Universidad Zaragoza
1542

Trabajo Fin de Máster

Desarrollo de un sistema de producción en Factory I/O y control avanzado mediante Redes de Petri

Design of a manufacturing system in Factory I/O and advanced control based on Petri Nets

Autor

Fernando Grima Montesa

Director

Cristian Florentín Mahulea

Escuela de Ingeniería y Arquitectura /Universidad de Zaragoza

Curso 2020-2021

Índice

Índice de figuras

Índice de tablas

Resumen

El presente trabajo trata conceptos avanzados en el ámbito de desarrollo, modelado y técnicas de control en sistemas de producción de mediano y gran tamaño. El objetivo que persigue este trabajo es la investigación, en temas y conceptos de la ingeniería de control, sobre los métodos de control basados en las redes de Petri. Se plantea la problemática que surge en el control de sistemas de un tamaño considerable, donde se deben estar realizando varias operaciones a la vez y existe cierta concurrencia y/o conflicto entre las diversas actividades que se ejecutan. Para poder trabajar con un sistema de tal magnitud, se ha dispuesto de la ayuda del simulador Factory I/O.

Factory I/O es una herramienta de simulación de sistemas de producción industrial desarrollada en 2014. Desde hace unos años es un software muy utilizado para el apoyo de estudiantes, y que sigue en auge en el desarrollo e implementación de sistemas de control avanzados en simulaciones. Tras barajar varias posibilidades, se ha creado una escena de simulación basada en lo que podría ser una cadena de montaje de una empresa industrial mediana.

Como se comentaba, el sistema de control que se ha implementado se basa en las redes de Petri, más concretamente en las redes de Petri coloreadas (CPN, *coloured petri nets*). Las CPN se diferencian de las redes de Petri lugar/transición, en que se añade más información a la red, aumentando su complejidad, pero reduciendo así el número de lugares y transiciones. Se conocen también como redes de alto nivel. Desde el punto de vista del modelado y análisis, esto presenta una gran ventaja porque reduce notablemente el tamaño de la red, facilitando mucho el desarrollo de estas dos etapas esenciales para el diseño del controlador. Se exponen las técnicas de modelado y análisis utilizadas que se corresponden con esta técnica de control (refinamiento, grafo de alcanzabilidad, técnicas de reducción y simulación, etc.), para aplicarlas se necesita de herramientas potentes capaces de computar los modelos creados.

CPN tools (*coloured petri net tools*) es una herramienta muy conocida y utilizada en el ámbito de las CPN. Desarrollada entre el año 2000 y 2010, también es un software dedicado a la docencia y el aprendizaje de esta metodología de modelado y análisis de sistemas de control.

Este trabajo planteaba la posibilidad de llevar más allá estos principios y poder llegar a implementar, sobre un sistema simulado en este caso, el modelo obtenido y analizado. Lamentablemente, como se verá más adelante, la comunicación entre CPN tools y Factory I/O todavía no es viable y no se ha llegado a implementar el controlador desde la herramienta. Finalmente, para la implementación se ha utilizado la herramienta MATLAB. Por este motivo, uno de los trabajos futuros que se plantea al final de este documento es la implementación del controlador desde CPN tools cuando el equipo de desarrollo del programa haya proporcionado esta posibilidad.

Capítulo I Motivación y Objetivos

1.1 Motivación del trabajo

Con el auge de la industria 4.0, el sector industrial se está viendo envuelto en una época de cambios drásticos en la sistemática e implementación de las mecánicas de producción. Estos grandes cambios provocan la necesidad de desarrollar nuevos sistemas de control para que guíen y controlen las distintas etapas de la producción.

Las redes de Petri son una representación matemática y gráfica de este tipo de sistemas. Su parte gráfica ayuda, notablemente, a la visualización y comprensión de la estructura y el comportamiento del sistema modelado. Usualmente, las redes de Petri se utilizan para modelar los sistemas y así poder analizar propiedades de estos que pueden llegar a ser vitales para el buen funcionamiento y desarrollo del sistema.

Con este trabajo, se pretende motivar la buena praxis a la hora del desarrollo de las etapas de modelado y análisis de sistemas concurrentes. En muchas ocasiones son una parte del proceso que no se tiene en cuenta, bien por falta de tiempo, bien por ignorancia, pero que resultan de vital importancia para evitar posibles fallos en el buen funcionamiento del sistema de control que se desee implementar.

El hecho de realizar un buen análisis asegura que el sistema cumple perfectamente con las especificaciones deseadas, evitando así posibles bloqueos (tanto parciales como totales), lugares ilimitados (desbordamiento de materiales) o viendo si se cumplen propiedades tan importantes como la reversibilidad (en el Capítulo IV: Modelado y análisis del sistema de control, se tratan más en detalle estos temas).

1.2 Alcance y objetivos del trabajo

El objetivo principal de este trabajo fin de máster consiste en modelar, analizar e implementar un sistema de control avanzado, basado en CPN, en un sistema de producción simulado. Como se ha comentado en el apartado anterior, la gran ventaja de utilizar las CPN para implementar el sistema de control es utilizar técnicas de análisis formal (matemáticas) del sistema controlado antes de la implementación del controlador. El primer objetivo del trabajo ha sido el diseño de la escena de simulación a un nivel de complejidad, permitido por la herramienta de simulación, que se igualase al de un sistema de producción real (Capítulo III: Descripción del sistema de producción). El segundo ha sido el modelado y análisis formal basado en CPN del sistema, evaluando de esta manera propiedades indispensables como por ejemplo vivacidad o especificaciones como la verificación de que no existan lugares de capacidad infinita (Capítulo IV: Modelado y análisis del sistema de control). Por último, se ha implementado y verificado el correcto diseño y funcionamiento del sistema controlado utilizando la simulación (Capítulo V: Implementación del sistema de control y Capítulo VI: Verificación del sistema de control desarrollado).

El alcance de este trabajo ha llegado hasta la implementación de un sistema de control de puesta en marcha y funcionamiento del sistema, englobando las operaciones desde la llegada de materia prima hasta la salida de fábrica del producto. Como se comentará más en detalle en los próximos capítulos, el sistema globalmente puede encontrarse en tres estados: puesta en marcha, funcionamiento (donde está implementado el controlador), parada de producción o parada por emergencia (fallo del sistema). El alcance del trabajo abarca los tres primeros estados, en el caso de la parada por emergencia y vuelta al estado de funcionamiento ha quedado fuera del alcance del trabajo (a pesar de ello este trabajo sienta las bases para el desarrollo en profundidad de este apartado).

1.3 Estado del arte

Actualmente las redes de Petri coloreadas se utilizan para describir procesos de eventos discretos de naturaleza concurrente. El motivo de la elección de este tipo de metodología es que, a diferencia de los métodos tradicionales (como modelos de simulación), las redes de Petri proveen una representación gráfica muy intuitiva para el estudio de sistemas complejos, entre otros, destaca el uso de estas redes para el modelado de sistemas flexibles de producción. Recientemente se han realizado varios trabajos relacionados con la efectividad que poseen los modelos basados en esta técnica frente a los modelos tradicionales [1,2,3]. Hacen hincapié en el gran efecto que produce el haber modelado, analizado y verificado este tipo de redes sobre la descripción de la información procesada, los sistemas de microprocesadores y otros, ya que en estos sistemas existe una notable restricción en cuanto a tolerancia de errores y relaciones de tiempos.

Otro campo en el que es muy común el uso de redes coloreadas es el área de las comunicaciones. Las CPNs son una gran ventaja en este campo por el hecho de presentar los flujos que se dan en el sistema. Se han introducido muchos modelos ya, basados en la composición de los servicios Web, conexión Ethernet, o incluso en la orientación de la prestación de un servicio como podría ser la compra de un libro vía web [4,5].

Con respecto a los sistemas de producción, en el mayor de los casos los controladores se implementan en controladores lógico programables (PLC) para realizar el seguimiento e implementación del sistema de control diseñado. Existe ya algún trabajo en el que se emplea la formalidad de las redes de Petri para implementarla en un PLC mediante texto estructurado. Con el auge de los nuevos lenguajes de programación como Python, C, C++, MATLAB, Java y más, los PLCs están empezando a evolucionar para poder adaptar los códigos escritos en estos lenguajes de alto nivel a su lógica programable. La flexibilidad que se presenta con la tecnología IoT (*Internet of things*) demanda que los PLCs puedan entrar en contacto con sistemas de negocios, análisis de datos… ampliando así las posibilidades y capacidades de los sistemas automatizables [6].

La implementación del controlador no se ha realizado en un PLC debido a la falta de librerías de manejo de matrices en estándares como el texto estructurado, ya que, para este sistema, se deben manejar matrices de gran tamaño que puede aumentar en cada iteración. Es por este motivo por el cual este proyecto se centra en el modelado, análisis y la posibilidad de implementación utilizando una herramienta programación de alto nivel, donde se posea una librería de manejo de matrices más flexible, quedando a expensas de poseer PLCs en los que se puedan utilizar estas librerías.

Capítulo II CPN tools y las redes de Petri coloreadas

Para ayudar al lector que no esté familiarizado con la materia, las redes de Petri coloreadas son un formalismo para el modelado y análisis de sistemas de eventos discretos en los que la concurrencia, comunicación y otros tipos de sincronización juegan un papel crucial en el desarrollo de estos. Gracias a la información que aportan las CPN, se mejora la observación y análisis de las propiedades y especificaciones principales del sistema que engloba. Algunas de las propiedades y especificaciones más importantes de estos sistemas, como ya se ha detallado, son la vivacidad de la red (si todos los eventos de la red pueden llegar a ocurrir durante la ejecución de esta), la reversibilidad (si la red puede alcanzar en algún momento el marcado inicial), los posibles bloqueos que puedan aparecer, aparición de lugares de capacidad infinita, gestión de los recursos compartidos, etc(se entrará en mayor profundidad en apartados posteriores).

Las redes de Petri fueron inventadas por Carl Adam Petri en 1939 a la edad de 13 años. Estas redes se han utilizado en una infinidad de campos tecnológicos, incluyendo la ciencia computacional, la química o la biología [7].

Son un sistema de descripción formal (definido matemáticamente) que tiene detrás una representación gráfica. Se emplean para modelar y analizar sistemas concurrentes, es decir, sistemas secuenciales que evolucionan en paralelo y pueden finalizar en cualquier momento. La representación gráfica de las redes de Petri es muy útil para percibir fácilmente la estructura del sistema y así, razonar también de manera más sencilla, si el sistema se comporta cómo debe ser o cómo se desea que sea. Dentro de las redes de Petri existen dos nodos: lugares y transiciones. Estos lugares modelan las operaciones o actividades que se realizan a lo largo del funcionamiento del sistema o los recursos disponibles, mientras que, por otro lado, las transiciones conforman las condiciones necesarias para dictaminar si una operación se da por concluida, o no, para pasar a la siguiente.

La unión de estos lugares y transiciones se realiza mediante arcos, que pueden tener un peso distinto entre ellos (o unas expresiones como se verá más adelante en las redes coloreadas). Las uniones en una red de Petri deben cumplimentar que se realizan entre lugar-transición o transición-lugar, nunca entre dos nodos del mismo tipo.

Cuando una transición se dispara, consume de todos los lugares entrantes a la transición, un número de marcas igual al peso del arco que los une. En caso de no tener ese marcado, la transición no se puede disparar (no está sensibilizada). Si se dispara la transición, se producirán en los lugares de salida de la transición, un número de marcas igual al peso que posea el arco que los une. El marcado de la red viene dado por un vector columna, donde los elementos del vector son el número de marcas que posea cada lugar en el estado en el que esté el sistema.

Durante el diseño de la red pueden aparecer diversas tipologías de estructuras. Algunas de ellas son de gran importancia ya que facilitan reparar, como se decía, en el correcto comportamiento del sistema. Por ejemplo, algunas de las estructuras que aparecen en este trabajo son: el conflicto

(como por ejemplo el giro de un motor, no puede ocurrir que el motor pueda girar en ambos sentidos a la vez), la concurrencia, la sincronización o el recurso compartido.

Para facilitar al lector la comprensión del uso de las redes de Petri, a continuación, se explica brevemente un ejemplo:

Se pretende ejecutar una pequeña secuencia en el movimiento de dos carros (cada carro modelando un sistema secuencial). Se desea que comience su movimiento cuando se pulse el botón marcha, partiendo ambos carros del lado izquierdo. Cuando uno de los dos carros llegue al extremo derecho, esperará a que el otro carro también llegue a su relativo extremo. Acto seguido, los carros volverán a su posición inicial y esperarán de nuevo a que el otro carro llegue a su posición y a que el pulsador de marcha vuelva a ser presionado de nuevo. Este conjunto de actividades/operaciones y condiciones se transforma en la siguiente red de Petri (Figura 1):

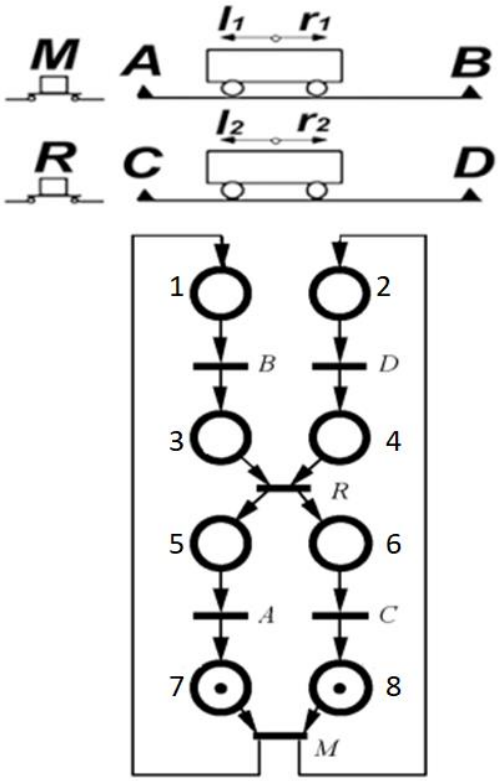


Figura 1: Ejemplo carros red de Petri, Fuente [8]

Los lugares 1 y 2 indican el movimiento de los carros hacia la derecha, donde se ejecutan las acciones r1 y r2, respectivamente. Después, los lugares 3 y 4 son los lugares de espera de los carros una vez alcanzados los extremos “B” y “D” respectivamente para cada carro. Antes de pasar a los lugares 5 y 6, el sistema debe pasar por una sincronización (transición R) para ejecutar las operaciones contrarias que son el movimiento opuesto, hacia el punto de partida (acciones l1 y l2). Alcanzado los extremos “A” y “C” se pasa a los lugares de espera 7 y 8 y llegados a este punto el sistema vuelve a su situación inicial esperando a la siguiente sincronización (transición M). La red también muestra muy bien la concurrencia del sistema a la hora de ejecutar sus actividades, los carros no tienen por qué ir a la misma velocidad y por tanto llegar al destino al mismo tiempo.

En sistemas de producción, una parte de la complejidad del sistema puede derivar de la existencia de muchos sistemas secuenciales que se ejecuten paralelamente con comportamientos similares, u objetivos comunes dentro del planteamiento de proceso deseado. Bajo estas condiciones de significantes simetrías en los componentes, surge un interés de poder plantear redes de Petri de alto nivel.

Existen varias extensiones de las redes de Petri comunes (lugar/transición): coloreadas, estocásticas, temporales, continuas, híbridas, etc. Estas tipologías de redes surgen de la necesidad de poder modelar sistemas de mucha complejidad, que en redes de Petri normales no sería viable realizar y analizar un modelo.

Este trabajo fin de máster se centra en las CPNs. Las redes coloreadas con respecto a las redes lugar/transición, se diferencian en que se puede asignar propiedades a las marcas y de este modo se puede distinguir entre dos marcas existentes en un mismo lugar, lo que no es posible en redes de Petri lugar/transición.

Por consiguiente, las CPNs vienen definidas por tipos de colores. Son de unos tipos de datos predefinidos, suelen tratarse de los tipos de datos utilizados en el lenguaje de programación: *strings*, *integers*, *floats*, *bool*… estos tipos de colores definen los colores de conjuntos y multiconjuntos que aparecen en los lugares de la red. Cada elemento del conjunto puede tener una multiplicidad, por ejemplo, {a,b} es un conjunto de marcas a y b, pero {a,a,b} no es un conjunto, en realidad se trataría de un multiconjunto {2`a,b}, dos marcas de a y una de b (se pondrá un ejemplo en el modelado de la red más adelante).

En el caso de los arcos, ahora se podrá imponer expresiones de mayor complejidad, como expresiones lógicas tipo “if then else”[9].

Continuando con el ejemplo planteado anteriormente y así aclarar lo dicho hasta ahora, se puede ver que, en caso de necesitar un número mayor de carros, el número de lugares y transiciones aumentaría en cuatro lugares y dos transiciones por carro (sistema secuencial) añadido. Sin embargo, con una red de Petri coloreada, la estructura de red no aumentaría, es más, solo serían necesarios cuatro lugares y cuatro transiciones, el tamaño de red dedicado a un solo carro. A cambio, la red gana nivel de complejidad ya que sería necesario crear un tipo de color llamado carro, por ejemplo, y dentro de él habría un color referenciando el número de carro correspondiente. También sería necesario realizar modificaciones en los arcos.

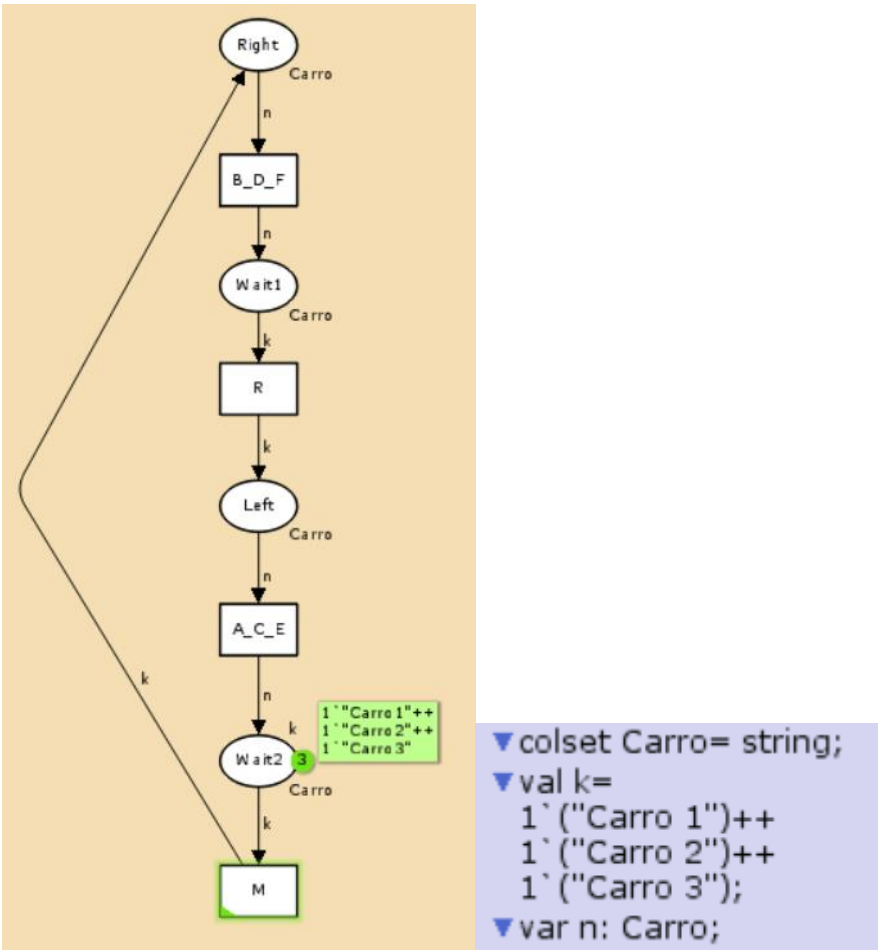


Figura 2: Ejemplo carros red coloreada

Los parámetros definidos como se puede apreciar son: el tipo de color (*colset*) llamado Carro que será de tipo *string*, la constante (val) "k" que define los colores de la red, es decir, las marcas posibles que pueden aparecer y la variable "n", que se utilizará para definir los pesos de algunos de los arcos.

Como se puede observar, se gana mucho tiempo y espacio en cuanto al planteamiento de la red. De este modo, simplemente con modificar la constante "k", para el número de carros totales, sería suficiente para modelar el sistema. La variable "n" se refiere a un solo carro, es decir, se necesita solo un color para poder sensibilizar la transición y por ello este valor puede ser variable, mientras que, las transiciones con "k" en sus arcos, necesitan que estén todos los colores en el lugar predecesor para sensibilizar la transición y por ello se puede definir como una constante (también producen la misma cantidad de colores en el lugar posterior).

De ahí que las CPNs se definan como una tupla:

$$CPN = (C; B, P; T; V, F)$$

Donde C es el conjunto finito de tipos de colores (*colsets*); B es el conjunto de marcas/colores creados, pertenecientes a los conjuntos del conjunto C; P es el conjunto finito de lugares que hay en la red, los cuales, como ya se ha explicado, son de un tipo de color, y poseen un vector de marcas de tamaño igual a B, cuyos elementos corresponden a la cantidad de marcas, de cada color definido en B, que aparecen en el conjunto o multiconjunto de dicho lugar; T es el

conjunto de transiciones; V el conjunto de variables utilizadas en los arcos de los tipos de multiconjunto C y por último, F es el conjunto de funciones utilizadas en las expresiones que se puedan usar en los arcos, guardias (determinan si es posible el disparo de una transición), etc[10].

El estado de una CPN es distribuido y representado por la unión de las marcas de todos los lugares pertenecientes a la red (el vector marcado).

Este es el caso, por ejemplo, de la red que concierne este trabajo. Se desea trabajar con tres tipos de piezas diferentes, de pintura azul, verde y gris (se tratará más en detalle en el Capítulo III). Para cada tipo de pieza, se realizan unas actividades, que, vistas en totalidad, son las mismas, pero vistas desde la perspectiva de una red de Petri lugar/transición, al no tener colores, se tendrían que considerar como independientes y serían varias subredes, como en el ejemplo expuesto de los carros. No obstante, con la red de Petri coloreada, se puede realizar solo una red que, dependiendo del tipo de pieza, actúe de una manera o de otra, reduciendo así en gran medida el número de lugares y transiciones que pudiesen aparecer (siendo exponencial este número, según se aumentase la cantidad de tipos de piezas).

Es por este motivo por el cual las CPN aparecen como una solución al problema del aumento del número de lugares que se dan en las redes de Petri sin colorear, simplificando así su desarrollo.

CPN tools es una herramienta de análisis de CPN muy potente y gratuita, para modelar y analizar el comportamiento de los sistemas discretos mediante la generación del espacio de alcanzabilidad y análisis de rendimiento mediante simulación. Su uso es relativamente sencillo, dado que el usuario interactúa con el programa mediante la interfaz gráfica de las redes de Petri. Es por ello por lo que este trabajo fin de máster se centra en investigar también acerca de la posible implementación de sistemas de control basados en CPN desde la herramienta CPN tools.

Capítulo III Descripción del sistema de producción

Como objetivo de este trabajo, el sistema que se quería representar tenía que ser de un tamaño y complejidad comparables a lo que son los sistemas de producción reales en sectores manufactureros de medianas y grandes empresas. De manera análoga, se ha pretendido que apareciese la mayoría de las estructuras típicas dadas en las redes de Petri, para así dar un análisis más completo del sistema. El sistema de producción que se ha simulado ha sido desarrollado con la mayor representatividad posible, dentro de lo que podían ofrecer las herramientas y objetos diseñados por Real Games en Factory I/O.

Se pretendía desarrollar un sistema que permitiera dar una visión genérica de las dificultades que existen, a la hora de diseñar e implementar un sistema de control de tal magnitud (Capítulos IV y V).

Se ha tratado de darle más profundidad al modelo en el sentido de diversidad del tipo de proceso productivo y pueden distinguirse dos partes. Al comienzo del sistema, se dispone de un total de cinco estaciones que van a producir las dos piezas fundamentales de la producción. Dos de ellas (M3) se dedican a producir aleatoriamente una cantidad de piezas bases que posteriormente serán identificadas con un sensor de visión. A su vez, las otras tres estaciones (M1) se dedican, exclusivamente, a la producción de las tapas de cada color que se unirán más tarde a las piezas bases correspondientes (existen muchas casuísticas posibles para esta parte de la estación, por simplificación se ha escogido esta).

Cuando la pieza base se ha clasificado con el sensor de visión, pasa a la espera de la estación número dos (M2 o Robot 2 en Factory I/O). Una vez se detecta una pieza base esperando, el Robot 1 elige la pieza tapa correspondiente y la introduce a la cinta de tapas de M2. Después de realizar las operaciones necesarias en M2, las cintas de transporte y el Robot 3 llevan la pieza, ya unida, al almacén intermedio donde se va a almacenar hasta su pedido por la parte del post proceso de embalaje o puesta en caja.

Esta parte es la parte más continua de producción, seguirá produciendo hasta que se alcance un límite de marcas, que se descongestionará extrayendo material del almacén intermedio (Figura 3). La siguiente parte de la instalación depende más del usuario, será este quién determine cuándo y cómo se ejecutan las operaciones.

Figura 3: Etapa de procesado de bases y tapas

Tras extraer del almacén las piezas, se llevan al Robot 4 que será el encargado de llevarlas a embalado o, llevarlas a puesta en cajas, según la decisión que tome el usuario. Se empezará con el post proceso cuando el almacén llegue a un límite de stock máximo en el almacén intermedio. Cada uno de estos procesos consta de un transporte hasta la estación que realiza las operaciones principales.

En el caso de la puesta en caja, simplemente se trata del Robot 6 que pone en las cajas apilables el número deseado de piezas. Como se ha comentado anteriormente, para esta parte de la instalación, se podrían dar infinidad de casuísticas, por lo que la elegida es meramente una puntualidad. Se ha escogido embalar y poner en cajas una cantidad de dos unidades por operación (detallado más adelante en el Capítulo IV).

La parte de embalado es más compleja, pues se tiene que embalar primero mediante el Robot 5 y posteriormente poner las cajas en palés con el Robot 7. Ambos procesos llevan sus productos a la cinta transportadora del almacén final, donde se guardarán las piezas hasta que se solicite su traslado a la zona de envío (Figura 4).

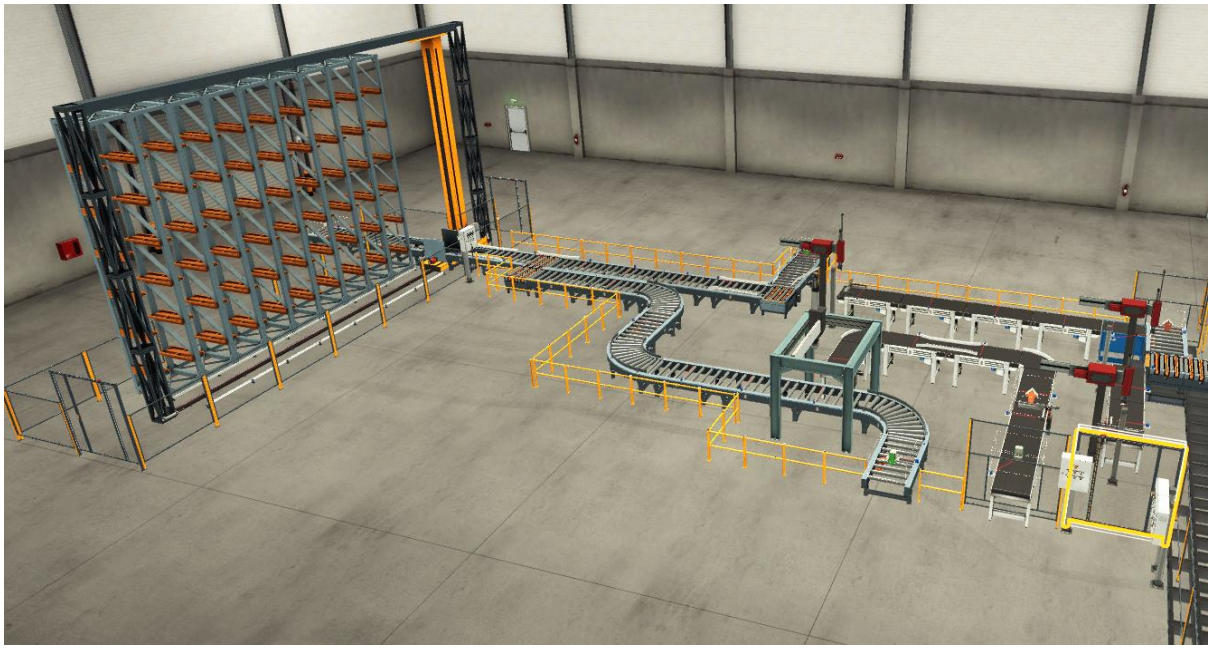


Figura 4: Etapa de post procesado y almacenamiento final

Ahora, en la Figura 5 se puede observar una vista del sistema de producción al completo:

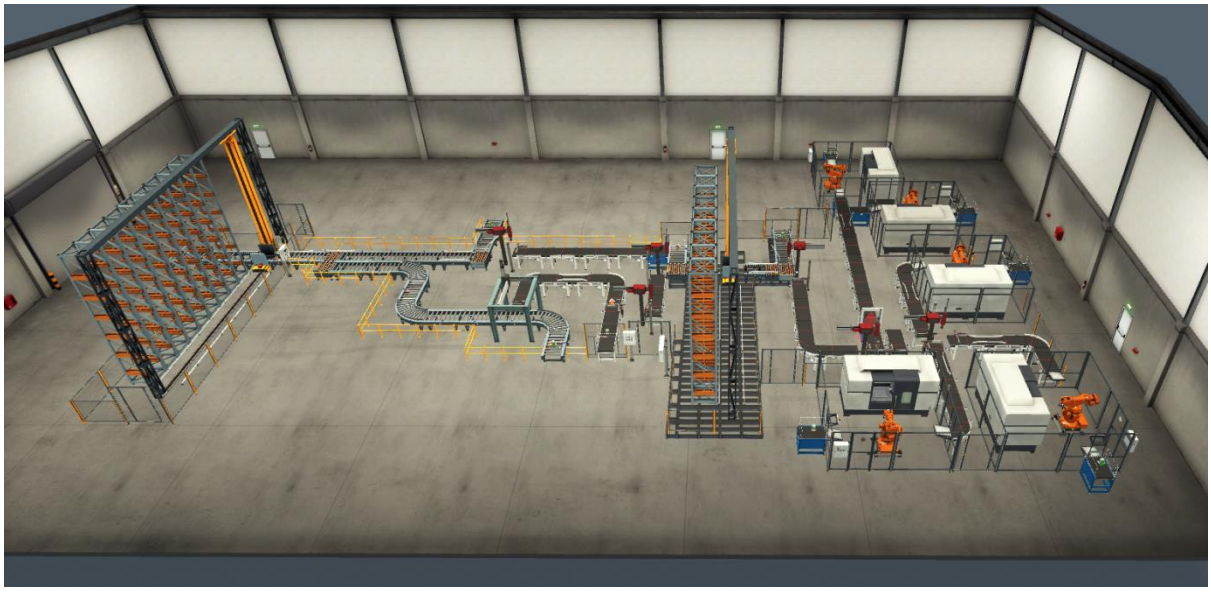


Figura 5: Cadena de producción completa

En la Figura 6 se encuentra el panel de control del sistema de producción, mediante los pulsadores de *Start*, *Stop* y la seta *Emergency* se pondrá en marcha o se parará el sistema. Los indicadores de luz muestran en qué estado se encuentra el sistema.

Figura 6: Panel principal de control del sistema de producción

En las Figuras 7 y 8 aparecen los paneles de control de los almacenes. Los pulsadores que tienen en común ambos almacenes son para evitar conflictos a la hora de qué material sacar.

El usuario elegirá qué material desea enviar a post proceso en el caso del almacén intermedio, y en el caso del almacén final, será el material que se desea sacar de fábrica (por defecto se ha elegido que sea el color "Azul"). Los indicadores de luz marcan qué material se desea sacar en todos los casos. También hay seis contadores en total, tres para cada almacén, en los que se puede llevar la cuenta del número total de piezas almacenadas.

En el caso del panel del almacén 1, mediante el uso del selector, se seleccionará el post proceso deseado, embalaje o puesta en cajas.

En relación con estos indicadores y selectores, hay que destacar también los potenciómetros en el cuadro derecho. Con estos potenciómetros se seleccionará el valor de stock máximo y mínimo que se desea establecer (por defecto a 18 y 0 respectivamente), mostrados en los visualizadores superiores. El robot del almacén pasará del estado de entrada de material al de salida cuando se supere o iguale el límite de stock máximo establecido. Este estado se prolongará hasta que se saque todo el material deseado, marcado con el límite mínimo de stock. El indicador "Output/ON_Input/OFF" se iluminará cuando el robot entre en modo salida de material del almacén y no lo hará en caso contrario.

Se presupone que el límite máximo debe ser superior al límite mínimo, por lo que, en caso de error humano y el stock mínimo sea igual o superior al máximo, si el robot llegase a entrar en el modo salida de material por alcanzar el límite superior, el stock mínimo se pondrá a 0 por defecto, hasta que el usuario introduzca un parámetro correcto. Se dispone del indicador de luz "Error_stock_parameters" que indica que se han introducido unos parámetros incoherentes. En un caso real, estos parámetros se podrían introducir mediante un sistema SCADA de forma semejante y así evitar este posible error humano no dejando introducir los parámetros en caso de error, se ha pretendido hacerlo de esta manera para hacer más interactiva la simulación.

Figura 7: Panel de control almacén intermedio

En el almacén 2, presionando el pulsador “Storage_Operation_WH2”, se dicta la orden de sacar o almacenar material según lo que se haya seleccionado previamente en el selector contiguo. En el Anexo III aparecerán todos los elementos que se han utilizado para la implementación del controlador.

Figura 8: Panel de control almacén final

Capítulo IV Modelado y análisis del sistema de control

4.1 Modelado de red

El sistema de control obtenido se ha conseguido tras modelar y analizar el sistema utilizando redes de Petri coloreadas. Partiendo de un modelado muy general, se ha creado un lugar para cada operación que se tiene que realizar (pudiéndose ejecutar varias acciones por operación). De igual manera, las transiciones irán vinculadas a funciones lógicas que permitirán su disparo, u ocurrencia, una vez evaluadas como verdaderas. Estas funciones lógicas se implementan mediante los sensores de movimiento o las variables auxiliares diseñadas

El primer paso que se debe dar, a la hora de modelar la red en base a la instalación que se tiene, es definir los colores y tipos de colores que se van a utilizar. Esta elección se basa en la búsqueda de elementos del sistema que realicen operaciones/actividades similares o que se podrían definir secuencialmente de la misma manera. Es una elección que a priori puede no ser muy obvia y que puede tener que modificarse según se vaya creando el modelo del sistema.

Para este caso, se han definido tres tipos de colores, dos de ellos de tipo *string* y el tercero tipo *integer*. El tipo llamado UNIT, es el tipo unidad, sería una marca sin color, *integer*, como en las redes de Petri lugar/transición. Por otro lado, el tipo Color de Objeto (CO), es de tipo *string*, en el que hay 3 marcados/colores diferentes: “AZUL”, “VERDE” y “GRIS”, correspondientes al color de las bases y las tapas que se fabrican (no confundir en este caso el color de las redes de Petri coloreadas, con el color de las piezas, no es lo mismo, ya que se podría definir un color que fuese la unión de azul con gris, por ejemplo, que pertenecería a un tipo de color producto de dos veces el tipo CO).

Además de estos dos tipos existe un tercero, más secundario, que se trata del tipo Máquina. Es el tipo de color que indica cuál de las dos estaciones M3 es en la que se están realizando las operaciones de carga de la materia prima y producción de bases, en este caso dentro de él se han definido dos colores, “M3_1” y “M3_2” correspondientes al número de la estación Las otras tres estaciones de fabricación se diferencian gracias al segundo tipo de color comentado. Un ejemplo de conjunto que aparece en la red se aprecia en el lugar 10 {Azul, Verde, Gris} y un multiconjunto aparecería en los almacenes teniendo el multiconjunto {18`Azul, 18`Verde, 18`Gris}. Tras la evaluación del sistema, se ha obtenido el siguiente modelo (Figura 9):

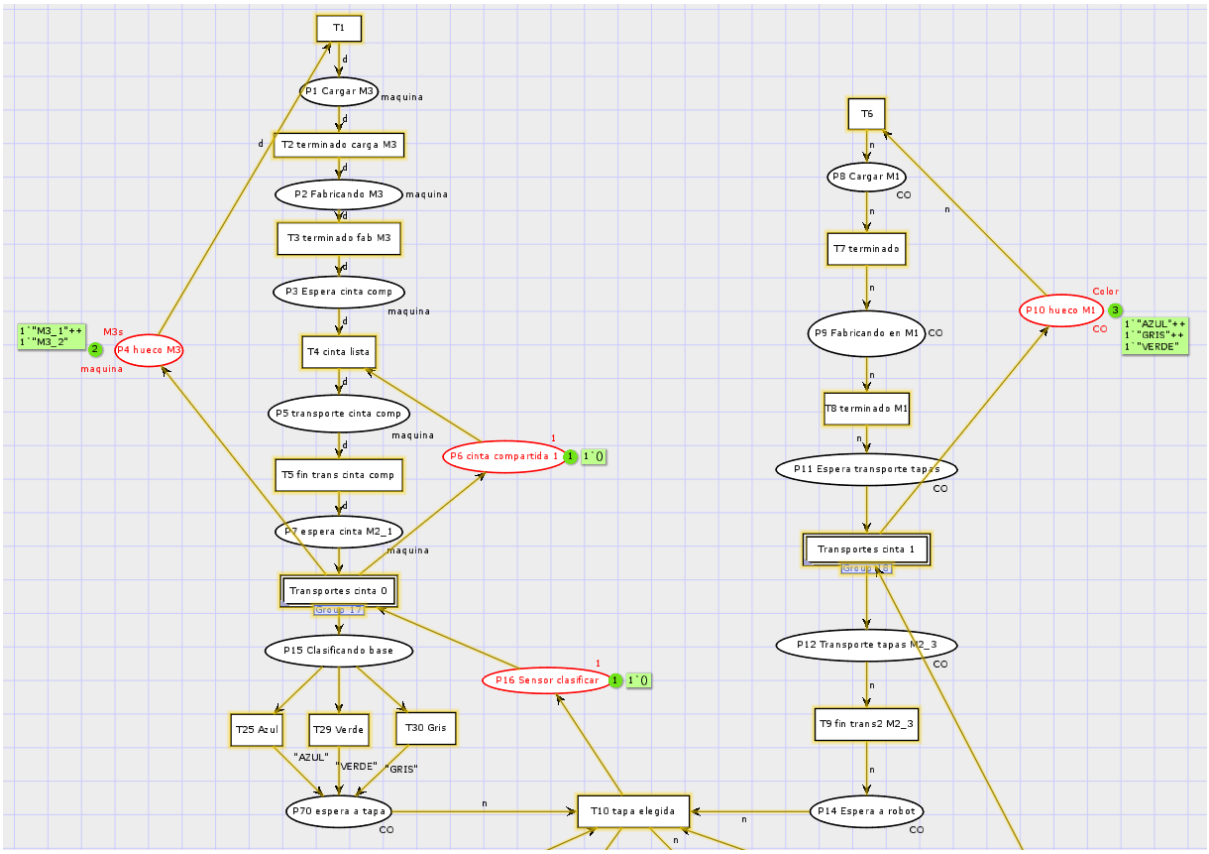


Figura 9: Parte de la Red de Petri coloreada modelando el sistema de fabricación

Debido a su tamaño, solo se ve parte de él, en el Anexo I se adjunta un repositorio que lleva a un documento con imágenes de toda la red (pinchando en la imagen se va hasta dicho anexo).

Se puede apreciar que el modelo creado del sistema empieza con la llegada de materia prima y acaba con la salida de las piezas del almacén final, las operaciones anteriores y posteriores al modelo no se tienen en consideración ya que no forman parte del alcance de este proyecto (se considera que hay materia prima infinita y la salida de fábrica se puede realizar en cualquier momento). Se pueden ver de color rojo los recursos que se utilizan dentro del sistema y en verde el marcado de los lugares.

Para que la red fuese más comprensible, se han creado subgrupos de redes o submodelos, para englobar actividades similares como los transportes de un mismo grupo de cintas o los movimientos de los robots, quedando un modelo jerarquizado (Imágenes adjuntas en el Anexo I). Estos subgrupos son las transiciones con borde doble y etiqueta, a esta metodología se le conoce por el nombre de refinamiento de la red. Los movimientos que realizan todos los robots, como la correspondencia de los grupos de cintas modelados en la red con Factory I/O se encuentran en el Anexo II.

Es importante destacar las variables y constantes que se emplean para colocar los marcados iniciales en los lugares correspondientes y en las expresiones de los arcos (Figura 10).

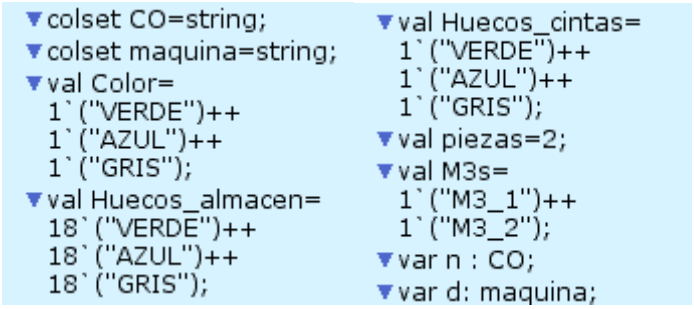


Figura 10: Parámetros empleados en la red

En el caso de las constantes, se han empleado esencialmente para poder modificar de manera más rápida y sencilla las capacidades de los distintos almacenes o para definir los colores que se van a utilizar (caso de la declaración "Color" y "M3s", que definen los cinco colores, a parte del tipo unidad, que se utilizan).

En los almacenes es necesario el uso de colores tipo CO para poder distinguir las piezas que se están almacenando, al igual que en las capacidades de algunas de las cintas (ya que según el marcado se utiliza una para cada tipo de pieza). De nuevo, esto es meramente una elección de todas las casuísticas que se pueden dar a la hora de elegir almacenar los objetos. Se podría haber elegido también, almacenar los objetos independientemente de su tipo en el almacén intermedio, pero del modo escogido, se puede limitar la cantidad de piezas que tenemos en "stock" de cada ejemplar, cosa que puede ser beneficiosa para asegurar que va a haber cierta cantidad de piezas de un modelo en el almacén.

Además de estas constantes, se ha definido una constante llamada piezas, que se utiliza para determinar qué cantidad de piezas se va a emplear en los postprocesos de embalado y puesta en caja. Si se desea otra cantidad, bastaría con cambiar este parámetro (para las cajas apilables, el máximo número que permite el simulador es de 3 piezas por cajas).

Como resultado del uso de tipos de datos especiales, es necesario definir variables para establecer los pesos de los arcos según el tipo de colores que aparezcan en el conjunto (o multiconjunto) de los lugares adheridos a dichos arcos. Se han definido "n" y "d" que son las variables que se emplean para determinar que los arcos consumen/producen marcas de tipo CO para "n" y Máquina para "d". Para el tipo de dato unidad, simplemente es necesario establecer el número de marcas que se consumen/producen. Ciertamente es de vital importancia decir que no hay tipos de colores en forma de producto, puesto que los únicos productos de marcas que aparecen son entre los tipos CO y Máquina con el tipo unidad. Al ser con el tipo unidad, no es necesario establecer un nuevo tipo de color, pero este hecho se verá reflejado más adelante en la formalidad de las redes de Petri (a la hora de definir las transiciones, matrices Pre y Post).

Otros aspectos importantes para destacar son la aparición de ciertos lugares y transiciones especiales en la red, que no son quizá tan claros de vincular con actividades del sistema meramente viendo la red. Empezando por arriba y descendiendo, existen tres transiciones que no se han podido agrupar (Figura 11), se trata de las tres transiciones en conflicto que modelan la clasificación de las bases según el color detectado.

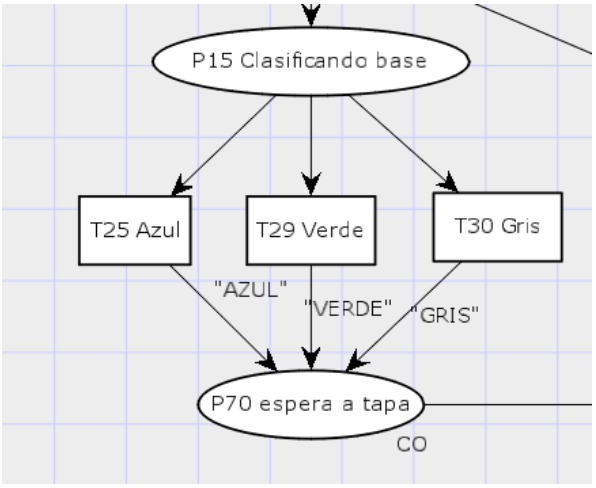


Figura 11: Modelado de la clasificación de las bases

Estas tres transiciones se disparan según el color de la pieza base detectado. Al realizarse un cambio de tipo de color, es necesario separarlas para que quede claro que existe un conflicto entre ellas y que cada una realiza un cambio de color distinto.

Otro de los lugares que se debe tener en cuenta (Figura 12), es el lugar que se encuentra tras el primer almacenamiento. Este lugar se utiliza para asegurarse de que las piezas que salgan del primer almacén sean del mismo color. El motivo por el que se ha determinado así es por el hecho de desear embalar y poner en cajas piezas del mismo color, si se produjese el caso de que salieran piezas de diferente color del almacén, el sistema se bloquearía (verificado con la parte de análisis). Para cambiar la casuística del sistema, será necesario cambiar los pesos de los arcos correspondientes a las entradas de esta transición, así como en caso de desear introducir modificaciones en la cantidad de piezas para el post procesado como se ha comentado previamente (para Factory I/O el máximo es de tres, pero para un sistema real podría ser mayor). También sería necesario utilizar lugares con un tipo de colores producto de dos colores tipo CO, ya que el conjunto que apareciese en el lugar sería un tipo de color producto de dos colores (marcas) de tipo CO.

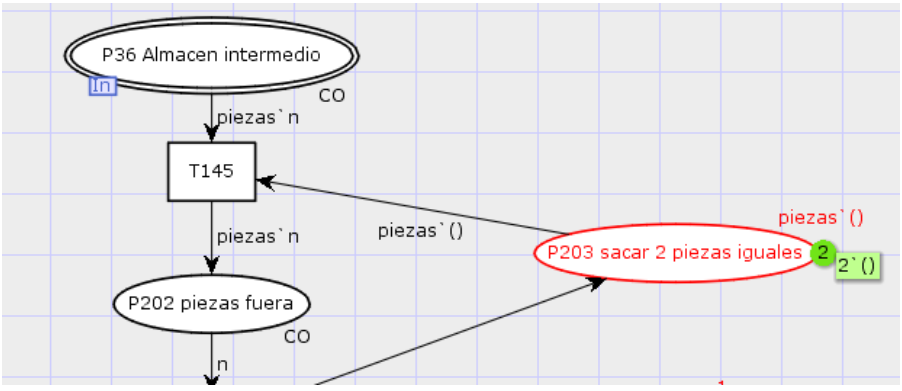


Figura 12: Modelado de la salida del almacén intermedio

Esta elección de realizar el postproceso a dos piezas del mismo tipo, también se puede ver modelado en las transiciones de la Figura 13:

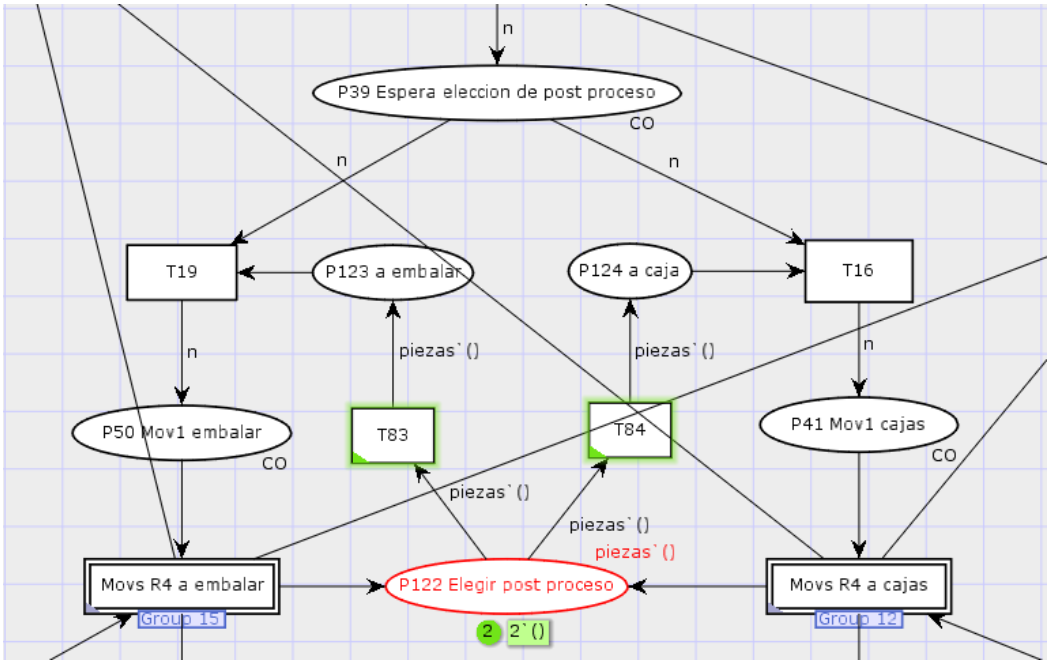


Figura 13: Modelado de la elección de post proceso

Es el usuario el que elige cómo quiere postprocesar las piezas, la elección se ha modelado de esta forma para evitar bloqueos en la red. En caso de que el usuario decidiese cambiar de postproceso en mitad del envío de las dos piezas, el sistema no responderá hasta que ambas piezas se hayan enviado al postproceso que estaba seleccionado cuando se ha enviado la primera de ellas. De ahí el lugar P122, este lugar no vuelve a tener dos marcas hasta que el post proceso escogido haya finalizado para la cantidad total de piezas deseada, bloqueando así la posibilidad de que se dispare la transición perteneciente al post proceso contrario durante dos marcados.

4.2 Análisis de la red

Hasta ahora se ha hablado solamente de la parte de modelado del sistema. Es importante saber que el diseño de la red es un proceso iterativo que se mejora gracias a los análisis que se aplican cada vez que se termina una iteración de modelado. El propósito de estos análisis es la verificación del cumplimiento de propiedades y especificaciones que se desean que cumpla o que debe cumplir el sistema. Algunas se han introducido ya en apartados anteriores, pero en estos próximos párrafos se expondrán más en detalle todas ellas. Lo primero que hay que destacar antes de entrar en materia, es el papel tan importante que realizan en este apartado las herramientas como CPN tools, en el caso que ocupa de las redes coloreadas, u otras como podría ser la herramienta Petri net Toolbox de MATLAB para las redes lugar/transición. Sin estas herramientas, el trabajo iterativo hasta alcanzar un modelo matemáticamente correcto, y consistente, sería un proceso largo y farragoso. Es por este motivo por el cual existen estas herramientas para automatizar y ayudar a realizar el análisis del modelo.

Dentro de las propiedades básicas de las redes de Petri se pueden separar en dos grandes grupos, las propiedades relacionadas con el comportamiento de la red (se tiene en consideración de dónde parte el sistema)[9].

Existen muchas propiedades dentro de cada uno de estos conjuntos, a continuación, se enumeran las propiedades básicas de comportamiento que se deben analizar en una red de Petri y su importancia con respecto a los sistemas de fabricación:

- Limitación: la red de Petri debe tener un número finito de estados, es decir, el marcado de los lugares de la red tiene que ser limitado y de ahí que el número de estados esté acotado. En caso de no serlo, las marcas tenderían al infinito y la red se bloquearía cuando llegase a este número por no poder representarse (un caso particular de la limitación es la propiedad conocida como seguridad, esto ocurre cuando el marcado de los lugares solo puede tomar valores de 1 o 0 marcas). Para los sistemas de producción es una propiedad muy importante ya que se estaría dando por sentado que se tienen recursos o capacidades de las operaciones infinitas, cosa que no es verdad ya que estos recursos y capacidades son limitados.
- Exclusión mutua: se dice que dos lugares están en exclusión mutua si para un M_o, estos lugares no pueden ser marcados simultáneamente en ningún marcado alcanzable desde M_o. Este es el caso del uso de un recurso compartido, el conjunto de lugares que los utilizan no puede superar el número de marcas que haya en ellos, por ejemplo (Figura 14):

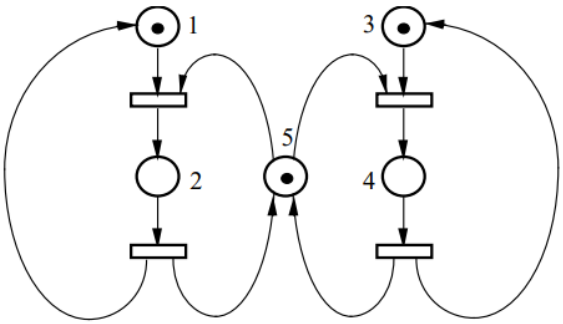


Figura 14: Ejemplo de exclusión mutua, Fuente [8]

El sumatorio de marcas de los lugares 2, 4 y 5, no puede superar en ningún caso la cantidad de recursos compartidos, en este caso uno, m(p2) + m(p4) + m(p5) = 1, no puede aparecer marcas simultáneamente en los lugares 2, 4 y 5. Estos lugares estarían en exclusión mutua y las transiciones superiores estarían en conflicto efectivo, al no poder disparar simultáneamente ni producir un marcado que pueda disparar la otra transición. Esta propiedad en los sistemas de producción implica operaciones que no se pueden ejecutar al mismo tiempo, es de vital importancia, puesto que si no se cumpliese esta propiedad se podrían realizar operaciones que llevarían a daños en el sistema o bloqueos (como el caso que se comentaba en capítulos anteriores de las operaciones de rotación de un motor, no se le puede pedir que gire en ambos sentidos a la vez).

- Bloqueos: el sistema no puede entrar en una situación en la que ningún evento puede ocurrir. Estas situaciones son las que se deben evitar a toda costa para que el sistema no colapse. En sistemas de fabricación implicaría una parada de la producción de la que no saldría sin ejecutar una acción externa.
- Vivacidad: generalizando la propiedad anterior, la vivacidad se trata de la posibilidad de ocurrencia de cualquier evento de la red. Se dirá que una red es NO viva, si existe alguna de las transiciones que deja de poder dispararse, produciendo así un bloqueo en la red.
- Reversibilidad: para los sistemas de producción o cíclicos, se desea que, en algún momento, o con una secuencia de eventos, desde cualquier estado alcanzable se pueda volver al estado inicial de partida para así poder iniciar un nuevo ciclo.

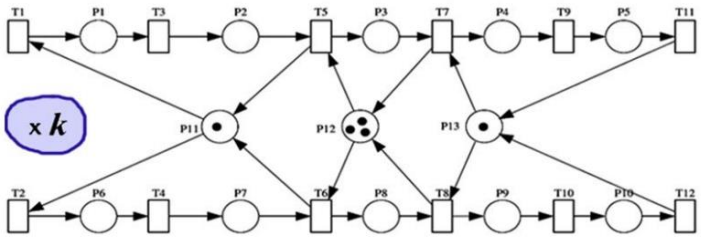
Para estudiar todas estas propiedades existen muchos procedimientos de análisis. En este trabajo se expondrán solamente todas aquellas que se han utilizado para verificar que se cumplen las

propiedades del sistema según se ha modelado, por tanto, es más una verificación de que el modelado se ha realizado correctamente.

La primera herramienta que se utilizó para determinar las propiedades de bloqueo, vivacidad, reversibilidad y limitación del sistema, es el grafo de alcanzabilidad, donde se obtienen todos los estados posibles del sistema. La ventaja que tiene esta metodología de análisis es que solo se puede utilizar para aquellos sistemas que sean limitados (si fuese ilimitado algún lugar, los estados alcanzables serían infinitos y no se podría obtener el grafo), esto es una ventaja ya que si no se puede obtener el grafo de alcanzabilidad se puede dar por falsa la propiedad de limitación del sistema.

Viendo el grafo de alcanzabilidad, un bloqueo viene dado por un estado en el que el grafo acaba, del cual no se puede ir a ningún otro estado. Obtenidos estos estados de bloqueo, habrá que introducir el marcado en la red de dicho estado y ver por qué el sistema se bloquea. Reajustando el modelo y volviendo a crear el grafo iterativamente se puede conseguir que el sistema por fin no se bloquee. Dentro del grafo de alcanzabilidad también se pueden ver visualmente las propiedades de vivacidad y reversibilidad a no ser que se produzca lo siguiente.

Uno de los problemas que plantea esta metodología es que se puede dar una explosión de estados en el caso de que se posean muchos marcados posibles. El grafo que se obtiene es muy grande dificultando el buen uso de este análisis y al tener tantos estados, percibir la vivacidad de la red y la reversibilidad es una tarea complicada por no decir imposible. Este problema de explosión de estados es en lo que se basa la complejidad de los sistemas de fabricación grandes. Se necesita una capacidad computacional y de tiempo muy grande para obtener dicho grafo, se habla de obtener una cantidad de estados del orden de millón o incluso billones de estados alcanzables en la red. Por poner un ejemplo, en la Figura 15 se observa una pequeña red de Petri lugar/transición que si repitiese en diez ocasiones (diez piezas realizasen estas operaciones) generaría un número de estados alcanzables del orden de 500 millones de estados:



k	Reachable markings
1	250
2	6.300
3	67.375
5	2.159.136
10	496.992.496
50	???

Figura 15: Problema de la explosión de estados para el grafo de alcanzabilidad [8]

Queda claro que el número de estados alcanzables para la red modelada no se puede estimar fácilmente, pero la probabilidad de que se supere la cifra comentada es muy alta.

A pesar de que el grafo sea grande y se haya podido obtener, la herramienta CPN tools soluciona este problema visual para comprobar las propiedades generando un informe de los estados de la red en los que aparecen bloqueos, las transiciones que no sean vivas, y la reversibilidad del sistema. Aunque CPN tools facilita mucho la obtención del grafo de alcanzabilidad y la información que recoge, otro

problema que presenta el análisis es que es muy sensible a cambios en la red, si se cambia el marcado, por ejemplo, hay que recalcular de nuevo el grafo de alcanzabilidad. Este proceso como ya se ha dicho es muy lento si la red es muy grande, por este motivo, existen otras técnicas que aumentan la velocidad de análisis y que pueden utilizarse previamente al uso de esta metodología.

El segundo método aplicado, es conocido por el nombre de reglas de reducción. Se basa en una serie de reglas que permiten aplicar transformaciones en la red para reducir su tamaño, y así facilitar su análisis (por ejemplo, combinándolo con el procedimiento anterior) las reglas empleadas son las siguientes:

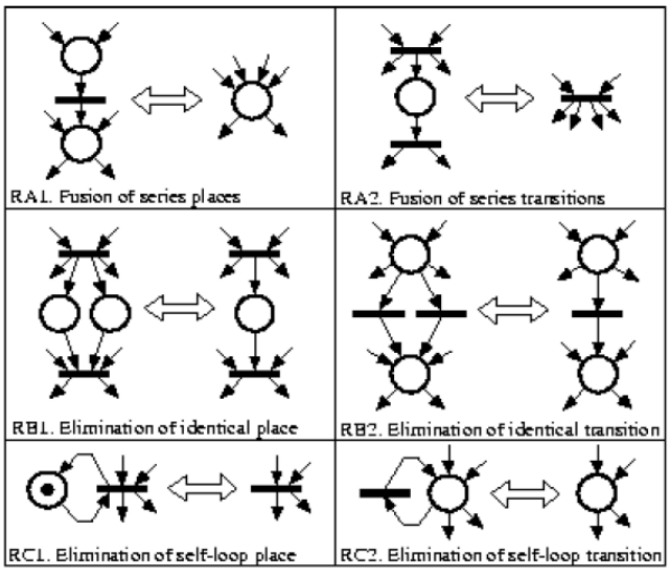


Figura 16: Conjunto de reglas básicas para la reducción de sistemas, Fuente: [8]

Este método, tiene la ventaja de que, aunque se vayan eliminando o fusionando lugares y transiciones, las propiedades de vivacidad y limitación se conservan si se siguen las reglas vistas en la Figura 16. También se conserva la reversibilidad, excepto si se aplica la regla RA1, que en ese caso esta propiedad deja de conservarse. Por consiguiente, se han aplicado todas estas reglas menos la primera para obtener una nueva red con la que poder obtener el grafo de alcanzabilidad y ver si se cumplen las propiedades.

Para verificar las primeras iteraciones del modelado de la red, se han utilizado también métodos de simulación. Simulando el sistema para un número grande de ocurrencias de eventos (se han realizado simulaciones para una ocurrencia de mil millones de eventos), se ha ido observando si aparecía algún bloqueo en la red por un modelado incorrecto.

Finalmente, tras comprobar que para un número grande de ocurrencias el sistema no se bloqueaba, se han utilizado las técnicas mencionadas para comprobar que efectivamente las propiedades del sistema se cumplían. Hecha la reducción y obtenido el grafo de alcanzabilidad, con 59400 estados, la red que se ha obtenido es totalmente viva, en ningún momento se anula la posibilidad de que vuelva a ocurrir algún evento de la red. No posee ningún marcado infinito al poder obtenerse sin ningún problema el grafo de alcanzabilidad, dato muy importante en los sistemas de automatización para que no se produzca un desbordamiento de la cadena como se venía comentando. Existe exclusión mutua en todos los recursos compartidos, cumpliéndose debidamente que el conjunto total de marcas no aumenta, ni disminuye, de la suma total de marcas de los lugares que pertenecen a la exclusión. También se obtiene otra propiedad no comentada que es el marcado local (*home marking*), si en este marcado local están

todos los marcados posibles (como en este caso), quiere decirse que todos los marcados de la red son alcanzables desde otros marcados que así lo sean definiendo, así como reversible este sistema.

Capítulo V Implementación del sistema de control

En las primeras etapas de este trabajo fin de máster, se realizó una búsqueda de distintos métodos con los que se pudiera realizar un puente entre el simulador Factory I/O y CPN tools. El simulador Factory I/O, dentro del abanico de posibilidades de comunicación, puede trabajar con el protocolo Modbus TCP/IP mientras que CPN tools utiliza comunicación TCP/IP para enviar/recibir información de otros soportes como, en este trabajo, sería el simulador Factory I/O. Lamentablemente, tras varias conversaciones con el equipo de soporte técnico de CPN tools se vio que, a día de hoy, esta comunicación aún no es viable, por lo que quedará a expensas de que el equipo técnico de CPN tools desarrolle esta posibilidad de comunicación (ver Capítulo VII). Por ello, se han utilizado otras herramientas para implementar el controlador obtenido en CPN tools y así verificar su diseño, en simulaciones del sistema de producción en Factory I/O.

El primer punto fue encontrar una librería de protocolo Modbus para poder realizar la comunicación en Factory I/O. Existen muchas librerías que permiten la implementación de comunicación Modbus, al igual que lenguajes de programación que las implementen. Se valoraron diferentes posibilidades como el uso de librerías como EasyModbus, Nmodbus, o libmodbus, pero finalmente se optó por utilizar la librería Modbus que utiliza MATLAB. El motivo fue que las primeras librerías se utilizan en lenguajes de programación como C y C++, estos lenguajes tienen un problema en común, y es que no tienen la flexibilidad y facilidad que ofrece MATLAB para el manejo de matrices. Es por este motivo que, al tener que utilizar matrices de gran tamaño, se ha preferido el uso de MATLAB frente a otros lenguajes de programación.

El protocolo Modbus es un protocolo que funciona con la modalidad maestro-esclavo o cliente-servidor, donde el maestro/cliente es el que controla en todo momento las comunicaciones con los esclavos/servidores, y son estos quienes se limitan a devolver los datos solicitados o ejecutar las operaciones indicadas por los clientes. En este caso, MATLAB se ejecutará como cliente y el simulador Factory I/O será el servidor.

Dentro del protocolo, existen diferentes variantes de la comunicación, como es el serial RTU o el Ethernet TCP/IP. La diferencia entre estas variantes es, básicamente, que el protocolo Modbus RTU utiliza un protocolo nivel serial y el protocolo TCP/IP utiliza una capa física de Ethernet (también el TCP/IP utiliza 6 bytes de encabezado para permitir el enrutamiento). El mensaje está estructurado de la siguiente manera (Figura 17):

Figura 17: Esquema de envío de dato Modbus TCP/IP, Fuente [11]

Para este trabajo, Factory I/O utiliza el protocolo Modbus TCP/IP, por lo que la trama enviada de datos sigue la siguiente estructura (Figura 18):

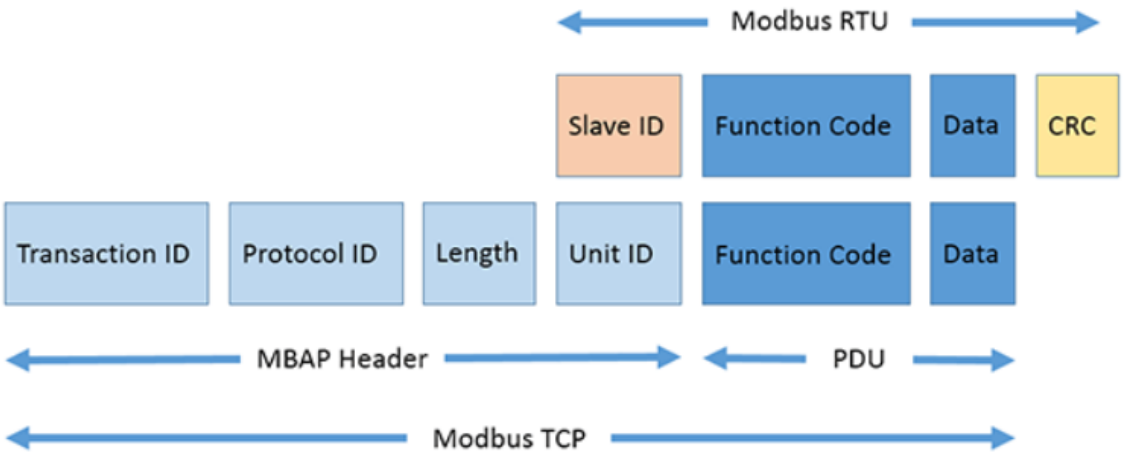


Figura 18: Comparativa de estructura de mensaje Modbus TCP/IP vs RTU, Fuente: [11]

De este encapsulado es importante destacar el código de la función. Este código dependerá de la orden que se quiera ejecutar, lectura o escritura, así como diagnósticos y otras funciones. Para la lectura y la escritura se tendrá que considerar el tipo de dato que se desea enviar o recibir, si es digital o analógico. Estos datos también se mapean de forma distinta en memoria según el tipo que sean, para las salidas digitales (*coils output*), se mapean en el rango de memoria 1-10000, las entradas digitales (*discrete inputs*) en el rango 10001-20000, las entradas analógicas (*input registers*) en el rango 30001-40000 y las salidas analógicas (*holding registers*) del 40001-50000. La trama que se presenta a continuación es un pequeño ejemplo de cómo sería la comunicación entre el maestro y el esclavo (Figura 19):

P:[0A][04][00][00][00][0A][71][76]
0A numero de periférico
04 función de lectura
00 00 registro donde se va a comenzar la lectura
00 0A numero de registros a leer: 10
71 76 CRC

R:[0A][04][14][00][00][08][4D][00][00][23][28][00][00][0F][A0][00][00][00][90][00][00][00][60][CB][2E]
0A Número del periférico que responde, 10 en decimal
04 Función de lectura - la que se ha utilizado en la pregunta
14 Número de bytes recibidos (20).
00 00 08 4D V1x 10 (registro 00 Hex) con valor en decimal 212,5 V
00 00 23 28 mA 1, en decimal 9000 mA
00 00 0F A0 W 1, en decimal 4000 W
00 00 00 90 varL 1, en decimal 144 varL
00 00 00 60 PF1 x 100, en decimal 96
CB 2E CRC

Figura 19: Ejemplo de trama Modbus TCP/IP

A pesar de que en el ejemplo aparezca el CRC (*Cyclic Redundancy check*), y no lo haga en la Figura 18, se debe a que el protocolo Modbus TCP/IP se considera fiable porque ya se realizan comprobaciones de errores, como parte de la capa física Ethernet TCP/IP. Depende mucho de la aplicación el que se envíe o no, pero a veces, aunque no hiciese falta y sea redundante, es menos costoso enviarlo que no hacerlo.

Tratado el protocolo Modbus, el primer paso dado, una vez que ya se tenía claro el soporte técnico que se iba a usar, ha sido comprobar la conexión entre Factory I/O y MATLAB. Para ello, MATLAB

posee una herramienta, Modbus Explorer, con la que tomar y enviar datos vía Modbus a un servidor, en este caso Factory I/O. En la Figura 20 se ve el entorno de la herramienta con una prueba realizada:

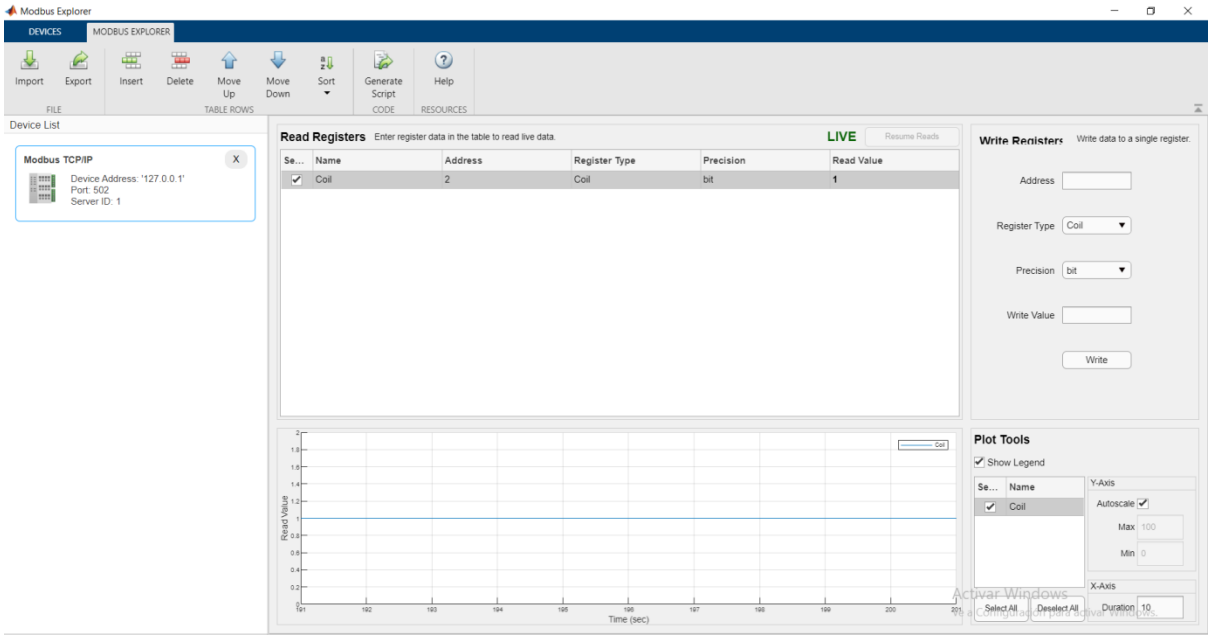


Figura 20: Herramienta Modbus Explorer MATLAB, Fuente: MATLAB [12]

A partir de la comprobación realizada, se pasa a traducir la formalidad de las redes de Petri a código en MATLAB.

El primer paso entonces es crear la conexión entre MATLAB y Factory I/O. En esta parte del código se genera la conexión Modbus usando Modbus TCP/IP. Lo primero que se debe hacer es crear el servidor Modbus que se va a utilizar. Los parámetros usados son los que aparecen a continuación, deben de estar acorde con el servidor Modbus que se utiliza, en este caso, el que se crea en Factory I/O. De desear utilizar otros parámetros, también sería necesario realizar los cambios adecuados en el servidor de Factory I/O.

Creado el vínculo entre MATLAB y Factory I/O, lo siguiente es entrar en la parte principal del código desarrollado. Se trata de un bucle *while* del cual no se saldrá mientras el estado del sistema sea distinto de stop. Dentro de este bucle, se pasa a una máquina de estados en la que se pueden distinguir principalmente tres estados: reposo, funcionamiento y emergencia. El primero se dedica al comienzo de la producción, no se empezará a producir hasta que se haya pulsado el pulsador "Start". El segundo estado envuelve el funcionamiento en sí de la producción, de este estado no se saldrá hasta que la producción se detenga ya sea por un fallo en el sistema, como por una parada deseada del sistema de producción.

Entonces, este bucle de producción se ejecutará infinitamente, hasta que se salga de él mediante los pulsadores de parada o emergencia. Cuando se finalice el bucle por una parada deseada, automáticamente se desconectará MATLAB del servidor Modbus.

Dentro del bucle se van a llevar a cabo ciertas operaciones siempre en el siguiente orden:

- Lectura del estado de las entradas de Factory I/O (sensores)
- Cálculo de variables auxiliares

- Comprobación de transiciones sensibilizadas
- Para las transiciones sensibilizadas, comprobación de las condiciones externas que las limitan
- Asignación de todas las acciones a los lugares que obtengan un nuevo marcado con el disparo de dichas transiciones (activar o apagar los actuadores)
- Cálculo del nuevo marcado según el disparo de las transiciones
- Envío de las acciones (actuadores) a Factory I/O.

5.1 Lectura del estado de las entradas y envío del estado de las salidas

La lectura y escritura de datos se realiza con las funciones "*write"* y *"read"* que posee la librería Modbus de MATLAB [13]. Estas funciones utilizan el objeto servidor creado por MATLAB, para enviar/recibir los datos desde el servidor. Hay que tener en cuenta también que las entradas y salidas pueden ser de tipo booleanas o de tipo real o entero (digitales o analógicas), en el caso de ser booleanas se trabajará con "*coils*" y en el caso de tratarse de enteros o reales serán de tipo registro. Es muy importante considerar este aspecto, pues no solo hay que enviar tipos de datos distintos, sino que también las memorias y direcciones de memoria son distintas para cada caso. Gracias a las funciones creadas en la librería, simplemente con poner el tipo de dato que se quiere escribir/leer y dirección en la que escribirlo/leerlo, ya se envía la trama necesaria para llamar a la función de escritura/lectura y a la posición de memoria.

Teniendo los datos de las variables almacenados, ya se puede proceder a la formalidad de las redes de Petri para la ejecución del sistema de control.

5.2 Formalidad de las Redes de Petri

La formalidad de las redes de Petri se define partiendo de la aproximación de las ecuaciones de estado. Para las redes de Petri el vector de estados será el marcado (m(k)) y las entradas y salidas los lugares y transiciones.

$$m(k+1) = m(k) + C(t) \text{ (ec.1)}$$

Donde C(t) es la matriz de incidencia, esta matriz define los cambios que se producen en los lugares, estados, según el evento/transición que haya ocurrido. Para que un evento pueda ocurrir se tiene que cumplir la siguiente condición:

$$m(k) + C(t) \geq 0 \text{ (ec.2)}$$

Esto implica que no puede ocurrir un evento si no existen marcas suficientes en los lugares de entrada. Esta matriz C(t) se calcula como la resta entre las matrices Post y Pre:

$$C(t) = Post(t) - Pre(t) \text{ (ec.3)}$$

Quedando las ecuaciones 1 y 2:

$$m(k) - Pre(t) \geq 0 \text{ (ec.4)}$$

$$m(k+1) = m(k) + Post(t) - Pre(t) \text{ (ec.5)}$$

Las matrices Pre y Post son las matrices que establecen las relaciones entre N lugares y M transiciones de dimensiones [NxM]. Cada elemento de la matriz es el peso del arco que une la transición$_j$ con el lugar$_i$, en el caso de la matriz Pre son todos los arcos de entrada para cada transición, y en el caso de la matriz Post son los arcos de salida.

En el caso de las CPN, las dimensiones de estas matrices aumentan. Ahora las filas de la matriz serán los lugares multiplicados por el número de colores que haya en la red (número de lugares multiplicados por la dimensión del vector B, explicado en el capítulo II), mientras que las columnas, serán las transiciones multiplicadas por todas las combinaciones de colores posibles que se den en todas las transiciones de la red. El vector de estados (marcado) será un vector columna que tendrá por filas el número de lugares por el número de colores, donde los elementos de la matriz será el número de marcas de dicho color que posea el lugar. Se añade a continuación un pequeño ejemplo de parte de la red coloreada con sus matrices y vectores creados (Figura 21):

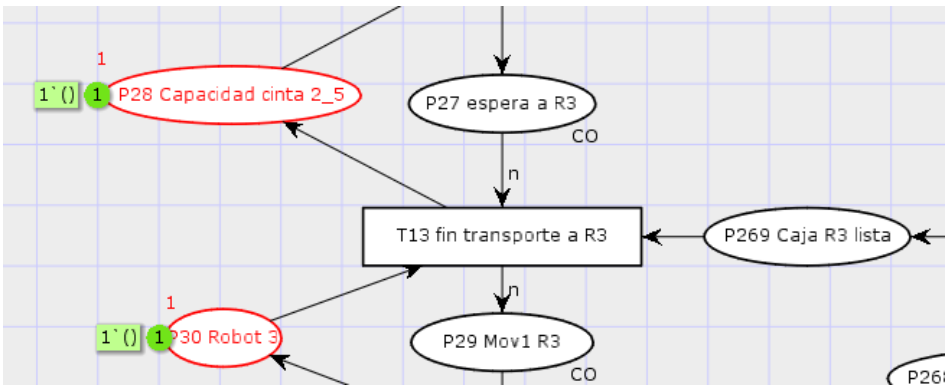


Figura 21: Ejemplo para la creación de las matrices Pre y Post

Para esta parte de la red, en el caso de la transición trece sus matrices Pre, Post, incidencia y marcado quedarán así (Figura 22-Figura 25):

$$[T13\_U \quad T13\_M31 \quad T13\_M32 \quad T13\_A \quad T13\_V \quad T13\_G \quad T13\_UxM31 \quad T13\_UxM31 \quad T13\_UxA \quad T13\_UxV \quad T13\_UxG]$$

$$\begin{pmatrix} P27\_U \\ P27\_M31 \\ P27\_M32 \\ P27\_A \\ P27\_V \\ P27\_G \\ P30\_U \\ P30\_M31 \\ P30\_M32 \\ P30\_A \\ P30\_V \\ P30\_G \\ P269\_U \\ P269\_M31 \\ P269\_M32 \\ P269\_A \\ P269\_V \\ P269\_G \end{pmatrix} = \begin{pmatrix} & & & & & & & & & & \\ & & & & & & & & & & \\ & & & & & & & & & & \\ & & & & & & & & 1 & & \\ & & & & & & & & & 1 & \\ & & & & & & & & & & 1 \\ & & & & & & & & 1 & 1 & 1 \\ & & & & & & & & & & \\ & & & & & & & & & & \\ & & & & & & & & & & \\ & & & & & & & & & & \\ & & & & & & & & & & \\ & & & & & & & & 1 & 1 & 1 \\ & & & & & & & & & & \\ & & & & & & & & & & \\ & & & & & & & & & & \\ & & & & & & & & & & \\ & & & & & & & & & & \end{pmatrix}$$

Figura 22: Matriz Pre

$$[T13\_U \quad T13\_M31 \quad T13\_M32 \quad T13\_A \quad T13\_V \quad T13\_G \quad T13\_UxM31 \quad T13\_UxM31 \quad T13\_UxA \quad T13\_UxV \quad T13\_UxG]$$

$$\begin{pmatrix} P28_U \\ P28_M31 \\ P28_M32 \\ P28_A \\ P28_V \\ P28_G \\ P29_U \\ P29_M31 \\ P29_M32 \\ P29_A \\ P29_V \\ P29_G \end{pmatrix} = \begin{pmatrix}
\square & \square & \square & \square & \square & \square & \square & \square & 1 & 1 & 1 \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & 1 & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & 1 & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & 1
\end{pmatrix}$$

Figura 23: Matriz Post

Y por tanto su matriz de incidencia:

$$[T13\_U \quad T13\_M31 \quad T13\_M32 \quad T13\_A \quad T13\_V \quad T13\_G \quad T13\_UxM31 \quad T13\_UxM31 \quad T13\_UxA \quad T13\_UxV \quad T13\_UxG]$$

$$\begin{pmatrix} P27_U \\ P27_M31 \\ P27_M32 \\ P27_A \\ P27_V \\ P27_G \\ P28_U \\ P28_M31 \\ P28_M32 \\ P28_A \\ P28_V \\ P28_G \\ P29_U \\ P29_M31 \\ P29_M32 \\ P29_A \\ P29_V \\ P29_G \\ P30_U \\ P30_M31 \\ P30_M32 \\ P30_A \\ P30_V \\ P30_G \\ P269_U \\ P269_M31 \\ P269_M32 \\ P269_A \\ P269_V \\ P269_G \end{pmatrix} = \begin{pmatrix}
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & -1 & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & -1 & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & -1 \\
\square & \square & \square & \square & \square & \square & \square & \square & 1 & 1 & 1 \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & 1 & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & 1 & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & 1 \\
\square & \square & \square & \square & \square & \square & \square & \square & -1 & -1 & -1 \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & -1 & -1 & -1 \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square \\
\square & \square & \square & \square & \square & \square & \square & \square & \square & \square & \square
\end{pmatrix}$$

Figura 24: Matriz de incidencia

El marcado inicial será:

$$\begin{pmatrix} P27\_U \\ P27\_M31 \\ P27\_M32 \\ P27\_A \\ P27\_V \\ P27\_G \\ P28\_U \\ P28\_M31 \\ P28\_M32 \\ P28\_A \\ P28\_V \\ P28\_G \\ P29\_U \\ P29\_M31 \\ P29\_M32 \\ P29\_A \\ P29\_V \\ P29\_G \\ P30\_U \\ P30\_M31 \\ P30\_M32 \\ P30\_A \\ P30\_V \\ P30\_G \\ P269\_U \\ P269\_M31 \\ P269\_M32 \\ P269\_A \\ P269\_V \\ P269\_G \end{pmatrix} = \begin{pmatrix} \square \\ \square \\ \square \\ \square \\ \square \\ \square \\ 1 \\ \square \\ \square \\ \square \\ \square \\ \square \\ \square \\ \square \\ \square \\ \square \\ \square \\ \square \\ 1 \\ \square \\ \square \\ \square \\ \square \\ \square \\ \square \\ \square \\ \square \\ \square \\ \square \end{pmatrix}$$

Figura 25: Vector marcado

El número de combinaciones posibles entre marcas de colores diferentes que hay en toda la red es de once, todos los colores individuales (seis) más el producto de estos por el tipo unidad (cinco). En la red modelada hay un total de 280 lugares y 224 transiciones. Realizando cálculos, esto da unas dimensiones de matrices Pre y Post de 1680 filas por 2464 columnas, con la mayoría de los elementos igual a cero. Como consecuencia de este hecho y para ahorrar tiempo de computación, se trabaja y almacena solamente las matrices y vectores dispersos.

Para la creación de las matrices Pre, Post y el vector inicial se han creado archivos Excel rellenados de tal forma que se vean las relaciones entre los lugares (filas) y las transiciones (columnas). Se han rellenado los elementos según el tipo de relación que guarden, si en la transición se realizan productos entre unidad y tipo CO aparecerá casillas azules si las marcas que se producen o consumen son de tipo unidad, si las marcas son de tipo CO serán de color naranja. En caso de ser un producto entre tipo unidad y tipo máquina aparecerán en un marrón pálido. Cuando las marcas producidas de tipo CO sean mayores a 1, aparecerán de color morado.

Si no hay producto de tipos de colores (entradas y salidas a la transición sean del mismo tipo de color) las casillas parecerán en verde para los de tipo unidad, en rosa para las de tipo CO y en marrón para las de tipo máquina.

Debido a su tamaño, en la Figura 26 solo se muestra una pequeña parte de una de ellas. Se adjunta un enlace a una carpeta drive donde se encuentran los ficheros completos:

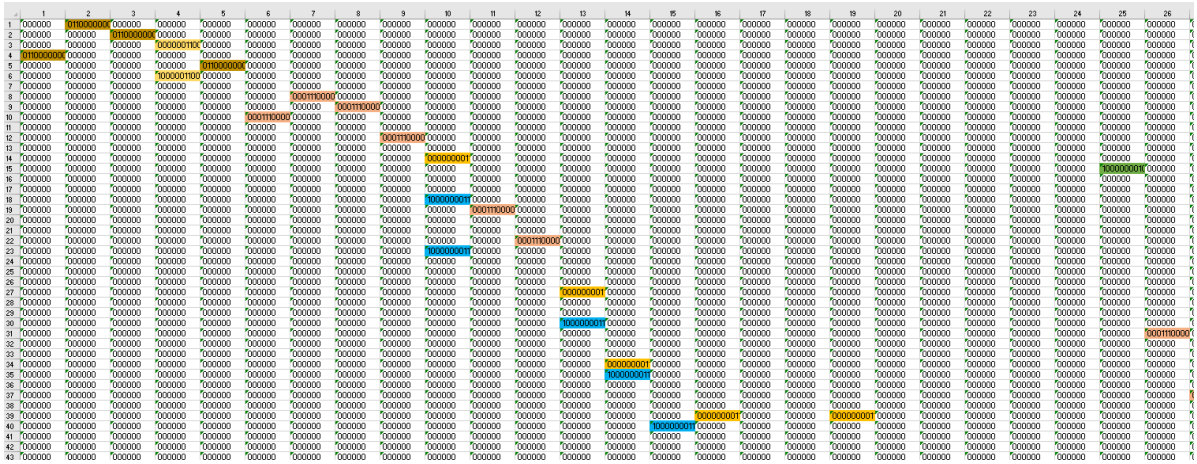


Figura 26: Imagen reducida de la creación de la matriz Pre

https://drive.google.com/drive/folders/1f7VrAf1cHj_KxU5qFFwy8Up2yGWNGKMg?usp=sharing

Gracias a un pequeño script, cada casilla se transforma a una matriz [6x11] para las matrices, y [6x1] para el vector marcado, según la explicación detallada anteriormente (se adjunta dicho script en el Anexo IV, "Crear_matrices_PRE_POST_marcado").

5.3 Comprobación de transiciones sensibilizadas

Definidas ya las matrices, la ecuación 4 modela la función "Ver_transiciones_sensibilizadas". En esta función se comprobará qué transiciones están sensibilizadas, cuáles son los eventos que según el valor de los lugares predecesores podrían llegar a ocurrir. Sin embargo, esta comprobación por sí sola es incompleta. El sistema trabaja en tiempo continuo, aunque se haya modelado como un suceso de eventos discretos. A pesar de que los lugares previos estén con marcas suficientes para poder ejecutar las siguientes transiciones, es necesario comprobar que las acciones que se estaban ejecutando previamente han finalizado o, que se está en las condiciones apropiadas para que ocurra el siguiente evento.

5.4 Comprobación de las condiciones externas

Por este motivo, teniendo ya las transiciones que están sensibilizadas guardadas en el vector "trans_sens", se ha de realizar la comprobación de las condiciones externas, función "Ver_condiciones_externas". Se ha realizado mediante la lectura de posiciones del vector "trans_sens" que indica cuáles son las transiciones que se desean comprobar (ya que son las únicas sensibilizadas para ese instante) con la ayuda de una máquina de estado. Donde cada estado será una transición o un conjunto de ellas que tengan las mismas condiciones externas. Es quizá la parte más extensa y farragosa del código al tener tantas transiciones, por lo que se adjunta en el Anexo IV más en detalle. En caso de cumplir con las condiciones externas, se han almacenado nuevamente las transiciones en un nuevo vector llamado "fire_trans". Dentro de estas condiciones externas, se han necesitado utilizar varias variables adicionales para resolver los conflictos efectivos detallados al final del Capítulo previo. Estas variables se explican en el apartado "Cálculo de variables auxiliares".

5.5 Disparo de transiciones y Cálculo del nuevo marcado

Con este vector, y nuevamente una estructura como la anterior, se han asignado los valores de las salidas/actuadores que se tienen que actualizar para cada transición disparada, función "Disparar_transiciones" (ver de nuevo Anexo IV para más detalles). Tras actualizar los valores de los actuadores, se calcula el nuevo marcado según modela la ecuación 1, función "Calcular_marcado".

5.6 Cálculo de variables auxiliares

Por último, queda detallar los cálculos y las variables auxiliares definidas que se ha necesitado implementar para conseguir completar el control, resolviendo los conflictos efectivos que no se han podido evitar en el modelado del sistema.

5.6.1 Contadores posición almacenes

En primer lugar, se han definido tres matrices [2x3] dedicadas a llevar el control del número de piezas que se han introducido al almacén y de la posición en la que se tienen que almacenar por tipo de color ("cont_blue", "cont_green", "cont_gray"). La topología de almacenamiento que se ha considerado es de tipo "*First in, First out*" (FIFO), por esa razón, los contadores se irán actualizando según este criterio conforme se vayan disparando las transiciones de entrada y salida de los almacenes. Se definen de esta manera, los elementos de la primera columna de la matriz como contadores para llevar el control de la posición de entrada, los elementos de la segunda columna contadores para llevar la posición de salida y los de la tercera columna para llevar la cuenta de la cantidad total de piezas almacenadas. Las filas corresponden al número de almacén.

5.6.2 Función Variables almacenes

En segundo lugar, la función "Variables_almacenes" define variables que se basan en la detección de cambios de estado de los pulsadores de los almacenes. Su objetivo es cambiar de estado la estructura "State_Warehouses" para definir las prioridades en las transiciones de almacenamiento o extracción de materiales de los dos almacenes (transiciones 14, 145, 146, 22, 185). De las funciones aquí detalladas esta es la única que resuelve los conflictos efectivos que aparecen y no se han podido resolver de manera simple con un selector. El primer campo de la estructura define qué piezas se tienen que extraer del almacén 1, el segundo es idéntico para el almacén 2 (T145, T185). El tercer campo define qué operación debe realizar el robot del primer almacén en función del nivel de stock mínimo y máximo que se desee tener en ese almacén (T14, T146), lo modifica el usuario. Una vez alcanzado este stock máximo, se activarán y desactivarán las variables "reached_stock" que se utilizarán más adelante en el apartado de comprobar condiciones externas. El cuarto campo de la estructura indica cuándo el usuario ha decidido cambiar de operación en el segundo almacén. Este último campo se ha establecido debido a que no se sabe cuándo se desea extraer el material almacenado, porque podría depender de un sistema de envíos, o de la disponibilidad de recursos de transporte para realizar dichos envíos (T22, T185).

5.6.3 Función Detección movimiento

Finalmente, la función "Detección_movimiento" se ha creado para detectar todas aquellas acciones que se detectan con un sensor de movimiento (donde 1 es en ejecución y 0 es parado). El objetivo es detectar los flancos de bajada del sensor para saber con precisión cuándo el movimiento ha finalizado. El sensor de movimiento por sí solo es insuficiente para determinar si el movimiento ha finalizado, al detectar con un periodo de retraso, el primer dato recibido (un cero) se confunde con que el movimiento ha finalizado antes de empezar (quizá se podría haber realizado esta detección de otras formas, pero se ha deseado realizarlo así por no forzar valores de los sensores y limpieza del código). Dentro de esta función, se establecen también flancos de bajada en algunos sensores difusores, para saber cuándo la pieza los ha atravesado.

5.6.4 Estado de emergencia

Fuera ya del objetivo y alcance principales del trabajo, y detallado muy brevemente, se ha dispuesto también de la base para un estado de emergencia al que se entra en caso de pulsar la seta de emergencia. Este estado se debe basar en la Guía GEMMA [14] (*Guide d'Etude des Modes de Marches et d'Arrêts*); como una visión general, esta guía establece los criterios a seguir en los procedimientos de los arranques y paradas de los sistemas de automatización. Para las paradas determina que se deben poner a cero todos los motores y la electroválvula general del sistema, dejando así desconectado el sistema de la red eléctrica. Debido a este motivo, al entrar en este estado, se ha dejado a cero todos los actuadores a excepción de unos pocos determinados por la función "Emergency_Values" detallada más adelante. Los operarios deberán realizar las acciones necesarias para que el sistema vuelva a un estado anterior al de la parada de emergencia, y se restablecerá el vector marcado a dicho estado. Esta parte del código simplemente establece lo que podría ser un trabajo futuro a este proyecto (ver más detalles Capítulo VII)

Al salir de cualquiera de los estados presionando el pulsador "Stop_Button", la simulación finaliza y se desconecta MATLAB del servidor. Hasta aquí son las funciones que se han utilizado para implementar el sistema de control.

5.7 Funciones secundarias

Detallar también brevemente las funciones que se han utilizado para la inicialización y creación de la estructura de datos, para almacenar el nombre, dirección y valor de los sensores y actuadores:

5.7.1 Lectura de variables

En esta función se toman los valores de las etiquetas correspondientes a los sensores y actuadores de Factory I/O. El fichero "Etiquetas.txt" se extrae del propio simulador una vez se han colocado todas las entradas y salidas que se van a utilizar. Cuando se han extraído los datos, hay que realizar unas pequeñas modificaciones para utilizarlos. Ya hecha esta modificación, se han calculado unas variables que servirán para parametrizar el código y así poder reutilizar parte de él para la implementación de un sistema de control distinto, introduciendo así los datos dentro de una variable tipo tabla de MATLAB (“Tags”).

5.7.2 Emergency Values

Esta función se utiliza para asignar valores a aquellos actuadores que deben tener un valor en concreto en caso de que se realice una parada de emergencia. Por ejemplo, en el caso de las pinzas, si estas ya tenían el valor 1 asignado (están sujetando una pieza), no pueden soltar las piezas sin supervisión de un operario ya que podrían dañar piezas que estén en buen estado. Será necesaria la actuación manual de un operario para realizar cambios en estos actuadores. Recalcar de nuevo que es meramente la base de trabajos futuros como se comentaba en párrafos previos y por tanto está fuera del alcance del trabajo.

5.7.3 Inicialización de matrices y variables

En esta función se inicializan las matrices PRE y POST necesarias para determinar el marcado siguiente, y para comprobar qué transiciones se pueden llegar a disparar, así como el primer marcado de la Red. Para que el cálculo se realice con mayor velocidad, las matrices y vectores se han definido dispersas, para que MATLAB solo guarde las posiciones que sean distintas de cero.

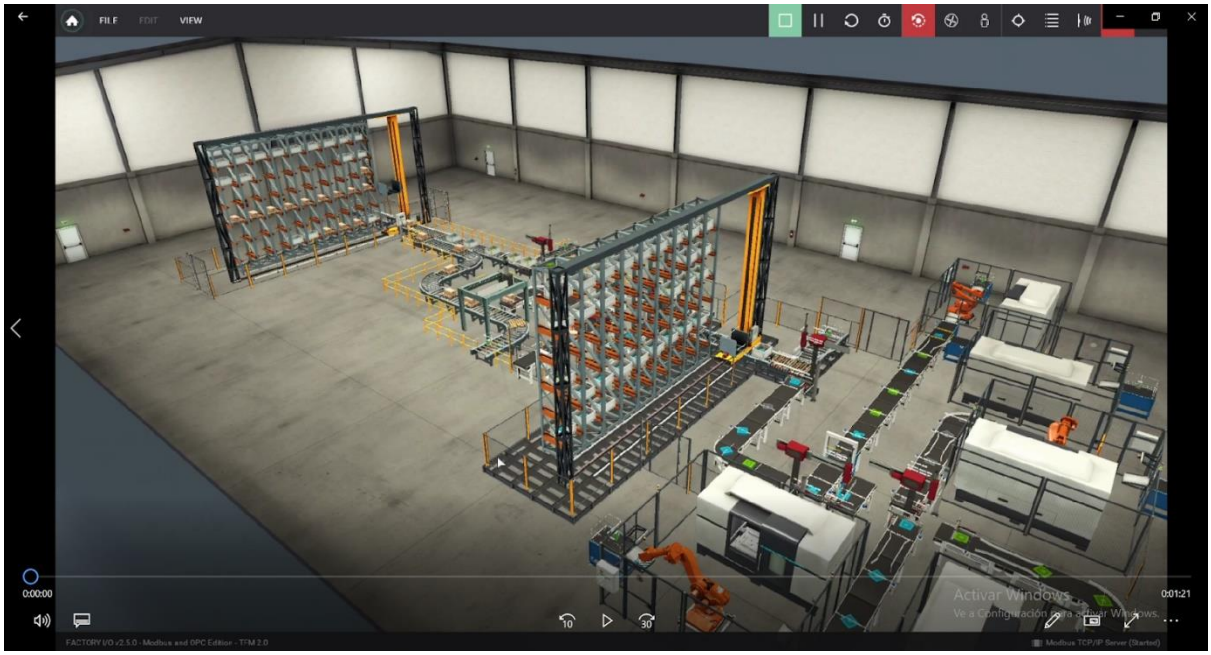
Se inicializan, por supuesto, las variables necesarias que modelan los estados de los almacenes, así como las variables utilizadas para detectar la finalización de movimientos (funciones comentadas anteriormente). Además, se inicializan los contadores que se emplean en el almacenamiento para saber en qué posición se debe almacenar las piezas y qué cantidad de piezas por tipo de color se tiene en cada almacén.

Existe también una estructura de datos, no comentada previamente, que se define como la estructura de prioridades. Aquí se definen las prioridades que poseen algunas transiciones que están en conflicto. Solo se ha definido una prioridad debido a que el resto de los conflictos se han resuelto bien con pulsadores o mediante otras variables. Sin embargo, si se decidiese cambiar alguno de los métodos utilizados para resolver conflictos y definir prioridades, se haría dentro de esta estructura. La prioridad definida resuelve el conflicto entre la transición número 20 y 17, son los eventos que ocurren al utilizar la cinta compartida número 2 (ver Anexo I para más detalles). Se ha deseado que, cuando entren en conflicto, sea cada vez una de ellas la que tenga prioridad de ocurrencia sobre la otra (por defecto se ha decidido que la prioridad en el primer conflicto la posea la transición 17).

Con todo esto el sistema de control queda implementado y listo para su ejecución, en el capítulo siguiente se verá un pequeño ejemplo de ello.

Capítulo VI Verificación del sistema de control desarrollado

Con el fin de poder realizar la verificación del sistema de control implementado y, así entonces, el correcto funcionamiento del sistema se adjunta en este capítulo un enlace a un vídeo donde se puede ver el sistema de producción en ejecución.



https://drive.google.com/drive/folders/1qv3E9hqsxGfM7HHRvRqZKWOdP5trpfIa?usp=sharing

Capítulo VII Conclusiones y resultados

Para dar por finalizada la memoria de este trabajo fin de máster, en este capítulo se exponen las conclusiones y resultados extraídos de todo el trabajo realizado. El primer resultado que se ha obtenido es un sistema de control basado en redes de Petri coloreadas totalmente operativo para el funcionamiento del sistema de producción creado. Se ha concluido que, hoy en día, todavía no es posible poder implementar el sistema de control desde la herramienta principal, CPN tools, y que hay que basarse en la formalidad de la red para poder implementarlo desde otra herramienta que posea una librería Modbus. Se ha verificado igualmente, el propósito de este tipo de técnicas de control en la aplicación de sistemas de gran complejidad y tamaño. Se puede afirmar que las redes de Petri coloreadas son técnicas avanzadas y muy potentes con gran utilidad en el sector de la automatización ya que, gracias a los análisis que realizan, aseguran el cumplimiento de las propiedades del sistema controlado y las especificaciones deseadas.

Como trabajo futuro, quedaría la implementación del controlador desde la herramienta CPN tools cuando el equipo de desarrollo del sistema (actualmente la Universidad de Tecnología de Eindhoven, Países Bajos) cree la posibilidad de poder realizarlo. Asimismo, otra tarea futura que quedaría por realizar es la posible creación de protocolos para el apartado de la vuelta del estado de emergencia al funcionamiento. Se ha dispuesto de las bases para la integración de este apartado dentro de la máquina de estados como ya se ha comentado en los capítulos precedentes, pero quedaría profundizar en los criterios necesarios para realizar con buena praxis la parada de emergencia según explica la guía GEMMA; se podría incluso, llegar a modelar una red según explica esta guía [14].

Quedaría abierta, además, la posibilidad de implementar otros tipos de redes a la escena de simulación creada, con el fin de estudiar otros aspectos igual de importantes, como podrían ser los tiempos de producción, cuellos de botella, gestión y disponibilidad de recursos, etc. Incluso se podrían estudiar las distintas casuísticas de la cadena de producción que no se han escogido para este trabajo fin de máster.

Se puede finalizar diciendo que, se han logrado todos los objetivos y alcance que se habían planteado, obteniendo un modelado y análisis completo basado en redes de Petri coloreadas y la posterior implementación del controlador en un sistema “real”.

Bibliografía

[1]. Duggan, J.A. Comparison of Petri Net and System Dynamics Approaches for Modelling Dynamic Feedback Systems. In Proceedings of the 24th International Conference of the System Dynamics Society, Nijmegen, The Netherlands, 23–27 July 2006; pp. 1–22.

[2]. Safarinejadian, B. Discrete Event Simulation and Petri net Modeling for Reliability Analysis. Int. J. Soft Comput. Softw. Eng. 2012, 2, 25–36.

[3]. Simon, E.; Oyekan, J.; Hutabarat, W.; Tiwari, A.; Turner, C.J. Adapting Petri nets to DES: Stochastic modelling of manufacturing systems. Int. J. Simul. Model. 2018, 17, 5–17.

[4]. Kumar, P.; Gupta, R. Dependency Modeling of a SOA Based System Through Colored Petri Nets. J. Comput. Inf. Technol. 2016, 24, 253–269.

[5]. V. Valero et al., "A Petri net approach for the design and analysis of Web Services Choreographies", The Journal of Logic and Algebraic Programming, vol. 78, pp. 359–380, 2009. http://dx.doi.org/10.1016/j.jlap.2008.09.002

[6]. Ira Sharp, Director of Product Marketing—Automation, Phoenix Contact [Internet]. A New Generation of PLCs for a New Generation of Engineers[Consultado 3 Abril 2021]. Disponible en:https://www.automationworld.com/products/control/article/21109779/a-new-generation-of-plcs-for-a-new-generation-of-engineers

[7]. Dennis Kafura. Petri net Notes [Internet]. Computational Thinking [Consultado 25Abril 2021]. Disponible en:https://people.cs.vt.edu/kafura/ComputationalThinking/Class-Notes/Petri-Net-Notes-Expanded.pdf

[8]. Apuntes del Curso de Ingeniería de Control Grado en Tecnologías Industriales. Redes de Petri [Consultado 2 Mayo 2021]

[9]. Jensen, K., Kristensen, L.M. & Wells, L. Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems. Int J Softw Tools Technol Transfer 9, 213–254 (2007).https://doi.org/10.1007/s10009-007-0038-x

[10]. Atif Memon Advances in Computers. 1st Edition. Academic Press 26th Febraury 2015

[11] Modbus.org. MODBUS APPLICATION PROTOCOL SPECIFICATION V1.1b3. [Internet] 26 Abril 2012 [consultado 18 Mayo 2021]; p2-15 Disponible en: https://modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf

[12]. MathWorks. The MathWorks, Inc. Modbus Explorer App. [Consultado 15 Julio 2021] Disponible: https://es.mathworks.com/help/instrument/configure-a-connection-in-the-modbus-explorer.html

[13]. MathWorks. The MathWorks, Inc. Librería Modbus. [Consultado 22 Julio 2021] Disponible: https://es.mathworks.com/help/instrument/modbus-communication.html

[14]. Guía GEMMA. Agencia ADEPA. 1993

Anexo I: Imágenes del sistema de producción y de la Red de Petri coloreada

Como se ha expuesto anteriormente, la red que se ha creado es demasiado grande para poder visualizarse de manera sencilla. Por este motivo, se adjunta un enlace a un repositorio donde se puede encontrar toda la información al respecto. Detallar antes a que se refiere cada submodelo.

Top	Red de Petri coloreada, nivel superior
Grupo 1	Movimientos del robot 1
Grupo 2	Movimientos del robot 2
Grupo 3	Transportes del grupo de cintas 2
Grupo 4	Transportes grupo de cintas 5 y movimientos R6
Grupo 5	Transportes grupo de cintas 4 y movimientos R5
Grupo 6	Transportes grupo de cintas 6
Grupo 7	Transportes grupo de cintas 7
Grupo 8	Transportes grupo de cintas 8
Grupo 9	Movimientos del robot 3
Grupo 10	Movimientos de entrada robot almacén 1
Grupo 11	Movimientos de salida robot almacén 1
Grupo 12	Movimientos del robot 4 post proceso puesta en cajas
Grupo 13	Movimientos del robot 7
Grupo 14	Movimientos de entrada robot almacén 2
Grupo 15	Movimientos del robot 4 post proceso embalaje
Grupo 16	Movimientos de salida robot almacén 2
Grupo 17	Transportes grupo de cintas 0
Grupo 18	Transportes grupo de cintas 1

Tabla 1: Clasificación de subgrupos de la red según las operaciones modeladas

https://drive.google.com/drive/folders/1iiG0loE3jGiRIMg4lkfqUfOjdwnfkWUz?usp=sharing

En este repositorio y a modo de curiosidad, se introduce también una imagen correspondiente a como correspondería el modelo sin colorear diseñado en Petri net Toolbox, es simplemente a título de curiosidad ya que debido a la dificultad que supone, no se llegó a modelar completamente de lo que realmente ha sido el modelado final del sistema con las redes de Petri coloreadas, pero se puede apreciar rápidamente el tamaño que supondría si se hubiera llegado a modelar entera.

Anexo II: Tablas resumen con los movimientos de robots y correspondencia de cintas

En las siguientes páginas aparecen una serie de tablas resumen, con todos los movimientos que realizan los robots. La primera columna corresponde al nombre del lugar donde se ejecutan y el resto son los movimientos según la pieza o el post proceso al que se envían. Se ha nombrado además del número y nombre del lugar, el submodelo donde aparecen para facilitar su búsqueda dentro de la red.

Nombre del lugar	Movimientos pieza Azul	Movimientos pieza Verde	Movimientos pieza Gris
Top, P17 Mov1 R1	Rotar CW, cerrar mordaza 1	Adelantar brazo, cerrar mordaza 3	Rotar CCW, cerrar mordaza 2
Group 1, P100 Mov2 R1	Baja brazo	Baja brazo	Baja brazo
Group 1, P101 Mov3 R1	Coger pieza, abrir mordaza 1	Coger pieza, abrir mordaza 3	Coger pieza, abrir mordaza 2
Group 1, P102 Mov4 R1	Subir brazo	Subir brazo	Subir brazo
Group 1, P103 Mov5 R1	Rotar CW	Rotar CW	Rotar CCW
Group 1, P273 Mov6 R1	Adelantar brazo	Liberar actuador de rotación	Adelantar brazo
Group 1, P104 Mov7 R1	Baja brazo	Rotar CW	Baja brazo
Group 1, P105 Mov8 R1	Subir brazo, soltar pieza	Baja brazo	Subir brazo, soltar pieza
Group 1, P106 Mov9 R1	Retroceder brazo	Subir brazo, soltar pieza, retroceder brazo	Retroceder brazo
Group 1, P107 Mov10 R1	Rotar CW	Rotar CW	Rotar CW
Group 1, P274 Mov11 R1	Liberar actuador de rotación	Liberar actuador de rotación	Liberar actuador de rotación
Group 1, P108 Mov12 R1	Rotar CW	Rotar CW	Rotar CW

Tabla 2: Acciones realizadas por el Robot 1

Nombre del lugar	Movimientos
Group 2, P74 Mov1 R2	Adelantar brazo
Group 2, P75 Mov2 R2	Bajar brazo, cerrar mordaza 5
Group 2, P76 Mov3 R2	Coger pieza, abrir mordaza 5
Group 2, P77 Mov4 R2	Subir brazo
Group 2, P78 Mov5 R2	Retroceder brazo
Group 2, P79 Mov6 R2	Bajar brazo, cerrar mordaza 4
Group 2, P80 Mov7 R2	Abrir mordaza 4
Group 2, P81 Mov8 R2	Subir brazo
Group 2, P82 Mov9 R2	Rotar CCW
Group 2, P275 Mov10 R2	Liberar actuador de rotación
Group 2, P83 Mov11 R2	Rotar CCW
Group 2, P84 Mov12 R2	Bajar brazo

Group 2, P85 Mov13 R2	Soltar pieza, subir brazo
Group 2, P86 Mov14 R2	Rotar CW
Group 2, P276 Mov15 R2	Liberar actuador de rotación
Group 2, P87 Mov16 R2	Rotar CW

Tabla 3: Acciones realizadas por el Robot 2

Nombre del lugar	Movimientos
Top, P29 Mov1 R3	Adelantar brazo, rotar CCW, cerrar mordaza 6
Group 9, P189 Mov2 R3	Bajar brazo
Group 9, P190 Mov3 R3	Coger pieza, abrir mordaza 6
Group 9, P191 Mov4 R3	Subir brazo
Group 9, P192 Mov 5 R3	Retroceder brazo, rotar CW
Group 9, P193 Mov 6 R3	Bajar brazo, adelantar brazo
Group 9, P194 Mov 7 R3	Soltar pieza, subir brazo, retroceder brazo

Tabla 4: Acciones realizadas por el Robot 3

Nombre del lugar	Movimientos post proceso embalaje (Top P50, Group 15 P232-237)	Movimientos post proceso puesta en caja (Top P41, Group12 P212-217)
Mov1 R4	Adelantar brazo, bajar brazo	Adelantar brazo, bajar brazo
Mov2 R4	Coger pieza	Coger pieza
Mov3 R4	Subir brazo	Subir brazo
Mov4 R4	Adelantar brazo, rotar CW	Adelantar brazo, rotar CCW
Mov5 R4	Bajar brazo	Bajar brazo
Mov6 R4	Soltar pieza, subir brazo	Soltar pieza, subir brazo
Mov7 R4	Retroceder brazo, rotar CCW	Retroceder brazo, rotar CW

Tabla 5: Acciones realizadas por el Robot 4

Nombre del lugar	Movimientos
Group 5, P146 Mov1 R5	Bajar brazo, cerrar mordaza 7
Group 5, P147 Mov2 R5	Coger pieza, abrir mordaza 7
Group 5, P148 Mov3 R5	Subir brazo
Group 5, P149 Mov4 R5	Rotar CCW
Group 5, P279 Mov5 R5	Liberar actuador de rotación
Group 5, P150 Mov6 R5	Rotar CCW, adelantar brazo
Group 5, P151 Mov7 R5	Bajar brazo, cerrar mordaza 8
Group 5, P152 Mov8 R5	Soltar pieza, subir brazo, abrir mordaza 8
Group 5, P153 Mov9 R5	Retroceder brazo, rotar CW y subir mordaza 8 (subir cada dos veces)
Group 5, P280 Mov10 R5	Liberar actuador de rotación
Group 5, P154 Mov11 R5	Rotar CW

Tabla 6: Acciones realizadas por el Robot 5

Nombre del lugar	Movimientos
Group 4, P128 Mov1 R6	Bajar brazo, cerrar mordaza 9
Group 4, P129 Mov2 R6	Coger pieza, abrir mordaza 9
Group 4, P130 Mov3 R6	Subir brazo
Group 4, P131 Mov4 R6	Rotar CCW
Group 4, P277 Mov5 R6	Liberar actuador de rotación
Group 4, P132 Mov6 R6	Rotar CCW, adelantar brazo
Group 4, P133 Mov7 R6	Bajar brazo
Group 4, P134 Mov8 R6	Soltar pieza, subir brazo, retroceder brazo
Group 4, P135 Mov9 R6	Rotar CW
Group 4, P278 Mov10 R6	Liberar actuador de rotación
Group 4, P136 Mov11 R6	Rotar CW

Tabla 7: Acciones realizadas por el Robot 6

Nombre del lugar	Movimientos
Top, P55 Mov1 R7	Bajar brazo, desplazar eje Y, cerrar mordaza 10
Group 13, P219 Mov2 R7	Coger caja, abrir mordaza 10
Group 13, P220 Mov3 R7	Subir brazo
Group 13, P221 Mov4 R7	Desplazar eje X, desplazar eje Y
Group 13, P222 Mov5 R7	Bajar brazo
Group 13, P223 Mov6 R7	Soltar pieza, subir brazo
Group 13, P224 Mov7 R7	Desplazar eje X, desplazar eje Y a posición de reposo

Tabla 8: Acciones realizadas por el Robot 7

Nombre del lugar	Movimientos
Mov1 entrada (Top P33 y P63)	Desplazar palas izquierda
Mov2 entrada (Group 10 y 14, P195 y P225)	Subir palas
Mov3 entrada (Group 10 y 14, P196 y P226)	Retroceder palas a posición central
Mov4 entrada (Group 10 y 14, P197 y P227)	Acudir a posición de almacén deseada
Mov5 entrada (Group 10 y 14, P198 y P228)	Desplazar palas derecha
Mov6 entrada (Group 10 y 14, P199 y P229)	Bajar palas
Mov7 entrada (Group 10 y 14, P200 y P230)	Retroceder palas a posición central
Mov8 entrada (Group 10 y 14, P201 y P231)	Vuelta a posición de reposo

Tabla 9: Acciones realizadas por Robots almacenes operación entrada de material

Nombre del lugar	Movimientos
Mov1 salida (Group 11 y 16, P204 y P238)	Acudir a posición de almacén deseada
Mov2 salida (Group 11 y 16, P205 y P239)	Desplazar palas derecha

Mov3 salida (Group 11 y 16, P206 y P240)	Subir palas
Mov4 salida (Group 11 y 16, P207 y P241)	Retroceder palas a posición central
Mov5 salida (Group 11 y 16, P208 y P242)	Vuelta a posición de reposo
Mov6 salida (Group 11 y 16, P209 y P243)	Desplazar palas derecha
Mov7 salida (Group 11 y 16, P210 y P244)	Bajar palas
Mov8 salida (Group 11 y 16, P211 y P245)	Retroceder palas a posición central

Tabla 10: Acciones realizadas por Robots almacenes operación salida de material

A continuación, aparecen los transportes de cintas con el nombre de Factory I/O de la cinta o cintas que realizan dicho transporte:

Nombre del lugar	Nombre de las cintas Nº 0
Group 17 P246 transporte bases M2_1	Base_Belt_Conveyor 1-2
Group 17 P249 transporte bases M2_2	Base_Belt_Conveyor 2-3
Group 17 P252 transporte bases M2_3	Base_Belt_Conveyor 3-4
Group 17 P255 transporte bases M2_4	Base_Belt_Conveyor 4-5
Group 17 P258 transporte bases M2_5	Base_Belt_Conveyor 5-6
Top P15 clasificar base	Base_Belt_Conveyor 6-7
Top P22 transporte bases M2_6	Base_Belt_Conveyor 7-8

Tabla 11: Grupo de cintas 0

Nombre del lugar	Nombre de las cintas Nº 1
Group 18 P261 Transporte tapas M2_1	“Color”_Belt_Conveyor 1-2
Group 18 P264 Transporte tapas M2_2	“Color”_Belt_Conveyor 2-3
Top P12 Transporte tapas M2_3	“Color”_Belt_Conveyor 3-4
Top P19 Transporte tapas M2_4	Base_Belt_Conveyor 9

Tabla 12: Grupo de cintas 1

Nombre del lugar	Nombre de las cintas Nº 2
Top P25 Transporte1 almacen intermedio	Belt_Conveyor 1
Group 3 P90 Transporte2 almacen intermedio	Belt_Conveyor 1, Curved_Belt_Conveyor_2_CW
Group 3 P93 Transporte3 almacen intermedio	Curved_Belt_Conveyor_2_CW, Belt_Conveyor 3
Group 3 P96 Transporte4 almacen intermedio	Belt_Conveyor 3-4
Group 3 P99 Transporte5 almacen intermedio	Belt_Conveyor 4-5

Tabla 13: Grupo de cintas 2

Nombre del lugar	Nombre de las cintas Nº 3
Top P31 transporte 3_1	Chain_Transfer 6,7 y Loading_Conveyor 1
Top P37 transporte 3_2	Chain_Transfer 1,2 y Loading_Conveyor 2

Tabla 14: Grupo de cintas 3

Nombre del lugar	Nombre de las cintas Nº 4
Top P51 transporte embalaje	Belt_Conveyor 6
Group 5 P140 transporte embalaje 2	Belt_Conveyor 6-7

Tabla 15: Grupo de cintas 4

Nombre del lugar	Nombre de las cintas Nº 5
Top P42 transporte puesta en caja_1	Belt_Conveyor 8
Group 4 P110 transporte puesta en caja_2	Belt_Conveyor 8-9
Group 4 P113 transporte puesta en caja_3	Belt_Conveyor 9-10
Group 4 P116 transporte puesta en caja_4	Belt_Conveyor 10-11
Group 4 P119 transporte puesta en caja_5	Belt_Conveyor 11-12

Tabla 16: Grupo de cintas 5

Nombre del lugar	Nombre de las cintas Nº 6
Top P44 transporte almacen final 1_1	Chain_Transfer_3 y Roller_Conveyor_13
Group 6 P171 transporte almacen final 1_2	Roller_Conveyor_13-14
Group 6 P174 transporte almacen final 1_3	Roller_Conveyor_14-15
Group 6 P177 transporte almacen final 1_4	Roller_Conveyor_15-16

Tabla 17: Grupo de cintas 6

Nombre del lugar	Nombre de las cintas Nº 7
Top P53 transporte paletizadora_1	Box_Belt_Conveyor 1-2
Group 7 P158 transporte paletizadora_2	Box_Belt_Conveyor 2-3-4
Group 7 P161 transporte paletizadora_3	Box_Belt_Conveyor 4-5
Group 7 P164 transporte paletizadora_4	Box_Belt_Conveyor 5-6-7

Tabla 18: Grupo de cintas 7

Nombre del lugar	Nombre de las cintas Nº 8
Top P57 Transporte a almacen final 2_1	Roller_Conveyor 4-5
Group 8 P179 Transporte a almacen final 2_2	Roller_Conveyor 5-6
Group 8 P182 Transporte a almacen final 2_3	Roller_Conveyor 6-7-8
Group 8 P185 Transporte a almacen final 2_4	Roller_Conveyor 8-9-10
Group 8 P188 Transporte a almacen final 2_5	Roller_Conveyor 10-11

Tabla 19: Grupo de cintas 8

Nombre del lugar	Nombre de las cintas Nº Cinta compartida 1
P5 transporte cinta comp (color M3_1)	Base_Belt_Conveyor 1, Base_Conveyor_Scale 1
P5 transporte cinta comp (color M3_2)	Base_Belt_Conveyor 1, Base_Conveyor_Scale 2

Tabla 20: Grupo de cintas compartidas 1

Nombre del lugar	Nombre de las cintas Nº Cinta compartida 2
Top P48 Pasar a cinta final	Roller_Conveyor 16-17
Top P61 Pasar a cinta final 2	Chain_Transfer 4,5 (dirección derecha)
Top P62 transporte final almacen final	Chain_Transfer 5, Roller_Conveyor 17 y Loading_Conveyor_3

Tabla 21: Grupo de cintas compartidas 2

Nombre del lugar	Nombre de las cintas Nº Cinta final
Top P67 transporte final	Loading_Conveyor_4, Roller_Conveyor_18

Tabla 22: Grupo de cintas transporte final

Anexo III: Listado de sensores y actuadores utilizados

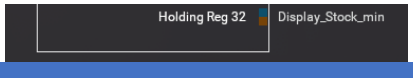
En la columna izquierda, aparecen todos los sensores que se han empleado dentro de la simulación, primero aparecen los sensores de tipo "*coil*" que tendrán valores tipo booleanos 1 o 0 y luego los sensores tipo "*inputreg*" que serán los que tengan valores de tipo real. A la derecha están todos los actuadores sobre los que se trabaja. Del mismo modo, al comienzo aparecen los de tipo "*coil*" con salida 1 o 0 y luego los de tipo "*holdregs*" que tendrán las salidas tipo real. Factory I/O trabaja con enteros de 16 bits sin signo, todos aquellos actuadores y sensores de tipo real que trabajen con voltios, la lectura y escritura ha de ser en milivoltios.

Diffuse_Sensor_Green_3	Coil 33	Coil 181	Green_Belt_Conveyor_3
Diffuse_Sensor_Green_4	Coil 34	Coil 182	Green_Belt_Conveyor_4
Diffuse_Sensor_Pallet_1	Coil 35	Coil 183	Box_Belt_Conveyor_1
Diffuse_Sensor_Pallet_2	Coil 36	Coil 184	Box_Belt_Conveyor_2
Diffuse_Sensor_Pallet_3	Coil 37	Coil 185	Box_Curved_Belt_Conveyor_3_CCW
Diffuse_Sensor_Pallet_4	Coil 38	Coil 186	Box_Belt_Conveyor_4
Diffuse_Sensor_Pallet_5	Coil 39	Coil 187	Box_Belt_Conveyor_5
Diffuse_Sensor_Pallet_6	Coil 40	Coil 188	Box_Curved_Belt_Conveyor_6_CCW
Diffuse_Sensor_Pallet_7	Coil 41	Coil 189	Box_Belt_Conveyor_7
Diffuse_Sensor_Pallet_8	Coil 42	Coil 190	Chain_Transfer_1
Diffuse_Sensor_Pallet_9	Coil 43	Coil 191	Chain_Transfer_2
Diffuse_Sensor_Pallet_10	Coil 44	Coil 192	Chain_Transfer_3
Diffuse_Sensor_Pallet_11	Coil 45	Coil 193	Chain_Transfer_4
Diffuse_Sensor_Pallet_12	Coil 46	Coil 194	Chain_Transfer_5
Diffuse_Sensor_Pallet_13	Coil 47	Coil 195	Chain_Transfer_3_Left
Diffuse_Sensor_Postp_1	Coil 48	Coil 196	Chain_Transfer_2_Right
Diffuse_Sensor_Postp_2	Coil 49	Coil 197	Chain_Transfer_4_Right
Diffuse_Sensor_Postp_3	Coil 50	Coil 198	Chain_Transfer_5_Right
Diffuse_Sensor_Postp_4	Coil 51	Coil 199	FACTORY I/O (Run)
Diffuse_Sensor_Postp_5	Coil 52	Coil 200	FACTORY I/O (Pause)
Diffuse_Sensor_Postp_6	Coil 53	Coil 201	Left_Positioner_1_Clamp
Diffuse_Sensor_Postp_7	Coil 54	Coil 202	Left_Positioner_2_Clamp
Diffuse_Sensor_Postp_8	Coil 55	Coil 203	Left_Positioner_3_Clamp
Diffuse_Sensor_United_1	Coil 56	Coil 204	Right_Positioner_4_Clamp
Diffuse_Sensor_United_2	Coil 57	Coil 205	Right_Positioner_5_Clamp
Diffuse_Sensor_United_3	Coil 58	Coil 206	Right_Positioner_6_Clamp
Diffuse_Sensor_United_4	Coil 59	Coil 207	Right_Positioner_7_Clamp
Diffuse_Sensor_United_5	Coil 60	Coil 208	Right_Positioner_8_Clamp
Diffuse_Sensor_United_6	Coil 61	Coil 209	Right_Positioner_8_Raise
Left_Positioner_1_Clamped	Coil 62	Coil 210	Right_Positioner_9_Clamp
Left_Positioner_2_Clamped	Coil 63	Coil 211	Left_Positioner_10_Clamp
Left_Positioner_3_Clamped	Coil 64	Coil 212	Conveyor_Scale_Base1_Emitter
Right_Positioner_4_Clamped	Coil 65	Coil 213	Conveyor_Scale_Base2_Emitter
Right_Positioner_5_Clamped	Coil 66	Coil 214	Conveyor_Scale_Blue_Emitter
Right_Positioner_6_Clamped	Coil 67	Coil 215	Conveyor_Scale_Gray_Emitter
Right_Positioner_7_Clamped	Coil 68	Coil 216	Conveyor_Scale_Green_Emitter

Right_Positioner_8_Clamped	Coil 69	Coil 217	Emitter_bases_1
Right_Positioner_9_Clamped	Coil 70	Coil 218	Emitter_bases_2
Left_Positioner_10_Clamped	Coil 71	Coil 219	Emitter_blue_lids
Right_Positioner_8_Limit	Coil 72	Coil 220	Emitter_boxes_1
Machining_Center_3_1_Is_Busy	Coil 73	Coil 221	Emitter_boxes_2
Machining_Center_3_2_Is_Busy	Coil 74	Coil 222	Emitter_gray_lids
Machining_Center_1_Blue_Is_Busy	Coil 75	Coil 223	Emitter_green_lids
Machining_Center_1_Gray_Is_Busy	Coil 76	Coil 224	Emitter_packages
Machining_Center_1_Green_Is_Busy	Coil 77	Coil 225	Emitter_pallets
Pick_&_Place_Box_Detected	Coil 78	Coil 226	Light_Indicator_Green
Stacker_Crane_1_Moving_Z	Coil 79	Coil 227	Light_Indicator_Red
Stacker_Crane_1_Moving_X	Coil 80	Coil 228	Light_Indicator_Yellow
Stacker_Crane_1_Left_Limit	Coil 81	Coil 229	Output/ON_Input/OFF
Stacker_Crane_1_Right_Limit	Coil 82	Coil 230	Error_stock_parameters
Stacker_Crane_1_Middle_Limit	Coil 83	Coil 231	FACTORY I/O (Reset)
Stacker_Crane_2_Moving_Z	Coil 84	Coil 232	Machining_Center_3_1_Start
Stacker_Crane_2_Moving_X	Coil 85	Coil 233	Machining_Center_3_2_Start
Stacker_Crane_2_Left_Limit	Coil 86	Coil 234	Machining_Center_1_Blue_Start
Stacker_Crane_2_Right_Limit	Coil 87	Coil 235	Machining_Center_1_Gray_Start
Stacker_Crane_2_Middle_Limit	Coil 88	Coil 236	Machining_Center_1_Green_Start
Diffuse_Sensor_Boxes_1	Coil 89	Coil 237	Remover_1
Diffuse_Sensor_Boxes_2	Coil 90	Coil 238	Remover_3
Diffuse_Sensor_Packages	Coil 91	Coil 239	Stacker_Crane_1_Left
Diffuse_Sensor_pallet_emitter	Coil 92	Coil 240	Stacker_Crane_1_Lift
Diffuse_Sensor_remover_1	Coil 93	Coil 241	Stacker_Crane_1_Right
Diffuse_Sensor_remover_2	Coil 94	Coil 242	Stacker_Crane_2_Left
Stackable_boxes	Coil 95	Coil 243	Stacker_Crane_2_Lift
Packaging	Coil 96	Coil 244	Stacker_Crane_2_Right
Diffuse_Sensor_Pallet_14	Coil 97	Coil 245	Roller_Conveyor_1
Diffuse_Sensor_Robot1_Position0	Coil 98	Coil 246	Roller_Conveyor_2
Diffuse_Sensor_Robot2_Position0	Coil 99	Coil 247	Curved_Roller_Conveyor_3_CCW
Diffuse_Sensor_Robot3_Position0	Coil 100	Coil 248	Roller_Conveyor_4
Diffuse_Sensor_Robot4_Position0	Coil 101	Coil 249	Roller_Conveyor_5
Diffuse_Sensor_Robot6_Position0	Coil 102	Coil 250	Roller_Conveyor_6
Diffuse_Sensor_Robot5_Position0	Coil 103	Coil 251	Curved_Roller_Conveyor_7_CW
Robot1_Detected	Coil 104	Coil 252	Roller_Conveyor_8
Robot2_Detected	Coil 105	Coil 253	Curved_Roller_Conveyor_9_CCW


Robot3_Detected
Robot4_Detected
Robot5_Detected
Robot6_Detected
Robot1_Rotating
Robot2_Rotating
Robot3_Rotating
Robot4_Rotating
Robot5_Rotating
Robot6_Rotating
Extract_Blue_WH1
Extract_Green_WH1
Extract_Gray_WH1
Extract_Blue_WH2
Extract_Green_WH2
Extract_Gray_WH2
Diffuse_Sensor_Boxes_3
Storage_Operation_WH2
Store_WH2
Extract_WH2
Robot7_X_Position
Robot7_Y_Position
Robot7_Z_Position
Conveyor_Scale_Base1_Emitter_Weight
Conveyor_Scale_Base2_Emitter_Weight
Conveyor_Scale_Blue_Emitter_Weight
Conveyor_Scale_Gray_Emitter_Weight
Conveyor_Scale_Green_Emitter_Weight
Robot1_PositionX
Robot1_PositionZ
Robot2_PositionX
Robot2_PositionZ
Robot3_PositionX
Robot3_PositionZ
Robot4_PositionX
Robot4_PositionZ
Robot5_PositionX
Coil 106
Coil 107
Coil 108
Coil 109
Coil 110
Coil 111
Coil 112
Coil 113
Coil 114
Coil 115
Coil 116
Coil 117
Coil 118
Coil 119
Coil 120
Coil 121
Coil 122
Coil 123
Coil 124
Coil 125
Input Reg 126
Input Reg 127
Input Reg 128
Input Reg 129
Input Reg 130
Input Reg 131
Input Reg 132
Input Reg 133
Input Reg 134
Input Reg 135
Input Reg 136
Input Reg 137
Input Reg 138
Input Reg 139
Input Reg 140
Input Reg 141
Input Reg 142
Coil 254
Coil 255
Coil 256
Coil 257
Coil 258
Coil 259
Coil 260
Coil 261
Coil 262
Coil 263
Coil 264
Coil 265
Coil 266
Coil 267
Coil 268
Coil 269
Coil 270
Coil 271
Coil 272
Coil 273
Coil 274
Coil 275
Coil 276
Coil 277
Coil 278
Coil 279
Coil 280
Coil 281
Coil 282
Coil 283
Coil 284
Coil 285
Coil 286
Coil 287
Coil 288
Coil 289
Coil 290
Roller_Conveyor_10
Roller_Conveyor_11
Roller_Conveyor_12
Roller_Conveyor_13
Roller_Conveyor_14
Roller_Conveyor_15
Roller_Conveyor_16
Roller_Conveyor_17
Roller_Conveyor_18
Roller_Conveyor_19
Loading_Conveyor_1
Loading_Conveyor_2
Loading_Conveyor_3
Loading_Conveyor_4
Robot7_Grab
Robot1_Grab
Robot2_Grab
Robot3_Grab
Robot4_Grab
Robot5_Grab
Robot6_Grab
Robot1_Rotate_CW
Robot1_Rotate_CCW
Robot2_Rotate_CW
Robot2_Rotate_CCW
Robot3_Rotate_CW
Robot3_Rotate_CCW
Robot4_Rotate_CW
Robot4_Rotate_CCW
Robot5_Rotate_CW
Robot5_Rotate_CCW
Robot6_Rotate_CW
Robot6_Rotate_CCW
Light_Indicator_Blue_WH1
Light_Indicator_Green_WH1
Light_Indicator_Gray_WH1
Light_Indicator_Blue_WH2


Robot5_PositionZ
Robot6_PositionX
Robot6_PositionZ
Set_stock_max
Set_stock_min
Input Reg 143
Input Reg 144
Input Reg 145
Input Reg 146
Input Reg 147
Coil 291
Coil 292
Coil 293
Coil 294
Coil 295
Coil 296
Coil 297
Coil 298
Coil 299
Coil 300
Coil 301
Coil 302
Coil 303
Holding Reg 30
Holding Reg 30
Holding Reg 30
Holding Reg 30
Holding Reg 30
Holding Reg 30
Holding Reg 30
Holding Reg 31
Holding Reg 31
Holding Reg 31
Holding Reg 31
Holding Reg 31
Holding Reg 31
Holding Reg 31
Holding Reg 31
Holding Reg 31
Holding Reg 32
Holding Reg 32
Holding Reg 32
Holding Reg 32
Holding Reg 32
Holding Reg 32
Holding Reg 32
Light_Indicator_Green_WH2
Light_Indicator_Gray_WH2
Chain_Transfer_6
Chain_Transfer_6_Left
Chain_Transfer_7
Light_Storage_Operation_WH2
Roller_Conveyor_Emitter_boxes_1
Roller_Conveyor_Emitter_boxes_2
Machining_Center_3_1_Stop
Machining_Center_3_2_Stop
Machining_Center_1_Blue_Stop
Machining_Center_1_Green_Stop
Machining_Center_1_Gray_Stop
Stacker_Crane_1_Target_Position
Stacker_Crane_2_Target_Position
Robot7_X_Set_Point
Robot7_Y_Set_Point
Robot7_Z_Set_Point
Robot1_SetX
Robot1_SetZ
Robot2_SetX
Robot2_SetZ
Robot3_SetX
Robot3_SetZ
Robot4_SetX
Robot4_SetZ
Robot5_SetX
Robot5_SetZ
Robot6_SetX
Robot6_SetZ
Display_Blue_WH1
Display_Green_WH1
Display_Gray_WH1
Display_Blue_WH2
Display_Green_WH2
Display_Gray_WH2
Display_Stock_max


Holding Reg 32
Display_Stock_min

Anexo IV: Código creado en MATLAB

Al tratarse el código realizado para este trabajo fin de máster de un código muy extenso y farragoso, no se ha considerado oportuno introducirlo ni en los anexos ni en la memoria. Sin embargo, para que el lector pueda visualizar lo que se ha creado en caso de mayor interés, se adjunta la carpeta "Código MATLAB" al repositorio creado para este trabajo. Dentro de esta carpeta, se encuentra un *live script* de MATLAB en el que se puede leer el código junto a pequeñas explicaciones en forma de texto. Este entorno es muy llamativo, ya que se trata de un entorno totalmente interactivo, combina el código con texto formateado y algunas funcionalidades más. Por si se prefieren ver los scripts por separado, en la subcarpeta se encuentran todas las funciones y el código principal.

https://drive.google.com/drive/folders/1NQk6cbprqVmnAhH4d3wzagp0KHS738bE?usp=sharing