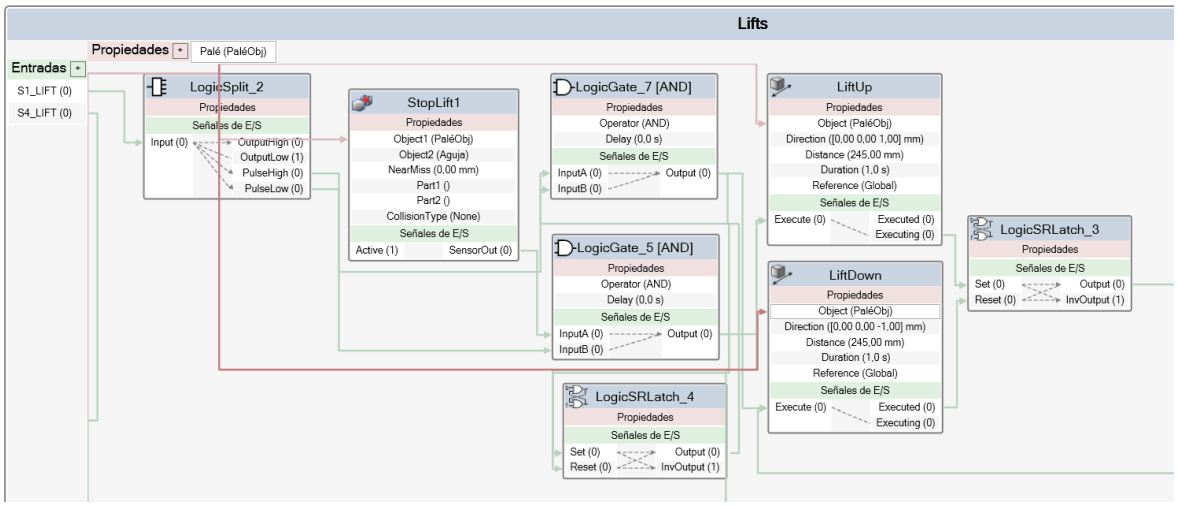

Figura 41 - Lógica movimiento vertical

5.4.1.4 Control de colisiones

El control de colisiones, pese a que no parece excesivamente complejo, ha sido el SmartComponent en el que más tiempo se ha tenido que invertir. Simular el comportamiento físico de los palés ha sido complicado sin realizar una simulación física en tiempo real, pero finalmente se ha conseguido un módulo simple y comprensible. El problema de esta lógica era que el palé tenía que detenerse al chocar con un mecanismo de bloqueo u otro palé que bloquease el paso, pero debía ignorar los posibles roces con los raíles de la minifábrica y otros mecanismos de guía. Además, al chocar con otro palé, con una lógica convencional estos dos quedarían atrapados indefinidamente. El palé de atrás chocaría con el de adelante y no podría avanzar hasta que dejase de detectarlo. Por su parte, el palé de delante estaría en colisión con el palé de atrás y no podría avanzar.

Para solucionar estos problemas, se han configurado varios mecanismos para no ser detectados por los palés. Además, se ha hecho una lógica de palés discriminatoria, que es capaz de detectar y distinguir las colisiones con piezas de la minifábrica, con otros palés o con las piezas de Lego que se monten encima del palé durante la demostración del funcionamiento.

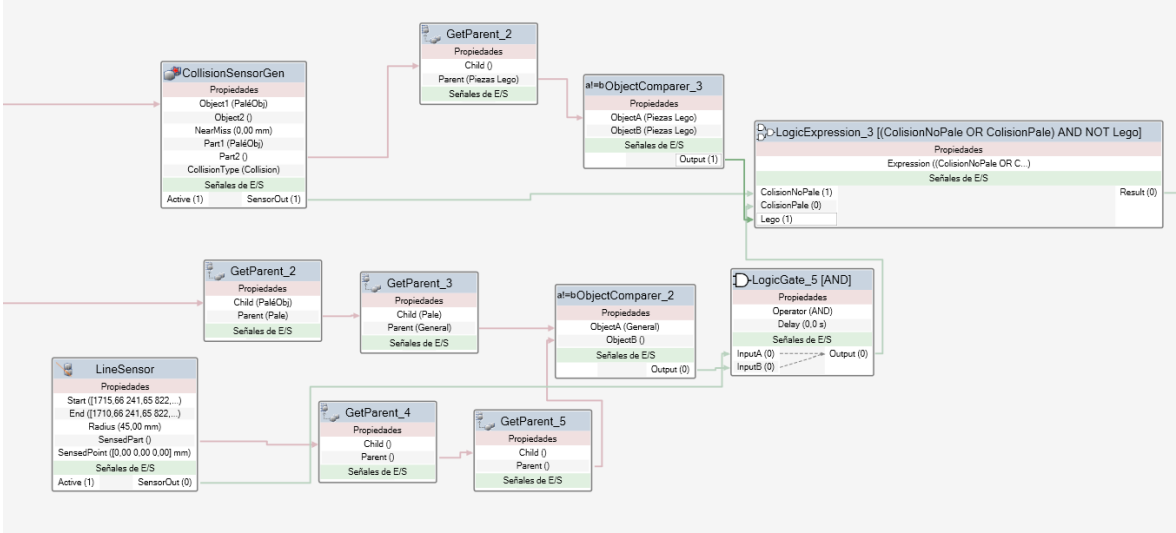

Figura 42 - SmartComponent Colisiones (vista interior)

El detector de colisiones activa la señal de colisión siempre que se detecta un objeto en contacto con el palé, mientras que el comparador al que está conectado detecta si la colisión ha sido con una pieza de Lego para activar la señal "Lego". Por otro lado, los palés cuentan con un detector en su parte delantera (Figura 43) para detectar otros palés. Esto se ha realizado de esta manera para que los palés solo se detengan cuando detectan un palé frenado delante, pero que puedan ignorar un palé que tengan detrás, ya que en ese caso no se impediría el avance del que está delante. Además, este sensor hace que haya unos milímetros de separación entre palé y palé. En teoría deberían estar en contacto para simular la situación de la minifábrica física, pero de esta manera no se podrían ignorar los palés de detrás y, además, entraría en conflicto con el sensor de colisiones general, causando un bloqueo permanente de todos los palés que entren en contacto entre sí.

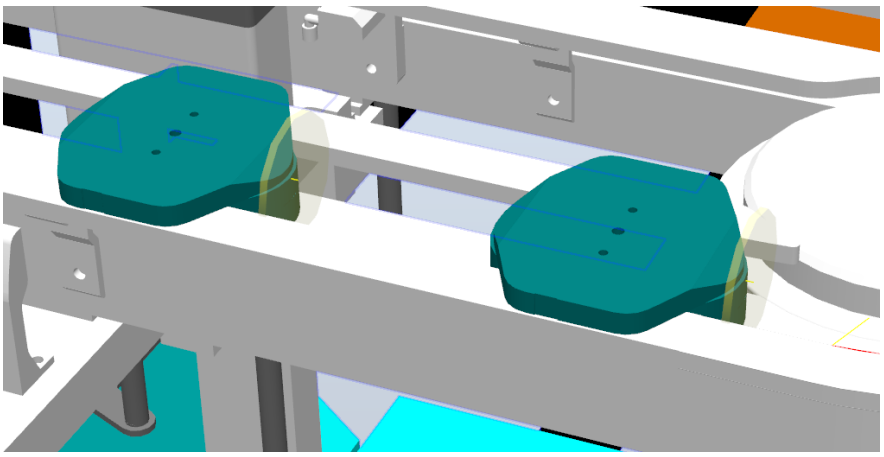

Figura 43 - Dos palés con sus detectores de palés

Estos detectores de palé son visibles en la simulación para facilitar la fase de desarrollo de la lógica, pero se puede configurar su visibilidad desde los ajustes del sensor.

5.4.2 MECANISMOS

Los mecanismos están agrupados en dos SmartComponent superiores llamados MecanismosS12 y MecanismosS34, que se encargan de recibir las señales de los mecanismos de las áreas de S12 y S34. Estas señales no vienen directamente del controlador, sino que pasan por un SmartComponent intermedio que se explicará en 5.4.5. Al llegar estas señales, pasan por un LogicSplit que recibe una señal y la divide en cuatro, según esté a nivel alto, bajo, flanco de subida o flanco de bajada. Los flancos se utilizan para activar la orden de movimiento del mecanismo.

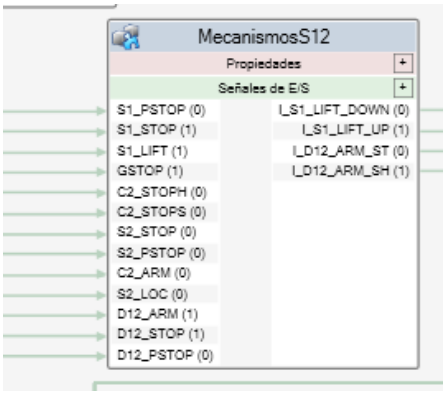

Figura 44 - SmartComponent MecanismosS12. Vista exterior.

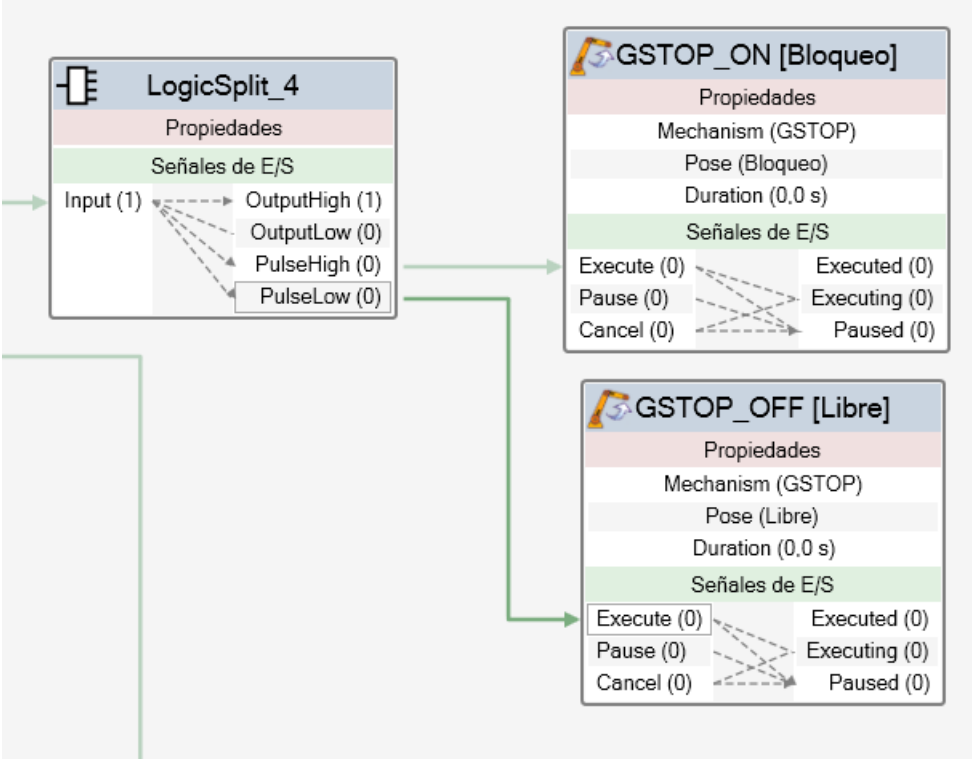

Figura 45 - Detalle de SmartComponent MecanismosS12. Mecanismo sin sensor.

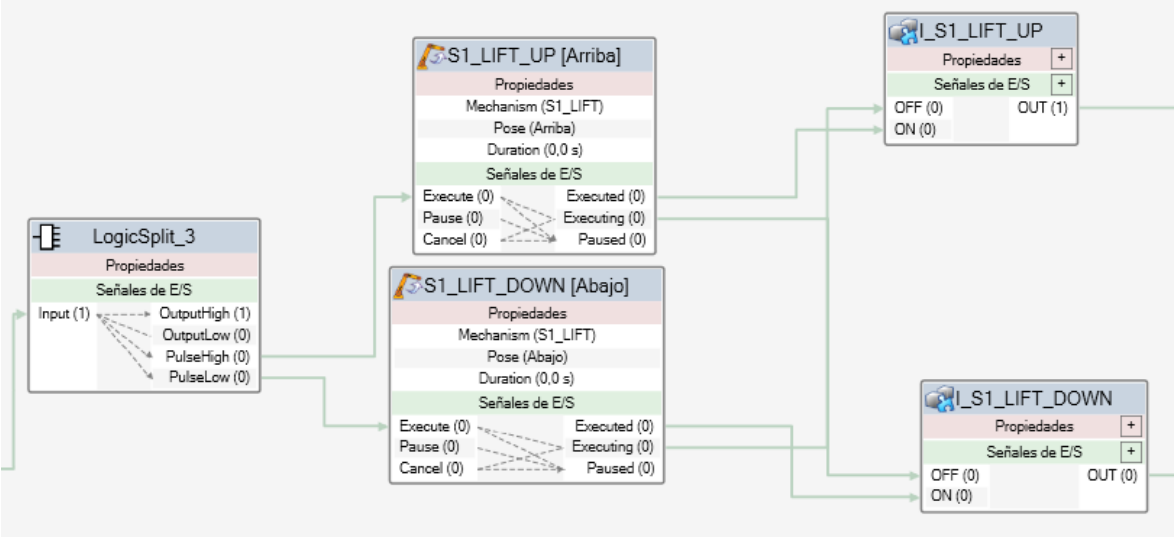

Figura 46 - Detalle de SmartComponent MecanismosS12. Mecanismo con sensor.

5.4.3 SENSORES

Al igual que los mecanismos, todos los sensores de la planta están modelados según SmartComponents anidados en un SmartComponent general llamado Sensores. Este SmartComponent no tiene entradas, y sus salidas se corresponden con las señales de los sensores, que se conectan directamente al controlador del IRB120.

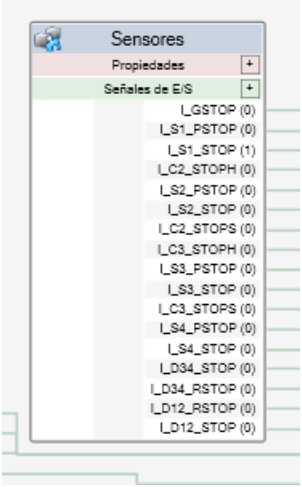

Figura 47 - SmartComponent Sensores. Vista exterior.

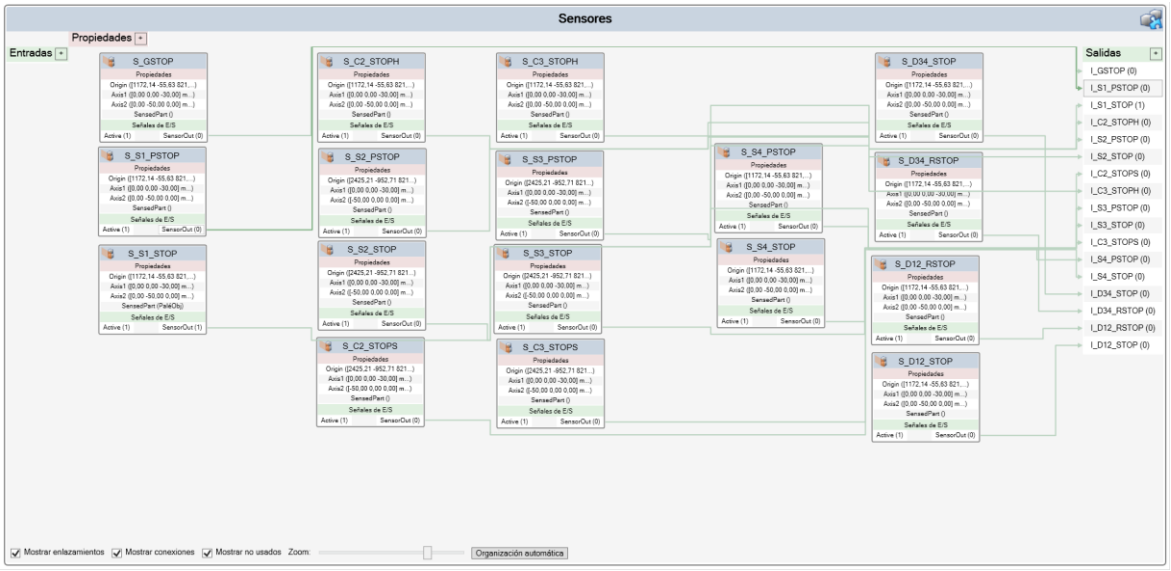

Figura 48 - SmartComponent Sensores. Vista interior.

5.4.4 CONTROLADOR

El controlador se conecta con el SmartComponent General en la pestaña Lógica de la Estación (Figura 49). Las salidas de uno son las entradas del otro. En el controlador se ejecuta el código RAPID y las señales de salida llegan al SmartComponent General, que es el que se encarga de que la simulación se ejecute de forma correcta.

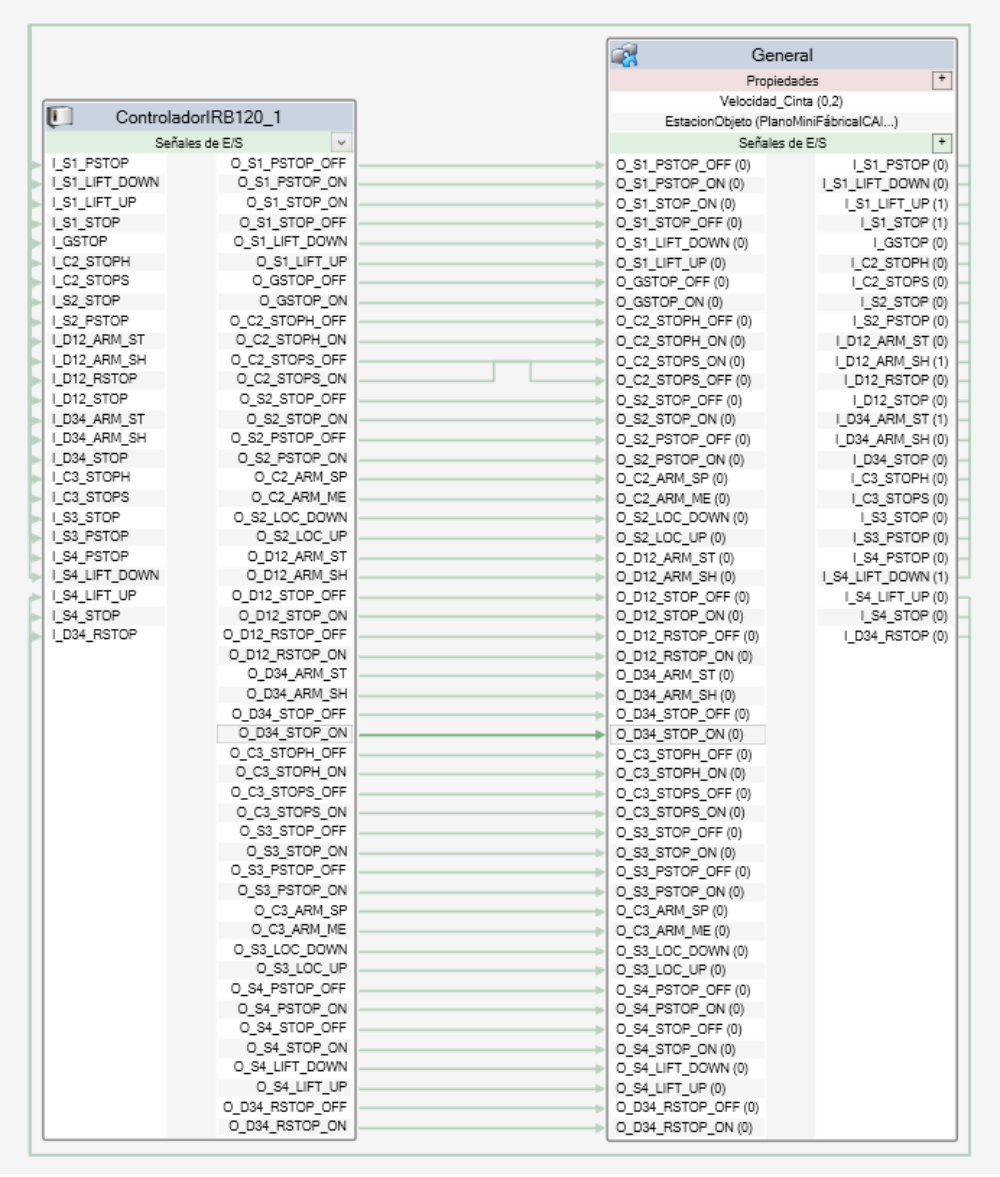

Figura 49 - Lógica de la estación

5.4.5 SEÑALES INTERMEDIAS

Dentro del SmartComponent General hay otros SmartComponents intermedios que no son realmente necesarios pero que facilitan la programación aportando limpieza en los diagramas de flechas.

Entre estos componentes están los acondicionadores, que toman las dos señales tipo O_xx que controlan cada mecanismos y las convierten en una sola señal, reduciendo a la mitad el número de flechas que pasan por la zona central del SmartComponent. Su lógica se ve en Figura 50 y Figura 51 y lo que simula es la respuesta de las válvulas de los mecanismos ante las señales de entrada. Ante un 00 o un 11 no cambia su estado, pero ante un 01 se activa la señal de salida y ante un 10 se desactiva.

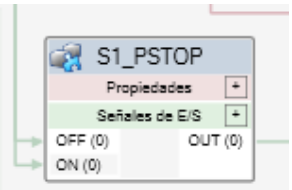

Figura 50 - Acondicionador P1_PSTOP. Vista exterior.

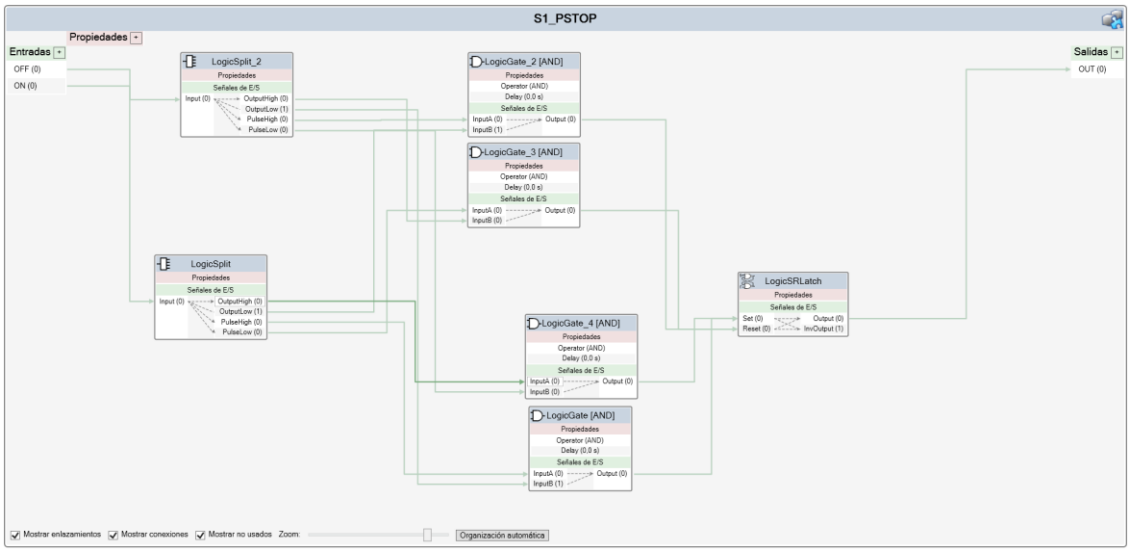

Figura 51 - Acondicionador P1_PSTOP. Vista interior.

También hay un buffer (Figura 52) que guarda las señales que han de ir a los palés. Todos los SmartComponent de tipo Palé son idénticos salvo por el objeto que se encargan de mover. Por ello, todos comparten las mismas señales. En vez de tener que enviar estas señales desde su origen hasta cada palé, abarrotando el SmartComponent General con flechas innecesarias, se ha creado un SmartComponent cuya única función es actuar de *buffer*, recibiendo estas señales y pasándolas inalteradas a su salida. De esta manera las señales llegan al *buffer* y se distribuyen a su salida, ayudando a la limpieza.

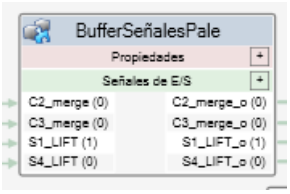

Figura 52 - Buffer. Vista exterior.

5.5 DEMOSTRACIÓN

Para la demostración de funcionamiento del Gemelo Digital (Figura 53) ha hecho falta programar un código RAPID para manejar la lógica de la estación y otro para controlar los movimientos de los robots. Estos dos códigos se ejecutan en el mismo controlador en las tareas S1 y T_ROB1 respectivamente. Además, se ha creado un SmartComponent independiente del SmartComponent General para la lógica del pick&place.

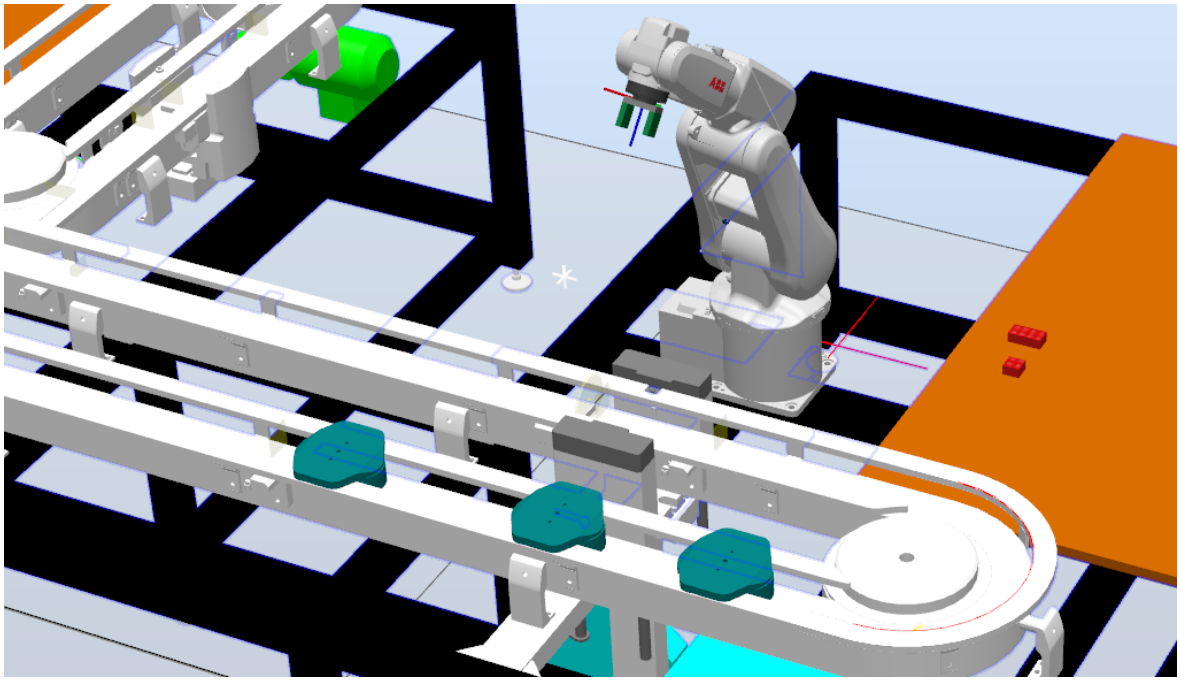

Figura 53 - Estado inicial de la demostración

La lógica de la tarea S1 consiste en lo siguiente:

El mecanismo Combine2 y el Divert12 están colocados de forma que los palés circulen exclusivamente por S1, sin entrar en S2 ni pasar al subconjunto S3+S4. Cuando el retenedor previo de S1 detecta un palé, este se desactiva durante 0.3 segundos para dejar pasar un único palé. Cuando el retenedor posterior detecta el palé, se sabe que el palé está en posición y el XBUL 11 T se eleva. Una vez arriba, se desactivan los retenedores previo y posterior para dejar pasar los palés que circulen por la estación mientras el robot trabaja en el palé del mecanismo elevador XBUL 11 T. Una vez terminada la tarea del robot, a la que se le dan 40 segundos en este caso, se vuelve a activar el retenedor previo para que no circulen más palés por debajo del mecanismo elevador. Se deja pasar un segundo para asegurar que no queda ningún palé circulando por debajo y se baja el mecanismo elevador, liberando el palé sobre el que se ha trabajado. Un segundo después, cuando el palé ya ha abandonado el mecanismo, se vuelve a activar el retenedor posterior y el programa vuelve a la primera línea, listo para ejecutarse de nuevo con otro palé. El código puede encontrarse en ANEXO I.

El robot está programado para esperar hasta que el palé esté en posición, es decir, hasta que se active la señal I_S1_LIFT_UP. Después, según se haya configurado el programa, se acerca hacia la pieza de Lego escogida, la coge y la coloca encima del palé en la posición indicada. El código puede encontrarse en ANEXO I.

Lo más interesante de este código es la función colocaLego(num x, num y, num z, num rot, bool largo, bool centro).

Se le introducen coordenadas (x, y, z), siendo el origen (0, 0, 0) el punto donde una pieza de Lego de 2x2 se colocaría en el centro del palé y a la primera altura disponible del mismo. Las coordenadas (-2,0,0) corresponderían a la posición en la que una pieza de lego estaría adyacente a la anterior pieza mencionada. Ya que son piezas de 2x2, hace falta moverse dos puntos en las coordenadas para que no haya colisión. De esta manera, si quisiéramos poner una tercera pieza de 2x2 encima de estas dos y en el centro de estás, se utilizaría la posición (-1, 0, 1).

El argumento *rot* es un número entero del 1 al 4 y define la orientación de las pinzas al colocar la pieza. Aunque en las piezas de 2x2 el resultado es el mismo sea cual sea la orientación, es posible que una de las orientaciones no permita que entren las pinzas y la otra sí. En las piezas de 2x4 la existencia de este argumento es claramente necesaria.

El argumento *largo* determina si la pieza que se coloca es de 2x2 o de 2x4. Como su nombre indica, con un TRUE colocará una de 2x4 y con un FALSE una de 2x2.

El argumento *centro* solo es útil en el caso de piezas de 2x4. En el caso de TRUE la pinza cogerá la pieza desde el centro de la misma, mientras que para FALSE cogerá la pieza 16mm desplazada hacia un lado, como se ilustra en Figura 54.

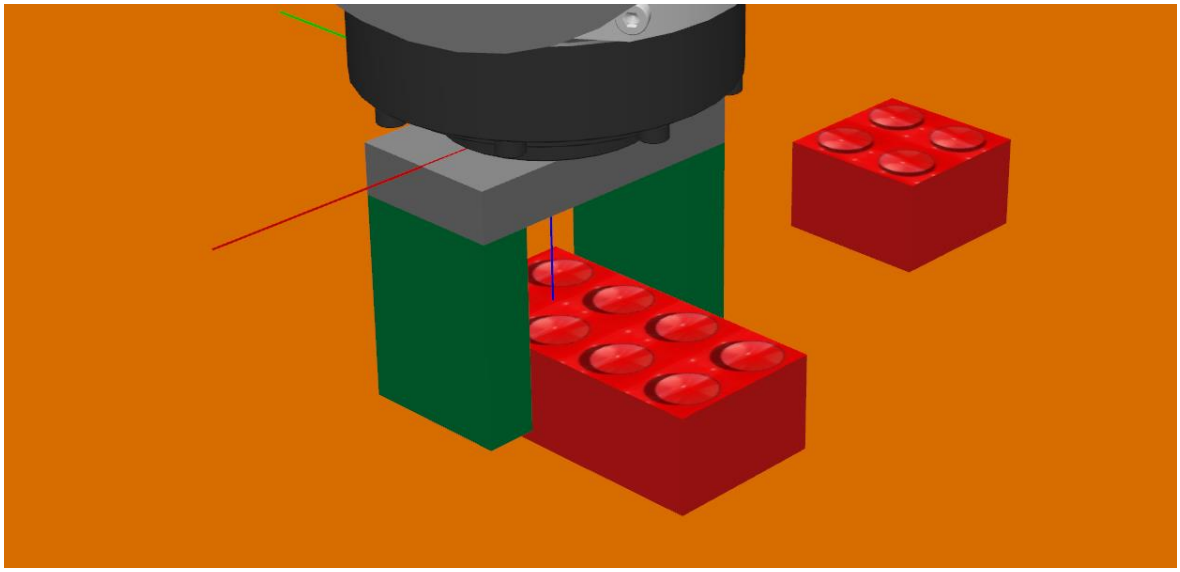

Figura 54 - Pinza cogiendo un Lego de 2x4 por un extremo.

Con esta lógica, programar la construcción de un pequeño robot humanoide sería tan sencillo como el siguiente código:

```
colocaLego 0, -2, 0, 1, FALSE, TRUE; !colocación de las dos piernas
colocaLego 0, 2, 0, 1, FALSE, TRUE;
colocaLego 0, -2, 1, 1, FALSE, TRUE;
colocaLego 0, 2, 1, 1, FALSE, TRUE;

colocaLego 0, 0, 2, 1, TRUE, TRUE; !unión de las piernas con pieza larga

colocaLego 0, 0, 3, 1, FALSE, TRUE; !torso
colocaLego 0, 0, 4, 1, FALSE, TRUE;

colocaLego 0, 1, 5, 3, TRUE, FALSE; !brazo 1
colocaLego 0, -1, 5, 1, TRUE, FALSE; !brazo 2

colocaLego 0, 0, 6, 1, FALSE, TRUE; !cabeza
```

El SmartComponent del pick&place (Figura 55) se llama “Piezas Lego”. No tiene entradas ni salidas, ya que no se comunica con otros SmartComponents ni con el controlador del robot.

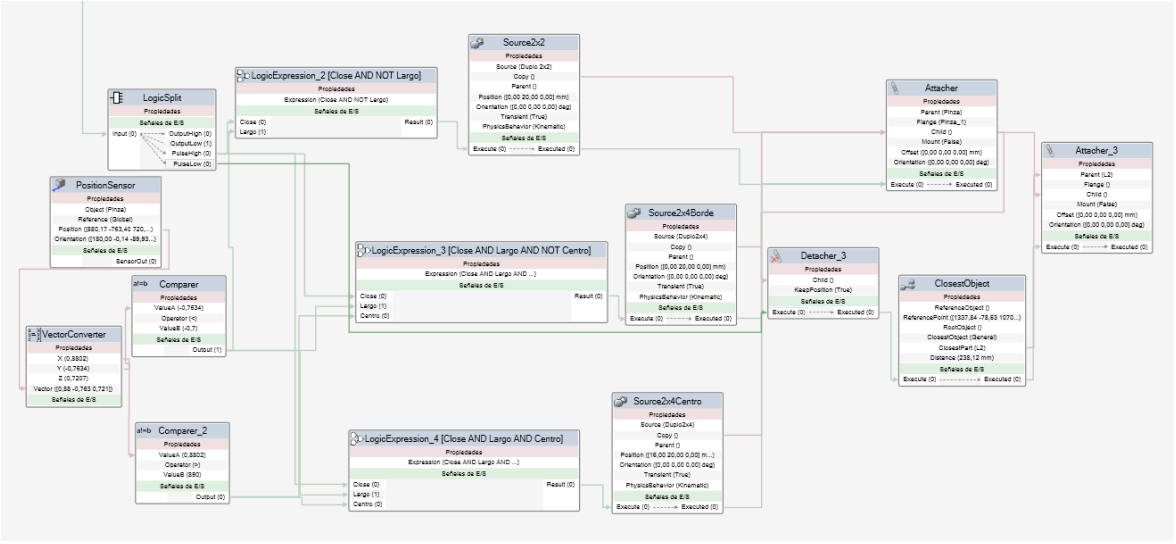

Figura 55 - SmartComponent Piezas Lego (vista interior)

El SmartComponent recibe la señal D10_2, que es la encargada de abrir y cerrar la pinza del robot. Cuando se detecta un flanco de subida para cerrar la pinza, el sensor de posición determina si la pinza está cogiendo una pieza de 2x2 o una de 2x4 por el centro o por su extremo según sus coordenadas.

Una vez se ha determinado el objeto a coger, se crea un clon del objeto y se ancla a la herramienta para que se mueva de manera solidaria con ella. Al detectar un flanco de bajada en la señal de la pinza, el objeto clonado se desancla de la pinza y se ancla al palé que esté colocado en el XBUL 11 T de S1.

Debido a la imposibilidad de adjuntar un vídeo en un documento escrito, a continuación se muestran los fotogramas más relevantes del vídeo de la demostración:

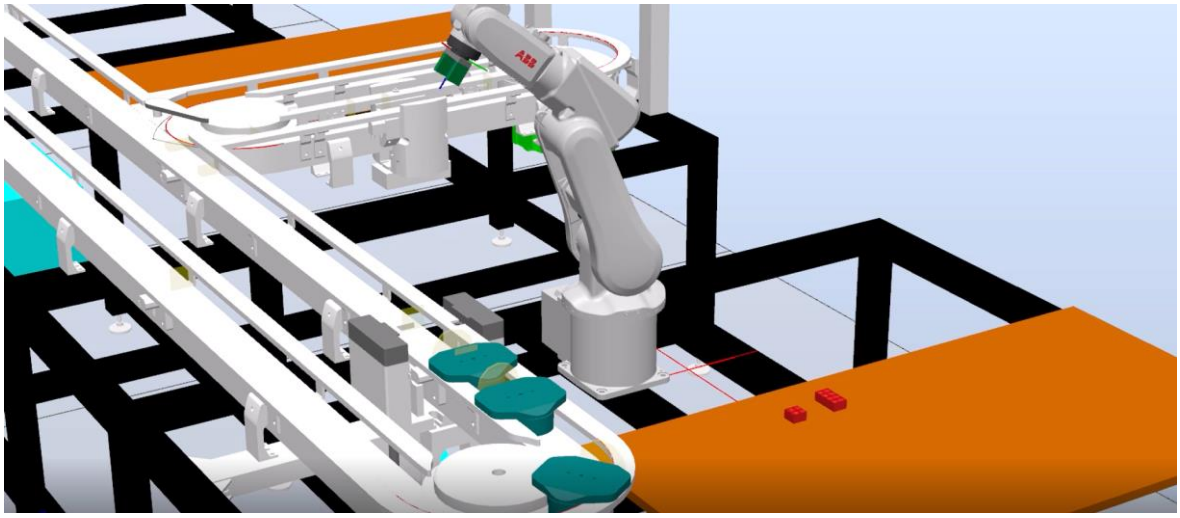

Figura 56 - Demostración. Parte 1.

Los palés llegan al retenedor previo de la estación S1.

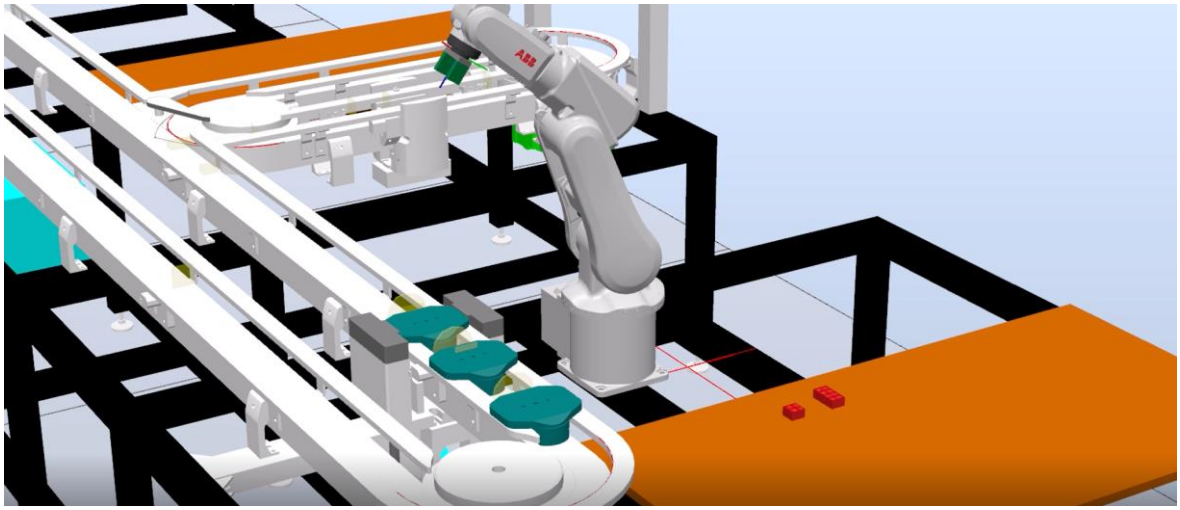

Figura 57- Demostración. Parte 2.

El sensor del retenedor lo detecta y desactiva el retenedor durante el tiempo suficiente como para que pase un único palé. Este palé se frena con el retenedor posterior de S1.

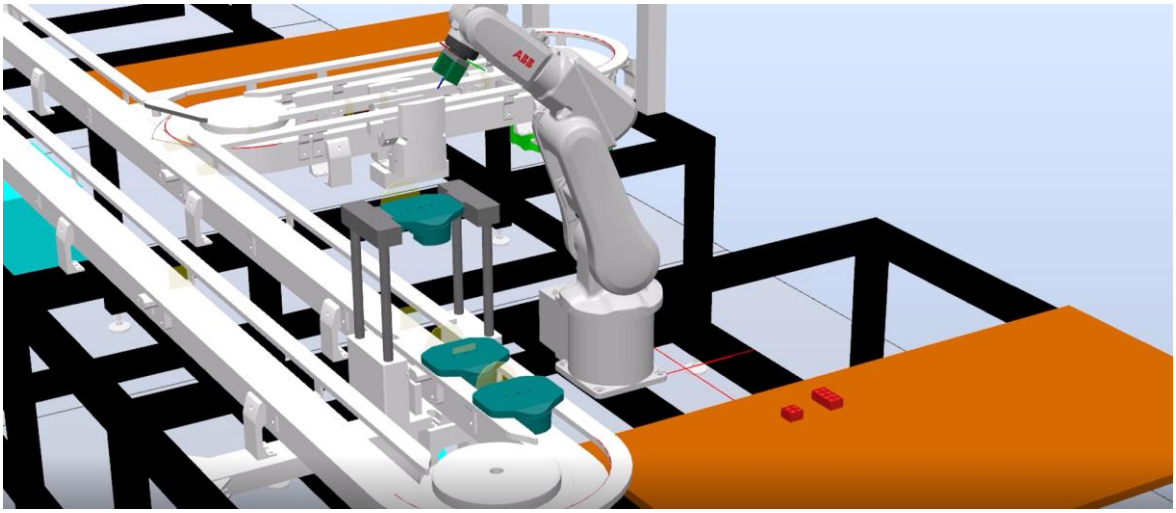

Figura 58 - Demostración. Parte 3.

El retenedor posterior detecta el palé y manda la instrucción de elevarlo para que el robot pueda trabajar con él.

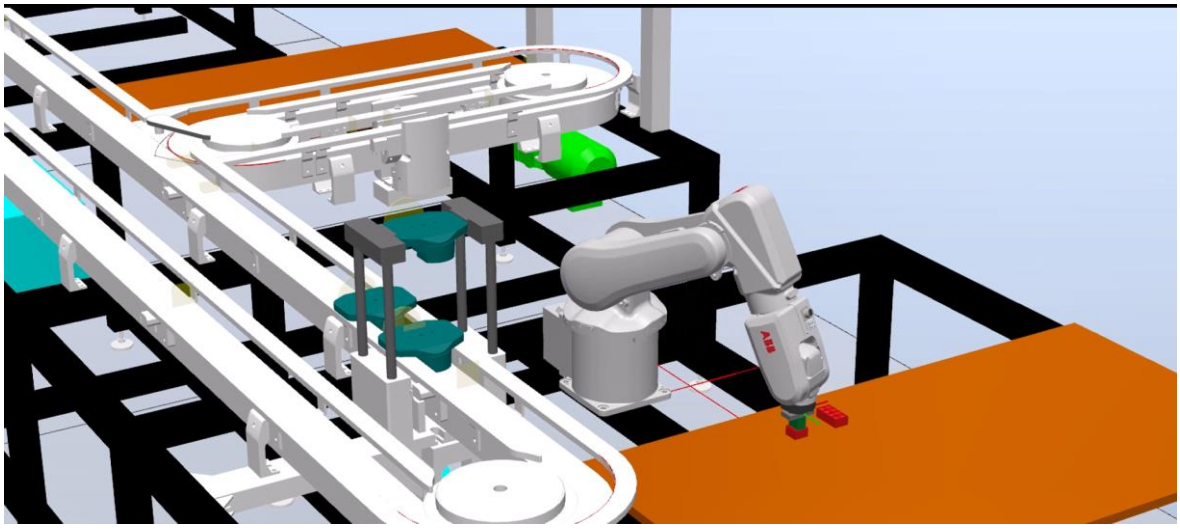

Figura 59 - Demostración. Parte 4.

Una vez elevado se desactivan los retenedores para dejar pasar al resto de palés. Mientras tanto, el IRB120 de S1 coge una pieza de Lego de 2x2. Será la base de la pierna del robot humanoide.

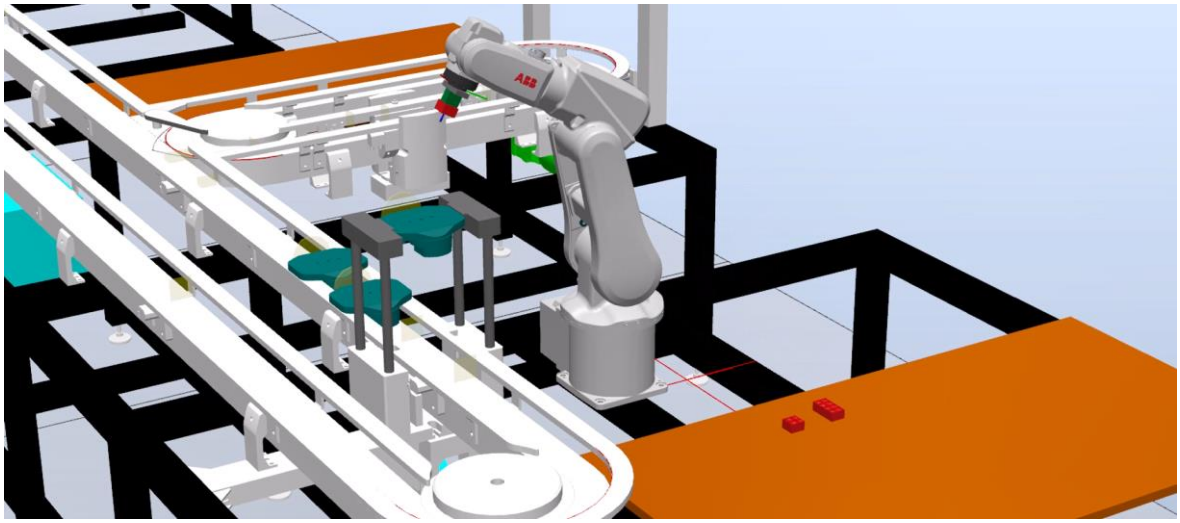

Figura 60 - Demostración. Parte 5.

El IRB120 eleva la pieza de Lego hasta su posición inicial "Home".

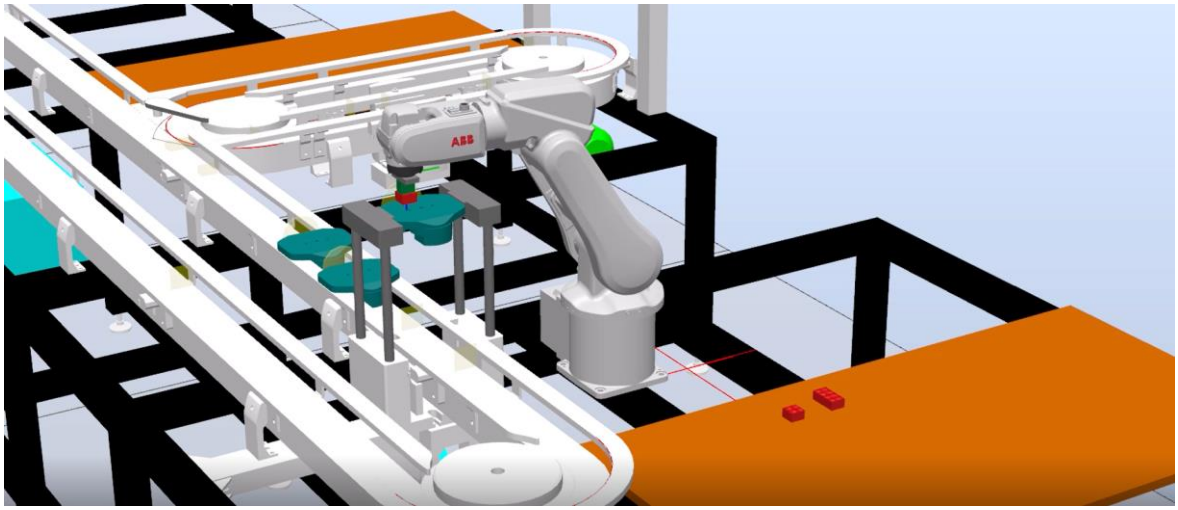

Figura 61 - Demostración. Parte 6.

El robot acerca la pieza a su posición final. En lugar de llegar directamente a esta posición, se para a 20mm por encima de la misma. De esta manera, el final de la trayectoria será totalmente vertical, evitando colisiones con otras piezas y asegurando una buena colocación. Todas las piezas que se coloquen con la función colocaLego() siguen este mismo comportamiento.

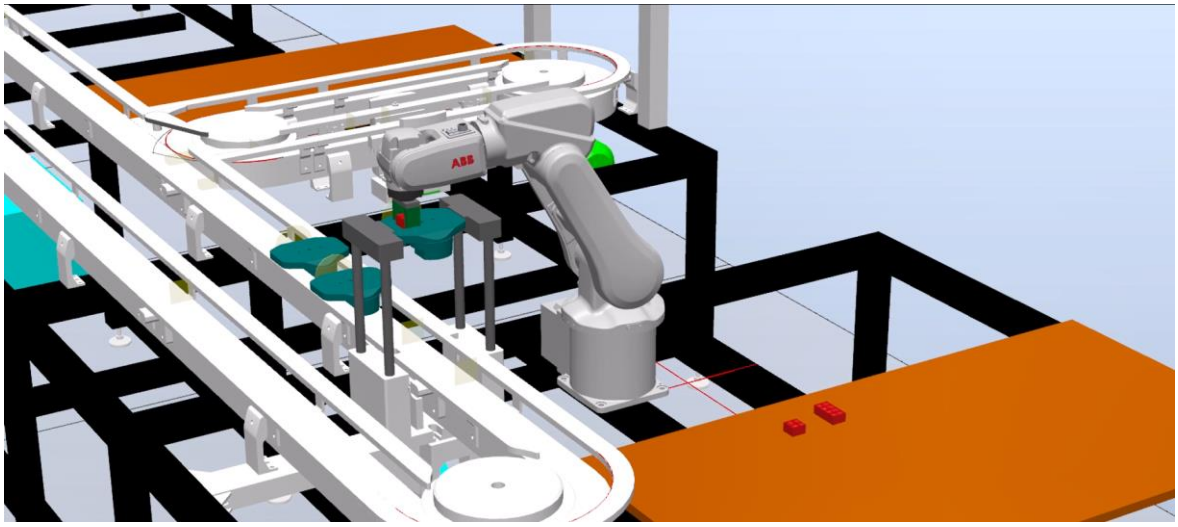

Figura 62 - Demostración. Parte 7.

Se coloca la pieza en su posición final.

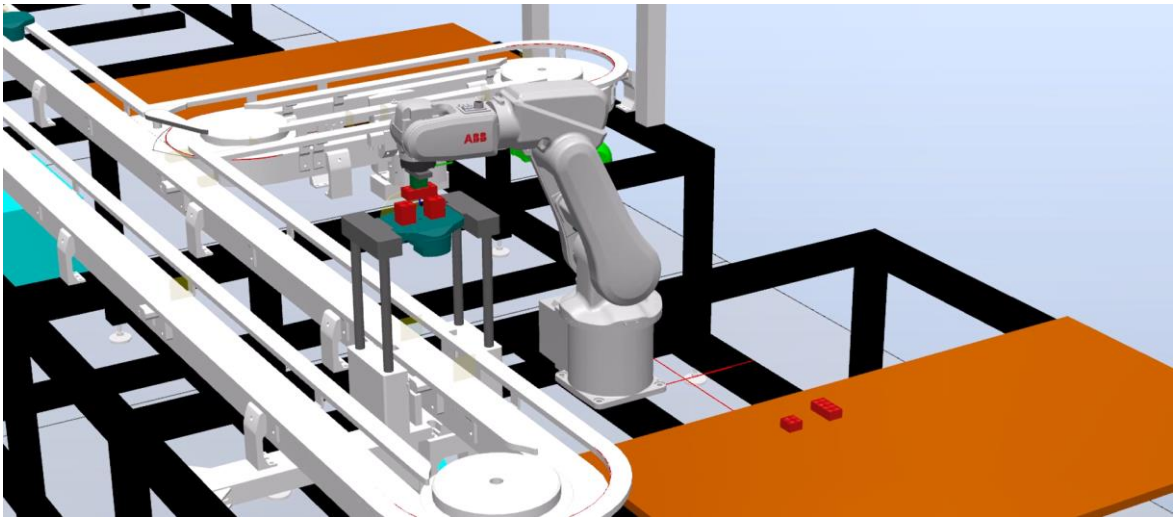

Figura 63 - Demostración. Parte 8.

Se ha repetido el proceso con otras dos piezas de Lego de 2x2 para completar las piernas. En este momento se está colocando una pieza de 2x4 que se ha cogido por su parte central y que se está colocando con la orientación por defecto de la función.

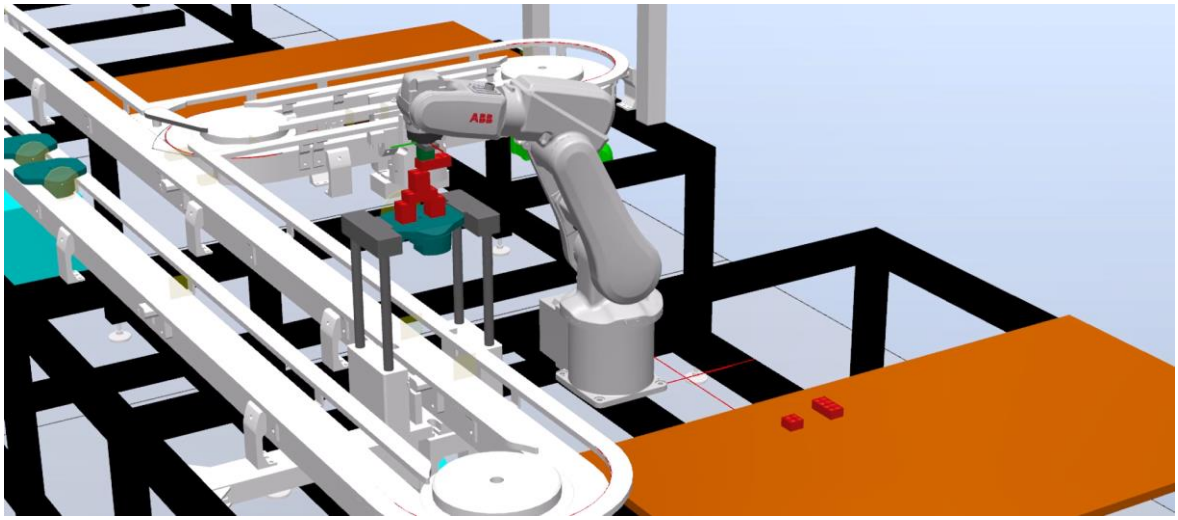

Figura 64 - Demostración. Parte 9.

Se han colocado otras dos piezas de 2x2 para el torso y se está colocando una pieza de 2x4 que se ha cogido por un extremo para que actúe como brazo derecho. Su orientación tiene valor 3, es decir, la opuesta de la orientación por defecto.

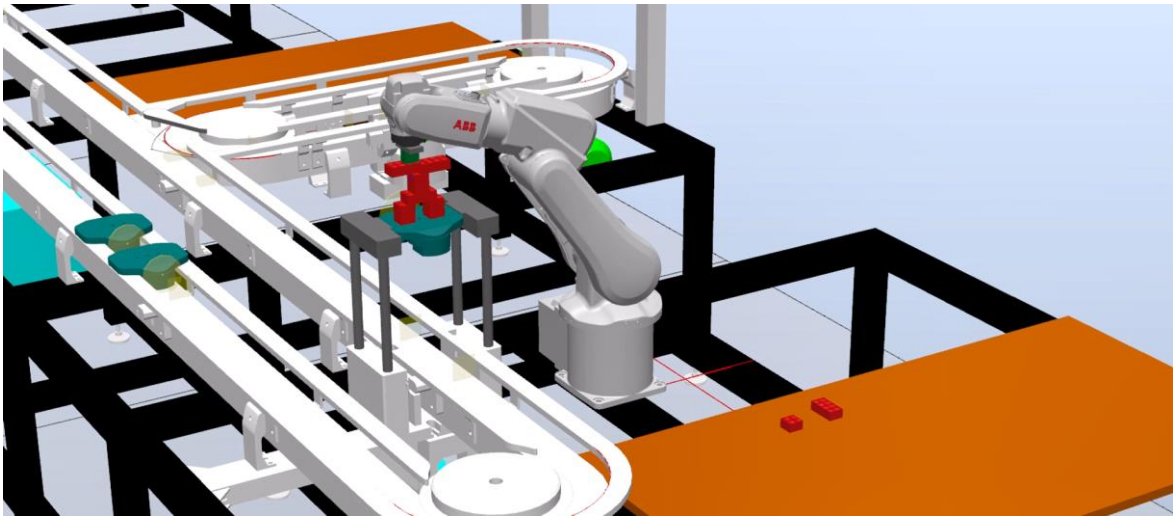

Figura 65 - Demostración. Parte 10.

Se coloca la última pieza de 2x4 para el otro brazo. También se ha cogido por un extremo, pero su orientación es la de por defecto.

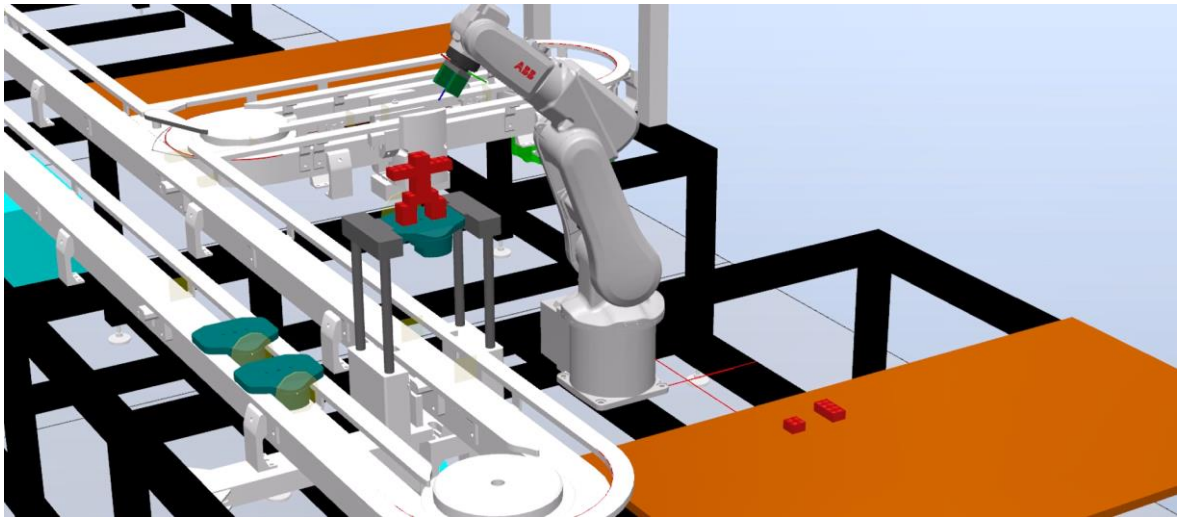

Figura 66 - Demostración. Parte 11.

Se termina la figura con una pieza de 2x2 para la cabeza. Mientras tanto, están llegando los siguientes palés, que ya han recorrido una vuelta entera al circuito S12.

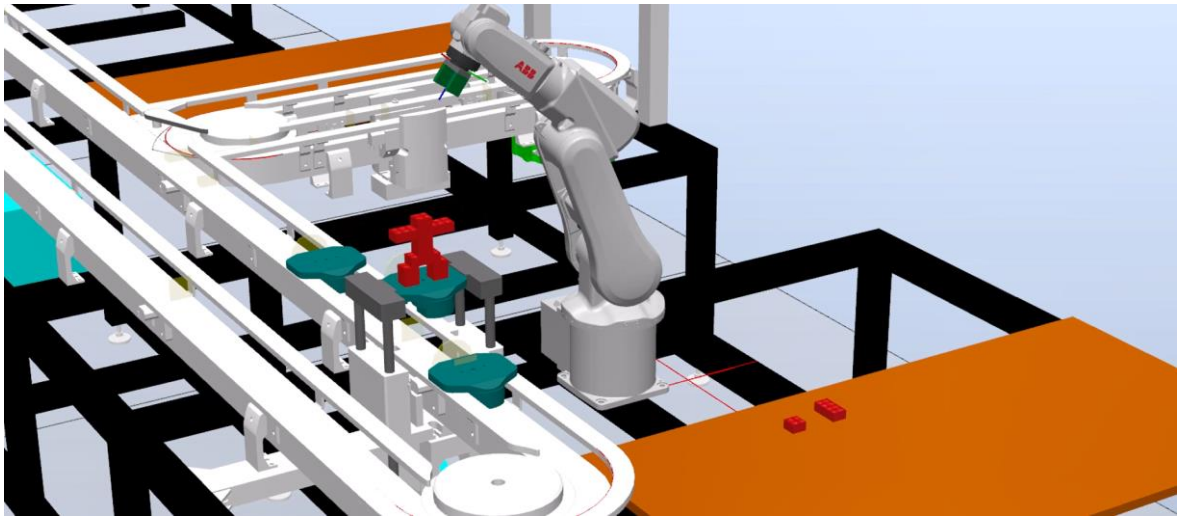

Figura 67 - Demostración. Parte 12.

Cuando el palé sobre el que se está trabajando va a bajar, se activa el retenedor previo y se esperan unos segundos para que un posible palé que esté debajo tenga tiempo de evacuar la zona. En este caso, de los dos palés que venían uno ha podido continuar y otro se ha quedado en el retenedor esperando a que baje el primer palé.

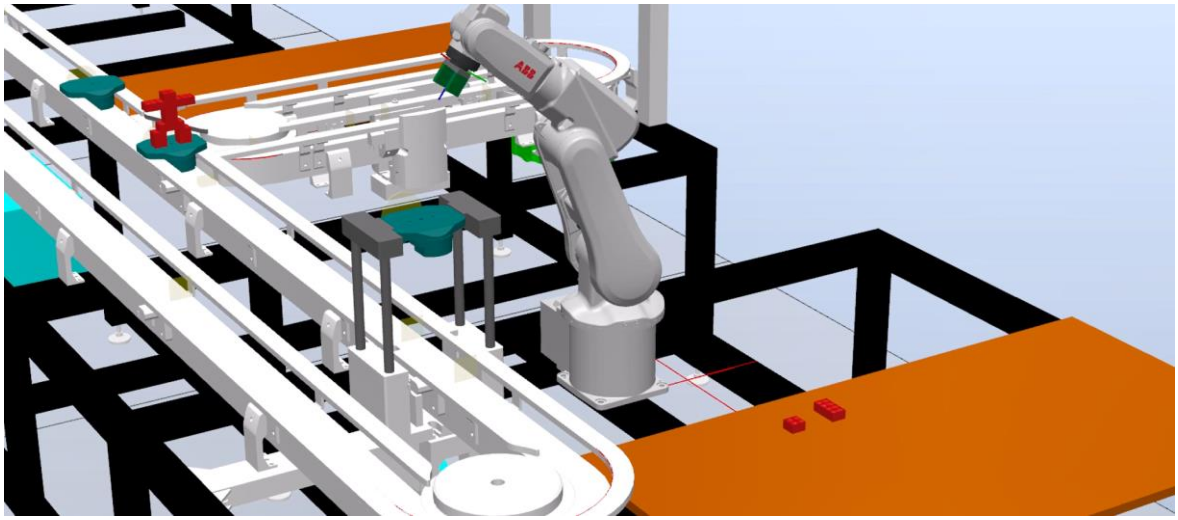

Figura 68 - Demostración. Parte 13.

El palé en el que se ha completado el proceso es libre, mientras que el palé que se quedó en el retenedor comienza el mismo proceso por el que ha pasado el primer palé.

El proceso continúa hasta que los tres palés cuentan con la figura de Lego montada sobre ellos.

Capítulo 6. ANÁLISIS DE RESULTADOS

El gemelo digital se ha programado con especial cuidado para evitar la presencia de fallos en la lógica del programa. Hasta ahora no se ha descubierto ninguno que altere el funcionamiento de la simulación. El modelo, sin embargo, no es perfecto, ya que tiene pequeños desperfectos de diseño:

Las posiciones exactas de los mecanismos, sensores y los robots no han sido comprobadas, así que necesitarán calibración para que el modelo sea lo más exacto posible.

El SmartComponent Palé es clonable, tal como se requería, pero clonarlo requiere conectar manualmente todas las señales necesarias, lo cual es un proceso rápido, pero probablemente evitable.

Los palés detectan la colisión con el entorno utilizando su propia geometría. Sin embargo, según está planteada la lógica, se necesitaba un sensor en la parte delantera del palé para detectar las colisiones con el palé de delante a la vez que se ignora el palé de atrás. Este sensor causa que haya un pequeño hueco entre palé y palé cuando colisionan. No es un fallo importante, ya que la distancia es pequeña, pero no es imperceptible.

Las señales están configuradas de manera que el gemelo digital se comporte de la misma manera que la minifábrica real, pero en el caso del gemelo digital todas las señales entran y salen del controlador de S1, mientras que en el caso real hay cuatro controladores y dos PLC.

Capítulo 7. CONCLUSIONES Y TRABAJOS FUTUROS

Se ha conseguido hacer un Gemelo Digital de la minifábrica de ICAI, que servirá para que varias personas puedan programarla al mismo tiempo sin interferir con la planta real. Para ello se han modelado los mecanismos de retención, localización, guía y una pinza, se ha modelado la lógica de la estación mediante SmartComponents, se ha configurado el controlador del IRB120 de la estación S1 y se ha creado una pequeña demostración del funcionamiento de la misma programando el controlador con código RAPID.

En un futuro se podrá mejorar lo mencionado en el Capítulo 6.

- Calibrar las posiciones de todos los mecanismos, sensores y robots.
- Mejorar el diseño del SmartComponent “Palé”.
    - Clonado más sencillo.
    - Reducción o eliminación del espacio entre palé y palé tras una colisión.
- Redistribución de señales y código RAPID a los cuatro controladores, dos PLC y tres remotas.

Aunque estas mejoras son deseables, el modelo funciona correctamente tal como está. Cualquiera que quiera trabajar con la minifábrica en RobotStudio podrá usar este archivo y trabajar directamente con código RAPID sin tener que invertir tiempo en aprender el funcionamiento de los SmartComponents que constituyen el núcleo de la simulación.

Capítulo 8. BIBLIOGRAFÍA

[1] Conrad Leiva. "Demystifying the Digital Thread and Digital Twin Concepts". Industry Week 2016. August, 2016. https://www.industryweek.com/technology-and-iiot/systems-integration/article/22007865/demystifying-the-digital-thread-and-digital-twin-concepts.

[2] Kennedy, Kara. "What is Digital Twins (+Impact on Business Modernization)", Learn Hub 2018. January 2018. http://learn.g2.com/trends/digital-twins.

[3] Pettey, Christy. "Prepare for the impact of Digital Twins". Smarter With Gartner 2017. September 2017. https://www.gartner.com/smarterwithgartner/prepare-for-the-impact-of-digital-twins/

[4] "RobotStudio: La herramienta de programación offline más utilizada del mundo en robótica". ABB 2020. https://new.abb.com/products/robotics/es/robotstudio

[5] Dong-Soo, Kwon. "The Bone-Mountable Surgical Robot System for Hip Arthroplasty: ARTHROBOT". Conference: Intelligent Robots and Systems, 2001. February 2001. http://www.researchgate.net/publication/3930231_ARTHROBOT_A_new_surgical_robot_system_for_total_hip_arthroplasty

[6] Rodríguez Mondéjar, José Antonio. "Práctica Planta: Integración de robot con planta" 2020.

[7] Rodríguez Mondéjar, José Antonio. "Guías rápidas RobotStudio" 2020.

[8] FlexLink. "Manual Instalación Posicionador Túnel".

[9] FlexLink. "Manual Posicionador Simple".

[10] FlexLink. "Tunnel Locating Module XBUL 11 T".

ANEXO I

Código de la tarea S1:

```
MODULE ControlS1YGSTOP
   VAR bool setupDone := FALSE;
   PROC main()
      IF setupDone THEN

      !!!CUERPO DEL PROGRAMA


      WaitDI I_S1_PSTOP, 1;!esperar a que llegue un pale
      WaitTime 1;

      !dejar pasar 1 palé
      SetDO O_S1_PSTOP_OFF, 1;
      SetDO O_S1_PSTOP_ON, 0;
      WaitTime 0.3; !comprobar tiempo necesario
      SetDO O_S1_PSTOP_OFF, 0;
      SetDO O_S1_PSTOP_ON, 1;

      WaitDI I_S1_STOP, 1; !esperar a que el palé se ponga en posición
      WaitTime 1;

      !subir liftS1
      SetDO O_S1_LIFT_DOWN, 0;
      SetDO O_S1_LIFT_UP, 1;

      !Esperar a que el palé haya subido
      WaitDI I_S1_LIFT_UP, 1;

      !Liberar el flujo de palés
      SetDO O_S1_STOP_ON, 0;
      SetDO O_S1_STOP_OFF, 1;

      SetDO O_S1_PSTOP_ON, 0;
      SetDO O_S1_PSTOP_OFF, 1;


      WaitTime 40; !Esperar a que el robot termine y pasen palés por debajo
```

```
    !Cerrar PSTOP y esperar a que los palés dejen de estar debajo del lift
    SetDO O_S1_PSTOP_ON, 1;
    SetDO O_S1_PSTOP_OFF, 0;
    WaitTime 2;

    !Bajar palé
    SetDO O_S1_LIFT_DOWN, 1;
    SetDO O_S1_LIFT_UP, 0;
    WaitDI I_S1_LIFT_DOWN, 1;

    !Dejar pasar palé y volver a cerrar
    WaitTime 2;
    SetDO O_S1_STOP_ON, 1;
    SetDO O_S1_STOP_OFF, 0;

    ELSE
    setup;
    setupDone := TRUE;
    ENDIF
  ENDPROC

  PROC setup()

    !activar flow global
    SetDO O_GSTOP_OFF, 1;
    SetDO O_GSTOP_ON, 0;

    !activar stopS1
    SetDO O_S1_PSTOP_ON, 1;
    SetDO O_S1_PSTOP_OFF, 0;

    !activar pstopS1
    SetDO O_S1_STOP_ON, 1;
    SetDO O_S1_STOP_OFF, 0;

    !bajar liftS1
    SetDO O_S1_LIFT_DOWN, 1;
    SetDO O_S1_LIFT_UP, 0;

  ENDPROC
ENDMODULE
```

Código de la tarea T_ROB1:

```
MODULE Module1
```

```
   CONST robtarget Home:=[[-60.983,-313.902,732.067],[0.043834723,-0.000002953,-
0.955679747,0.291126669],[-2,-1,-2,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
0.000842734],[0,0,1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
   CONST robtarget Pale:=[[-60.983,-405,492.734],[0,0,1,0],[-2,0,-
2,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
   CONST robtarget Lego:=[[396.686,179.771,75.734],[0.000821186,-0.706698924,-
0.707513423,-0.000844909],[0,0,1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
   CONST robtarget Lego2:=[[396.686,279.771,75.734],[0.000821186,-0.706698924,-
0.707513423,-0.000844909],[0,0,1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
   PERS robtarget paleTarget:=[[-60.983,-389,588.734],[0,1,0,0],[-2,0,-
2,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
   CONST robtarget Pale_H:=[[-60.983,-405,492.734],[0,0.707106781,0.707106781,0],[-
2,0,-1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
   CONST robtarget Pale_rot_4:=[[-60.983,-405,492.734],[0,-
0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
   CONST robtarget Pale_rot3:=[[-60.983,-405,492.734],[0,1,0,0],[-
2,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
   CONST robtarget
Lego3:=[[380.686010624,279.75257064,75.733441498],[0.000821186,-0.706698924,-
0.707513423,-0.000844909],[0,0,1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

   PROC colocaLego (num x, num y, num z, num rot, bool largo, bool centro)
      paleTarget.trans.x := Pale.trans.x + 32/2*x;
      paleTarget.trans.y := Pale.trans.y + 32/2*y;
      paleTarget.trans.z := Pale.trans.z + 19.2*z;
      IF rot = 1 THEN
         paleTarget.rot.q1 := Pale.rot.q1;
         paleTarget.rot.q2 := Pale.rot.q2;
         paleTarget.rot.q3 := Pale.rot.q3;
         paleTarget.rot.q4 := Pale.rot.q4;
      ENDIF
      IF rot = 2 THEN
         paleTarget.rot.q1 := Pale_H.rot.q1;
         paleTarget.rot.q2 := Pale_H.rot.q2;
         paleTarget.rot.q3 := Pale_H.rot.q3;
         paleTarget.rot.q4 := Pale_H.rot.q4;
      ENDIF
      IF rot = 3 THEN
         paleTarget.rot.q1 := Pale_rot3.rot.q1;
         paleTarget.rot.q2 := Pale_rot3.rot.q2;
         paleTarget.rot.q3 := Pale_rot3.rot.q3;
         paleTarget.rot.q4 := Pale_rot3.rot.q4;
      ENDIF
      IF rot = 4 THEN
```

```
        paleTarget.rot.q1 := Pale_rot_4.rot.q1;
        paleTarget.rot.q2 := Pale_rot_4.rot.q2;
        paleTarget.rot.q3 := Pale_rot_4.rot.q3;
        paleTarget.rot.q4 := Pale_rot_4.rot.q4;
    ENDIF


    IF largo THEN
        IF centro THEN
            MoveL Lego3,v1000,fine,Pinza_1\WObj:=wobj0;
        ELSE
            MoveL Lego2,v1000,fine,Pinza_1\WObj:=wobj0;
        ENDIF

    ELSE
        MoveL Lego,v1000,fine,Pinza_1\WObj:=wobj0;
    ENDIF
    SetDO D10_2,1; !Cerrar pinza
    MoveL Home,v1000,z100,Pinza_1\WObj:=wobj0;

    paleTarget.trans.z := paleTarget.trans.z + 20;
    MoveL paleTarget,v1000,fine,Pinza_1\WObj:=wobj0;

    paleTarget.trans.z := paleTarget.trans.z - 20;
    MoveL paleTarget,v1000,fine,Pinza_1\WObj:=wobj0;

    SetDO D10_2, 0; !Abrir pinza
    MoveL Home,v1000,z100,Pinza_1\WObj:=wobj0;
ENDPROC

PROC main()
    SetDO D10_2, 0; !Abrir pinza
    MoveL Home,v1000,z100,Pinza_1\WObj:=wobj0;
    WaitDI I_S1_LIFT_UP, 1;!esperar a que llegue un pale
    WaitTime 1;

    colocaLego 0, -2, 0, 1, FALSE, TRUE; !colocación de las dos piernas
    colocaLego 0, 2, 0, 1, FALSE, TRUE;
    colocaLego 0, -2, 1, 1, FALSE, TRUE;
    colocaLego 0, 2, 1, 1, FALSE, TRUE;

    colocaLego 0, 0, 2, 1, TRUE, TRUE; !unión de las piernas con pieza larga

    colocaLego 0, 0, 3, 1, FALSE, TRUE; !torso
    colocaLego 0, 0, 4, 1, FALSE, TRUE;
```

```
    colocaLego 0, 1, 5, 3, TRUE, FALSE; !brazo 1
    colocaLego 0, -1, 5, 1, TRUE, FALSE; !brazo 2

    colocaLego 0, 0, 6, 1, FALSE, TRUE; !cabeza

    WaitDI I_S1_LIFT_DOWN, 1;!esperar a que se vaya el pale
  ENDPROC
ENDMODULE
```