UNIVERSIDAD DE BELGRANO

Las tesinas de Belgrano

Facultad de Ingeniería y Tecnología Informática
Carrera de Ingeniería Informática

Real Time Food

Nº 125 **Mariano S. Rodríguez**

Tutor: Alicia Romano - Juan J. Goldschtein

Departamento de Investigación
Febrero 2005

Indice

Agradecimientos

Ing. Carlos A. Chueke. Por su ayuda en el armado de la placa interfaz del sistema.
Ing. Adrián Rudzikas. Por los elementos de hardware facilitados.
Juan Damico. Por los LEGOs facilitados.

1. Introducción al Trabajo Final de Carrera

Dentro de los distintos campos de aplicación de la informática se encuentran los sistemas denominados Sistemas de Tiempo Real (STR en adelante). Para introducirnos en estos sistemas podemos utilizar una definición canónica de los STR:

« Un sistema de tiempo real es aquel en el que la corrección de la computación no solo depende de la corrección de la lógica de la computación, sino también del tiempo en el cuál el resultado es producido. Si los límites temporales del sistema no son alcanzados, se puede decir que una falla del sistema ha ocurrido.» [FAQRTS03]

A fin de tratar obtener mayor experiencia en estos sistemas, he desarrollado uno que permita identificar fácilmente los componentes y su funcionamiento.

Dentro de las posibilidades evaluadas para la implementación de un S.T.R. fue considerado un sistema de alimentación de animales pequeños. Este sistema tiene como objetivo el llenado de recipientes de alimento una vez que es detectada la carencia del mismo.

A su vez es mi intención lograr un acercamiento a los sistemas de tiempo real, haciendo uno que brinde una solución a un problema simple ya que generalmente estos sistemas se los vincula con soluciones altamente específicas y complejas. Además considero que a medida que pase el tiempo los STR serán implementados para resolver problemas de menor complejidad.

Esta afirmación surge del análisis de dos motivos: uno es el costo de los componentes y otro es la aplicación de los principios de ingeniería de software de los sistemas tradicionales a los sistemas de tiempo real. Logrando bajar los costos de desarrollo que es uno de los motivos por los cuáles no es rentable aplicar una solución de tiempo real a problemas sencillos.

2.1. Objetivos del Trabajo Final de Carrera

El objetivo del presente trabajo es el desarrollo de un STR para controlar una cadena de alimentación de animales pequeños. Incluyendo la detección de la carencia de alimento hasta el llenado de los recipientes vacíos.

Para lograr el control se utilizarán sensores a través de los cuáles se podrá determinar el estado del recipiente. Para el llenado de los recipientes se usará un transporte cuya movilidad será alcanzada utilizando dos motores paso a paso. A su vez, es necesario contemplar que cada recipiente posea una prioridad, que será utilizada para determinar el orden de llenado.

2.2. Alcance del Trabajo final de Carrera.

El desarrollo del presente trabajo final de carrera comprende la construcción de una interfaz que permita conectar los sensores a la PC, el desarrollo del software necesario para la obtención de los datos y la construcción del transporte necesario para llenar los recipientes.

Los sensores son de dos tipos, unos incorporados a los recipientes para determinar su estado y otros utilizados para detectar la presencia del transporte.

Para poder alcanzar los límites temporales impuestos fue necesario utilizar funciones que funcionen paralelamente. Ya que mientras se ordenan los recipientes en la cola de atención debe seguir evaluándose el estado de los recipientes.

Al momento de elegir en el modo de ejecución de las mencionadas funciones se optó por utilizar hilos de ejecución (threads) debido a que todas las funciones compartían los datos a utilizar y además el comportamiento de los hilos de ejecución era eficiente.

A fin de lograr una implementación que permita cierta portabilidad de S.O. se utilizaron librerías (Pthreads.h) que respeten los estándares POSIX.

2.3. Límites.

Dentro de los aspectos a tener en cuenta tenemos:

- Cantidad de recipientes a ser atendidos, que son 3.
- La velocidad de movimiento del transporte está determinada por las revoluciones del motor paso a paso.
- El peso del alimento a ser transportado no puede exceder los 100 gr.
- La distancia entre la placa y el recipiente más lejano no puede exceder los 15 mts.
- El tamaño de la maqueta para permitir su transporte.

2.4. Entregables:

Los entregables consisten en el presente trabajo en su versión escrita, una maqueta que simula el entorno en el cuál funcionará el STR desarrollado y un disco compacto con todos los documentos relacionados con el trabajo. También se incluirá en el mencionado disco compacto el software desarrollado.

3. Marco Teórico

3.1 Sistemas de Tiempo Real.

« Un sistema de tiempo real es aquel en el que la corrección de la computación no solo depende de la corrección de la lógica de la computación, sino también del tiempo en el cuál el resultado es producido. Si los límites temporales del sistema no son alcanzados, se puede decir que una falla del sistema ha ocurrido.»[FAQRTS03]

Según la definición canónica, podemos afirmar que todo sistema cuyas limitaciones sean temporales es un sistema de tiempo real. Pero el Diccionario Oxford de Computación ofrece una definición que evidencia una relación más importante, que es una característica propia de los STR:

Cualquier sistema en el cual el tiempo en el que la salida es realizada, es significativo. Esto se dá porque la entrada corresponde a alguno movimiento en el mundo físico, y la salida está relacionada con ese movimiento. El retraso entre el tiempo de la entrada y el tiempo de salida tiene que ser lo suficientemente corto para una línea de tiempo aceptable. [GREHAN-MOOTE-CYLIAX98]

Algunos autores reflejan esta relación como si el STR fuera un subsistema de control y el entorno fuera el subsistema controlado, formando ambos un solo sistema. De esta manera se puede lograr tener una idea de cómo funciona un STR.

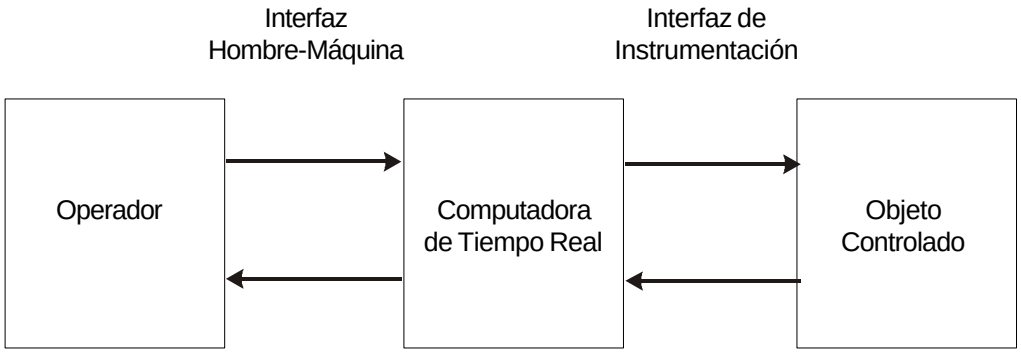

Fig.1 – Sistema de Tiempo Real

Si bien el tiempo es el factor determinante para definir el correcto funcionamiento de un STR, erróneamente se tiende a afirmar que cuanto más rápido se realiza una tarea, mejor funciona el STR. Para evaluar el correcto funcionamiento de un STR hay que tener en cuenta los límites según una línea de tiempo, que pueden estar expresados en milisegundos, en algunos casos y en segundos en otros.

En base a la importancia que tienen los límites temporales de un STR pueden llegar a clasificarse en dos subgrupos:

- Hard Real –Time. Son aquellos en los cuáles no cumplir con alguno de los requerimientos temporales genera una falla general del sistema. Entendiendo como general a una falla completa sin posibilidad de continuar con el funcionamiento.
- Soft Real –Time. A diferencia de los anteriores, se pueden llegar a no alcanzar algunos de los límites especificados pero igualmente el STR continúa su ejecución y la única consecuencia se refleja en una degradación de la calidad de las respuestas. Para estos STR se pueden definir niveles de tolerancia en el momento del diseño de los mismos.
  [MUÑOZ03]

Por la clase de restricciones de tiempo. • duros • blandos • firmes
Por la relación entre las escalas de tiempo. • Basados en eventos. • Basados en reloj. • Interactivos.

Por su integración en el entorno.
• Reactivos / No Reactivos.
  • Embebidos.
  • No embebidos.
  • débilmente acoplado.
  • Orgánicos.

Tabla 1 – Clasificación de los S.T.R. según distintos aspectos. [WAINER97]

Para responder a los estímulos del entorno es necesario nombrar dos características que están ligadas fuertemente a los STR:

Concurrencia: es necesaria para lograr atender a los estímulos del entorno y además para utilizar de manera más eficiente los recursos.[BIRREL89]

Fiabilidad y seguridad: dependiendo de las funciones que los STR desarrollan deben lograrse un nivel adecuado de fiabilidad y seguridad.

Scheduling: otro aspecto importante dentro de los STR es el programado temporal de ejecución de procesos. El objetivo del programado es facilitar todos lo recursos necesarios para la ejecución de cada proceso haciendo posible que se pueda predecir el funcionamiento del STR.

Dentro del Scheduling de STR podemos encontrar distintos tipos:

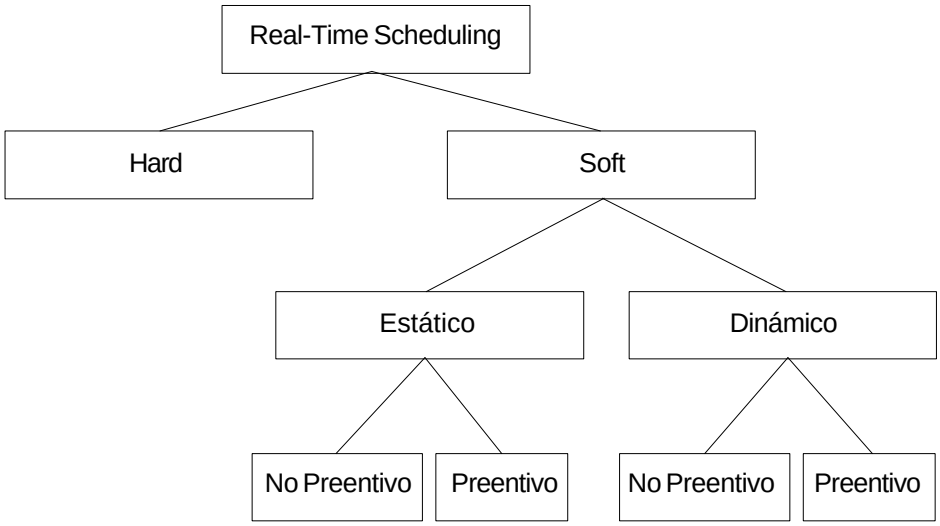

Fig. 2 – Tipos de Scheduling.

El presente trabajo se encuentra ubicado dentro de los STR blandos, basado en eventos, no embebido, con concurrencia de procesos y utiliza un schedule estático definido al momento de diseño.

Por lo tanto, el STR desarrollado permitirá que se supere el tiempo máximo para llenar el recipiente de alimento y continuará funcionando. En caso que la tolva principal se vacíe se producirá una falla grave del sistema y frenará su ejecución.

La activación del sistema se dará cuando un recipiente carezca de alimento.

Los procesos de control y atención de recipientes se ejecutan en forma concurrente utilizando librerías de threads y dentro de un schedule definido en base a los tiempos de ejecución de cada uno.

3.2 Estado del Arte.

En la actualidad los STR se encuentran ampliamente implementados en la gran mayoría de los aspectos de nuestra vida. Esta gran expansión ha hecho que gran cantidad entidades hayan desarrollado distintas soluciones al problema de la interacción con el entorno. Esta situación ha evidenciado una necesidad de desarrollar herramientas y estándares para construir los sistemas de tiempo real para lograr objetivos de la ingeniería de software tales como la reusabilidad, estabilidad y predicibilidad.

Dentro de los productos orientados a Sistemas de tiempo real se pueden clasificar en dos grandes grupos, uno que engloba a los sistemas operativos y otro que contiene a los productos necesarios para la obtención de datos.

El sistema que se desarrollo en este trabajo se encuentra en el segundo grupo, pero no he encontrado alguno similar dado que la mayoría está orientado a tareas más complejas. Justamente esta situación fue la que hizo que surgiera la necesidad de realizar un sistema de tiempo real que satisfaga una necesidad simple demostrando que estos sistemas pueden ser implementados a todo tipo de problema y no solamente a problemas altamente complejos y específicos. Ya que si la expansión de estos sistemas continúa, en próximos años estarán aplicados a la resolución de problemas sencillos.

4. Implementación del Proyecto.

4.1 Descripción del escenario.

El problema a solucionar por el sistema que es el motivo de este trabajo consiste en la atención de recipientes de alimentos utilizados para alimentar animales.

Todos los recipientes tienen que ser llenados de alimento cada vez que se vacíen.

Cada jaula puede tener un número distinto de animales, por lo tanto, la frecuencia de llenado será distinta para cada jaula; de esta manera, los recipientes correspondientes a las jaulas con mayor cantidad de animales tienen mayor prioridad al momento de ser atendidas.

Físicamente las jaulas se encuentran situadas cercanas y es posible utilizar un elemento que pueda transportar el alimento desde un depósito hasta los recipientes.

Hay corriente eléctrica permanente.

Todo el escenario se encuentra situado a la intemperie.

El tiempo máximo que un recipiente puede estar sin alimento es tres minutos.

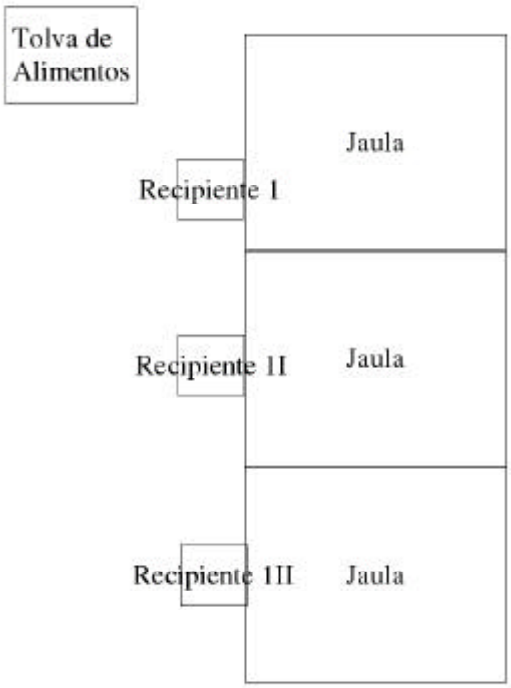

Fig. 3 – Distribución Física Actual

4.1.1 Análisis de las actividades que se desarrollan en la realidad.

A fin de lograr un mejor entendimiento de las tareas necesarias para el correcto funcionamiento de la situación actual es necesario descomponer el escenario en grupos de tareas.

De esta manera se identifican tres grupos de tareas:

1. Estado de los Recipientes.
2. Priorización y Ordenamiento de los recipientes a ser atendidos.
3. Llenado de los recipientes.

Si bien puede llegar a idenficarse una secuencia en la realización de las tareas, es posible realizarlas en forma paralela, ya que mientras se está llenando recipientes se puede vaciar otro recipiente que también debe ser llenado.

En la figura 4 (DFD) se pueden identificar los datos que son utilizados por las distintas tareas. Estos datos corresponden al estado en el que se encuentran los recipientes y donde se encuentra el transporte de alimento.

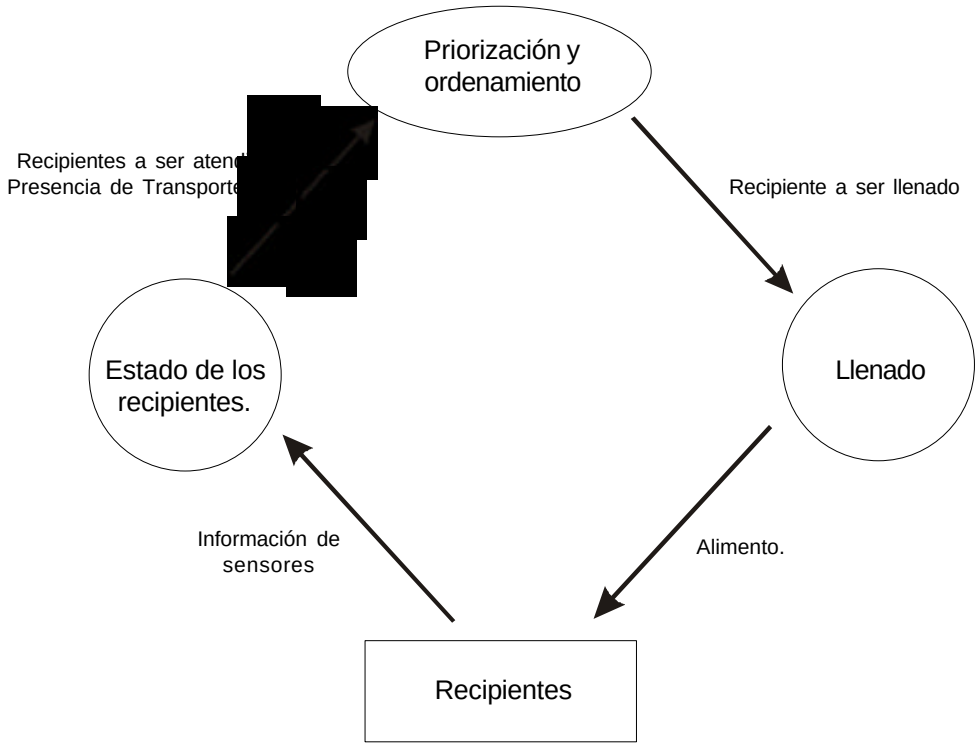

Fig. 4 – Actividades e información necesaria.

4.1.2.Descripción de las Actividades.

1. Estado de los Recipientes

La función principal de este actividad es definir la situación en que se encuentran los recipientes a fin de identificar los que necesitan ser llenados.

Las prioridades de cada recipiente se asignan en base a la cantidad de consumo de alimento que tengan los animales.

Pasos:

- Obtención de Recipientes sin alimentos.
- Verificación si ya se encuentra en la cola de recipientes a ser atendidos.
- Asignación de la prioridad.

2. Priorización y Ordenamiento de los recipientes a ser atendidos

Con la situación del escenario capturada se inicia la segunda etapa, que tiene como objetivo analizar los recipientes a vacíos junto con sus prioridades y actualizar la cola de recipientes a ser atendidos..

Pasos :

- Agregado a la cola de recipientes a ser atendidos.
- Ordenamiento de la cola según las prioridades.

3. Llenado de los recipientes.

Una vez determinado qué recipiente debe ser atendido, es necesario enviar el transporte para que pueda llenar nuevamente el recipiente.

Pasos:

- Envío de alimento.
- Actualización de cola de recipientes con carencia de alimento.

4.2. Diseño del sistema a desarrollar.

En la presente sección se encontrará la documentación asociada a la etapa de diseño del sistema a desarrollar. La información presentada consistirá en los aspectos más abstractos ya que en la sección de Implementación se encontrarán descripciones detalladas sobre cada uno de los aspectos del sistema desarrollado.

Basándose en los requisitos a cumplir por el sistema, se determinó que estará compuesto por una parte de software y otra de hardware que permite la interacción con el escenario. El trabajo conjunto permitirá que, mediante la utilización de sensores y motores, sea posible determinar de carencia de alimento, la identificación del recipiente a llenar y posteriormente el llenado del mismo sin la intervención del hombre. Quedando como responsabilidad del hombre el llenado de la tolva que contenga alimento para ser repartido en los recipientes.

Según los grupos de actividades definidos en la sección anterior, no es posible desarrollar módulos que realicen las tareas en forma aislada. Con la intención de facilitar el entendimiento, se utilizarán gráficos para los aspectos de hardware y luego se desarrollará los componentes de software y la interacción con el hardware.

4.2.1. Hardware:

La siguiente figura describe cómo será la conexión de las entradas de datos de los sensores de carencia de alimento. Es su función indicar cuando haya carencia de alimento en el recipiente.

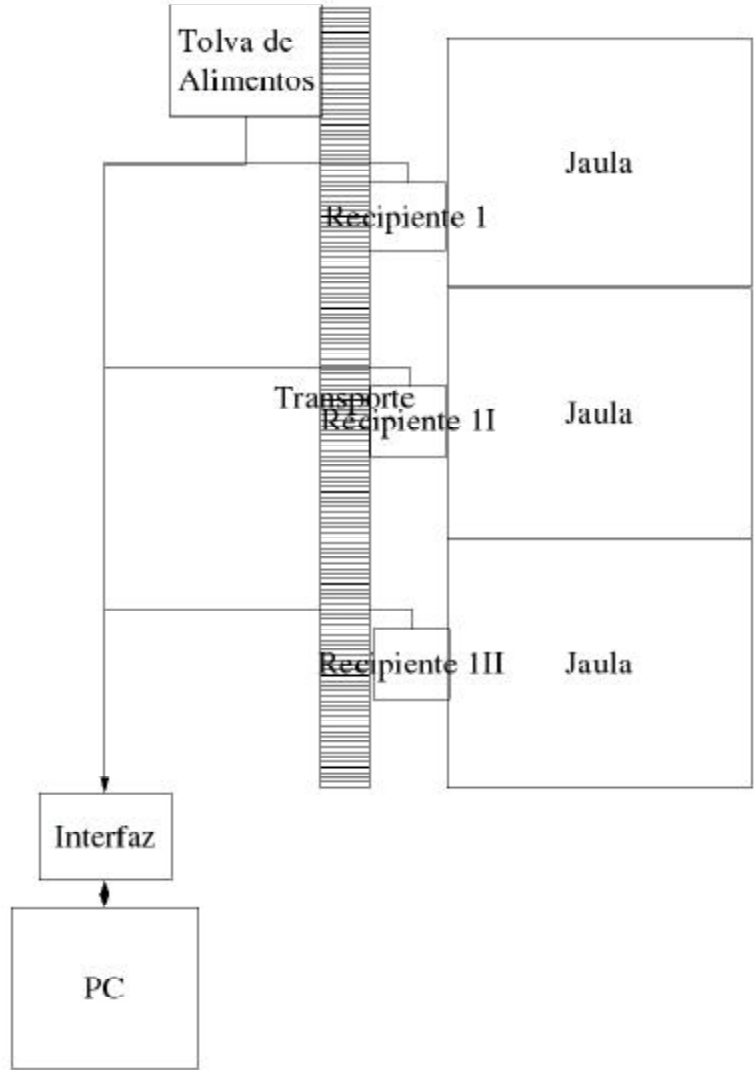

Fig. 5 – Entrada de Datos de carencia de alimento en el recipiente.

A su vez, también será necesario utilizar sensores que indiquen la presencia del transporte de alimento. En la siguiente figura se grafica cómo se vincularán estos sensores con la PC.

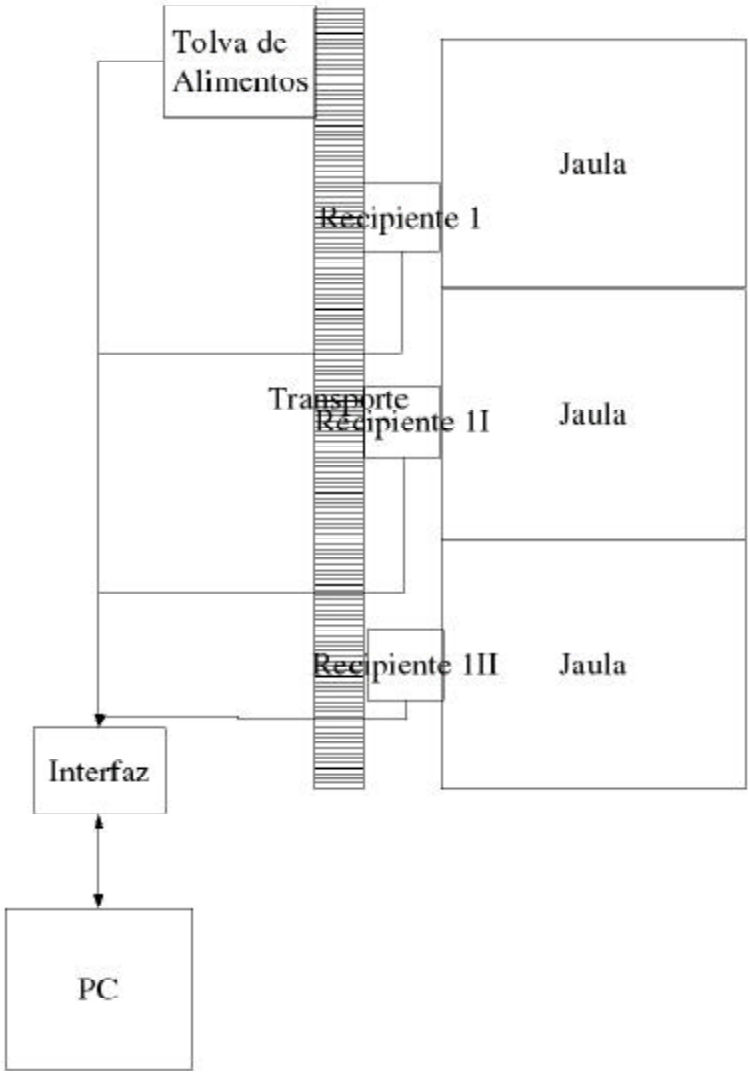

Fig. 6 - Entrada de Datos de presencia del transporte.

La interfaz se conectará a través del puerto paralelo de la PC y tendrá las siguientes funciones:

- expandir la cantidad de entradas del puerto paralelo.
- Reflejar los valores de los sensores para ser leídos desde la PC.
- Facilitar el manejo de los motores del transporte.

Para hacer llegar el alimento a los recipientes se utilizará un transporte que se vinculará a la computadora de la siguiente manera:

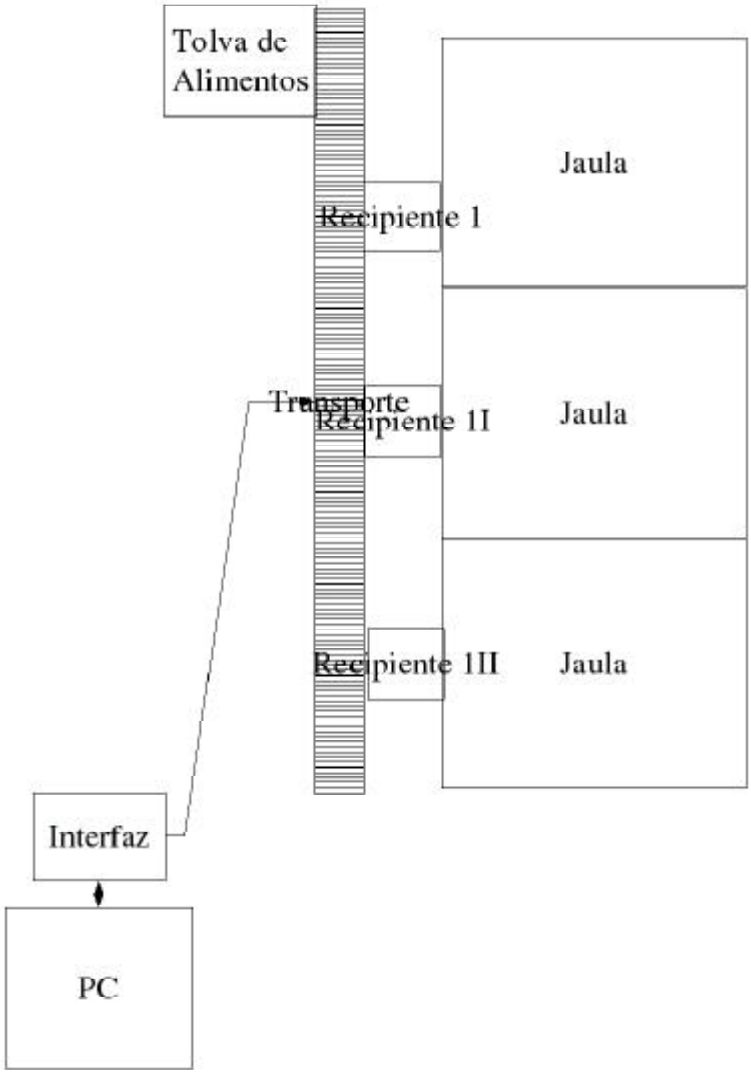

Fig. 7 – Salida de datos para controlar los motores

4.2.2. Software:

En este componente si se puede descomponer según las actividades que se enunciaron en la sección de análisis. Pero durante su funcionamiento no se ejecutarán en forma secuencial, sino que se ejecutarán concurrentemente para poder cumplir con los requisitos de los Sistemas de Tiempo Real.

A continuación se enumeran los procesos necesarios para alcanzar el objetivo:

1-1 Captura de estado de recipientes.
1-2 Captura de presencia de transporte.
2 Análisis y procesamiento.
2-1 Inserción de recipientes vacíos.
2-2 Ordenamiento de la cola de recipientes vacíos.
3 Atención del Recipiente.
3-1 Identificación del recipiente a ser atendido y actualización de datos.
3-2 Movimiento del transporte.
3-3 actualización de datos.

Para el funcionamiento de estas funciones es necesaria la existencia de la siguiente información:

Vector de estados de recipientes.
Vector de prioridades de recipientes.
Cola de recipientes vacíos.
Indicador de presencia de transporte.
Identificador del recipiente atendido.
Estado de ejecución del programa.

Debido a los requisitos del escenario, será necesario que las tareas se desarrollen en forma paralela ya que es imprescindible mantener actualizado en tiempo real los valores que representan elementos de la realidad. Se determinó que lo más apropiado sería la utilización de threads[1]. Los threads permiten que se ejecuten funciones que cumplan con distintos objetivos pero que compartan información. En el caso que se desarrolla en este trabajo, hay funciones que cumplen la función de proveedor de datos y otras que utilizan esos datos.[2] Esto hace necesario que el acceso a repositorios de datos sea controlado mediante la utilización de otras variables (Mutex) para lograr la consistencia en los datos utilizados.

Descripción de las funciones:

1.1 Captura de estado de recipientes.

El objetivo de esta función consiste en la obtención de los estados de los sensores de carencia de alimento de los recipientes. Para ello, accede a los valores de los sensores a través de la interfaz y actualiza el vector de estados de los recipientes.

Devuelve un vector con los estados de cada recipiente.

1.2 Captura de presencia de transporte.

El objetivo de esta función es identificar en qué recipiente se encuentra el transporte. Para poder cumplirlo, necesita obtener los estados de los sensores de presencia a través de la interfaz.

Devuelve una variable con el identificador del recipiente donde se encuentre el transporte. En caso de no encontrarse en ningún recipiente devuelve -1.

2.1 Inserción de recipientes vacíos.

Es su objetivo insertar los recipientes con carencia de alimento en la cola de recipientes vacíos. Utiliza el vector actualizado por la función 1.1 y verifica si ya están presentes en la cola de recipientes vacíos. A su vez, cuando los inserta en la cola, les agrega la prioridad correspondiente según el vector de prioridades.

Devuelve el la cola de recipientes vacíos actualizada.

2.2 Ordenamiento de la cola.

Esta función debe mantener ordenada la cola de recipientes vacíos según las prioridades de los recipientes. Durante el ordenamiento las prioridades pueden ser actualizadas, ya que cada vez que un recipiente es relegado de su posición, se incrementar una unidad en su prioridad.

Devuelve la cola de recipientes vacíos ordenada por las prioridades de los recipientes.

3.1 Identificación del recipiente a ser atendido y actualización de datos.

Para poder enviar el alimento primero tiene que definir a qué recipiente y una vez definido, aumentar la prioridad para que no sea desplazado y colocarlo en la variable de recipientes en atención.

Devuelve la cola de recipientes actualizada y la variable de recipiente en atención con el identificador del recipiente que se encontraba primero en la cola de vacíos.

3.2. Movimiento del transporte.

Una vez que la variable que contiene el recipiente a ser llenado posee un valor válido, se procede a movilizar el transporte hasta que llegue al recipiente correcto y lo llene. Una vez finalizada dicha tarea, el transporte regresa a su posición inicial.

Mientras que el transporte esté en movimiento no se elimina el recipiente de la cola de recipientes vacíos, sino que se incrementa su prioridad a un valor máximo. Esto se realiza a fin de evitar que se agregue nuevamente en la cola de vacíos o que sea desplazado del primer lugar.

3.3. Actualización de datos.

Luego de completada la fase de atención del recipiente se actualizan todos los repositorios de datos para continuar con la atención de próximos recipientes.

4.3. Implementación

4.3.1. Breve Descripción:

Este sistema es un STR que tiene como objetivo controlar el estado de tres recipientes de comida, determinar la carencia de alimento y llenar el recipiente vacío. Está compuesto por sensores, cuatro utilizados para determinar la presencia de alimento y otros cuatro para poder definir la ubicación del transporte de alimento. Una placa interfaz a la cuál se conectan los sensores, la alimentación eléctrica y el cable paralelo del puerto paralelo de la PC.

---

1. Hilos de procesos.
2. Se puede asociar con el problema del productor-consumidor analizado en la mayoría de los libros de sistemas de tiempo real.

4.3.2. Requisitos:

Ha sido desarrollado para ser ejecutado en un entorno operativo SUSE Linux 8.2, ya que para dicho entorno existen librerías que permiten la ejecución en forma paralela de procesos respetando los estándares POSIX[3].

Por lo tanto los requisitos para utilizar el sistema son los mismos que para el mencionado entorno operativo.[4]

Deben estar instaladas las librerías pthreads y PARAPIN.

Es necesaria tener libre una conexión de la fuente de alimentación para proveer 12 V de corriente continua a la placa.

4.3.3. Características del Producto:

El trabajo ha sido desarrolada en el lenguaje C++ utilizando librerías para threads y Parapin para utilizar el puerto paralelo.

4.3.4. Composición del Trabajo:

Como ya se ha definido anteriormente está compuesto por dos partes, una de hardware y una software. La primera es necesaria para interactuar con el entorno y la segunda para evaluar la situación y responder en forma adecuada.

Descripción del Componente de Hardware

Al momento de desarrollar el sistema se detectó la necesidad de incrementar la cantidad de entradas a través del puerto paralelo. Ya que el mencionado puerto provee 5 entradas, de las cuáles se utilizan 4 y para poder obtener la información necesaria para el correcto funcionamiento de la solución son necesarias 8 entradas como mínimo.[5]

Para ello se desarrolló una interfaz que permitiera expandir la cantidad de entradas de datos. A su vez, se integró a la mencionada interfaz los componentes necesarios para controlar motores paso a paso que serán utilizados para desplazar el transporte, tal como se explicará más adelante.

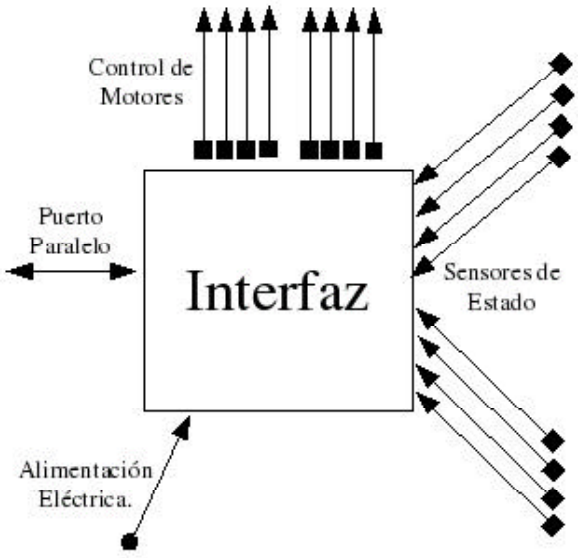

Fig. 8 – Características de la Interfaz

A continuación se explicará en forma individual cada una de las partes ya que para un mejor entendimiento, es conveniente.

Sensores de Estado:

Se utilizaron de dos tipos, unos denominados pulsadores y otros microswitches. Los primeros fueron colocados en los recipientes para indicar cuando es necesario su llenado. Los microswitches se utilizaron para poder determinar la presencia del transporte.

3. Para mayor información sobre los estandares posix remitirse al anexo POSIX.
4. Consultar anexo SUSE82 para mayor detalle.
5. Para más información sobre cómo incrementar las entradas de puerto paralelo, remitirse al anexo EXPANSION PUERTO PARALELO.

Pulsadores:

Se utilizaron pulsadores normales abierto modelo B163A, que se posicionaron en los recipientes para detectar la presencia de alimento.

Fig. 9 – Pulsador B163A

Microswitches:

Para detectar la presencia del transporte se utilizaron microswitches normales abiertos modelo B175FP. Estos se colocaron de manera tal que al paso del transporte se activaran.

Fig. 10 – Microswitch B175FP

Alimentación Eléctrica:

La interfaz es alimentada con 12 V. provenientes de la fuente de alimentación de la PC, como así la masa es conectada directamente a la masa de la misma PC.

Control de Motores:

Los ocho cables son utilizados para controlar dos motores paso a paso. Uno de los motores tiene la función de mover el transporte desde la posición inicial hasta el recipiente a ser atendido y luego regresarlo a la posición. El segundo motor tiene como objetivo volcar el contenido del transporte en el recipiente.

Puerto Paralelo:

Mediante esta conexión es posible el envío de datos y recepción de datos desde la PC. Se utiliza un conector DB25, de los cuales se utilizan todos los cables denominados de datos, 4 de entrada o estado, uno de control y todos los correspondientes a la masa del puerto paralelo.

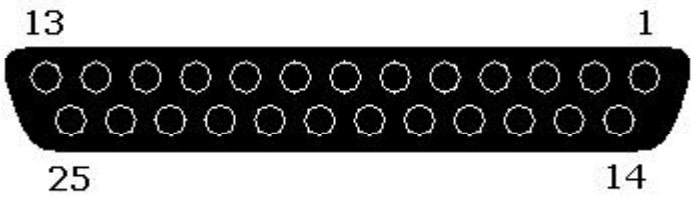

Fig. 11 – Conector DB25.

Interfaz:

Está compuesta por dos partes, una correspondiente a la captura de datos y otra al control de los motores.

Para la captura de datos es necesaria la expansión de entradas del puerto paralelo, ello se logra mediante la utilización de un circuito integrado denominado 74ls157[6]. Este circuito permite multiplexar la información presente en 8 sensores y enviarla mediante 4 conectores que transmiten su contenido al puerto paralelo de la PC.

Mediante la aplicación de intensidad en una de sus patitas, es posible cambiar entre los primeros 4 y los últimos 4 sensores conectados al integrado. Para controlar el valor de esta patita es utilizado el pin 16 del conector DB25 del puerto paralelo.

6. Para más información sobre este circuito remitirse al anexo 74LS157

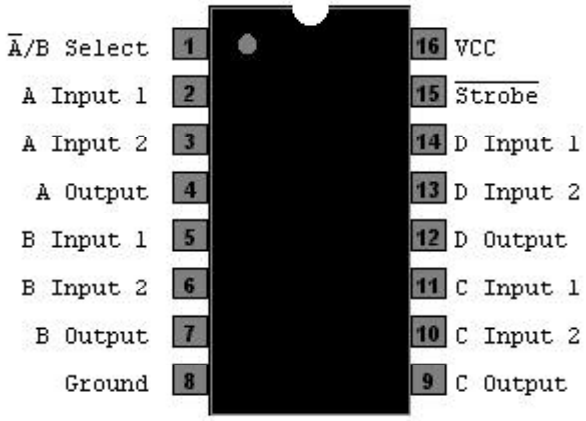

Fig. 12 – Circuito 74LS157.

Para visualizar el correcto funcionamiento fueron utilizados LEDs que se conectaron a los distintos conectores.

A continuación se incluye una imagen capturada de la herramienta CircuitMaker que facilita el diseño de circuitos electrónicos.

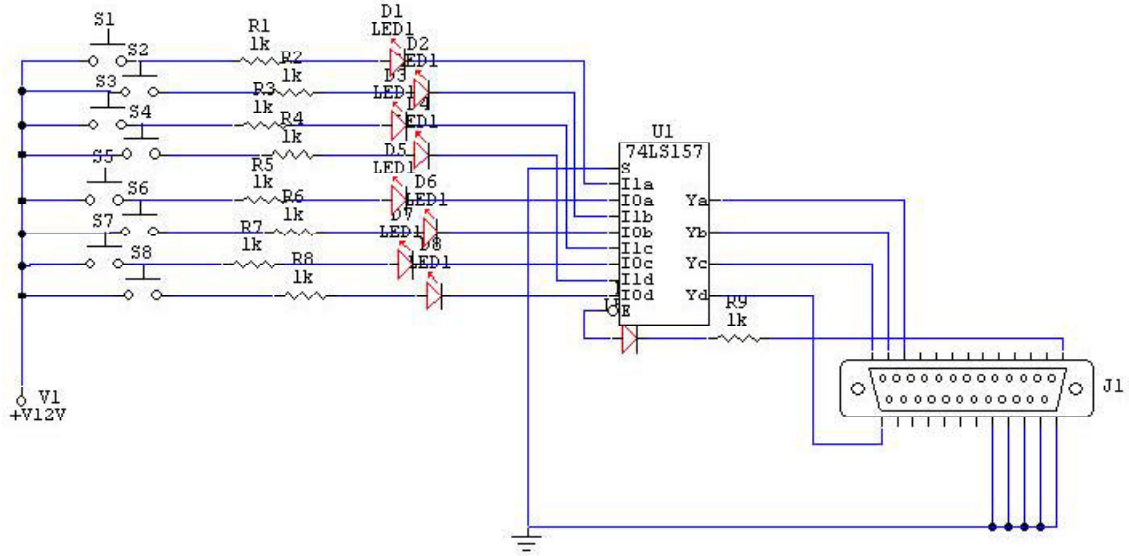

Fig. 13 – Circuito de entrada.

Para utilizar los motores paso a paso se utilizó un circuito integrado (ULN2803[7]) cuya función es invertir el valor de sus entradas y reflejarlas en sus salidas.

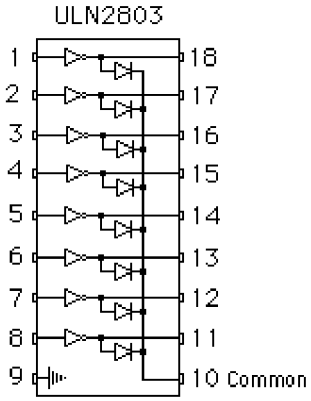

Fig. 14 – Circuito ULN2803.

7. Para mayor información remitirse al anexo ULN2803.

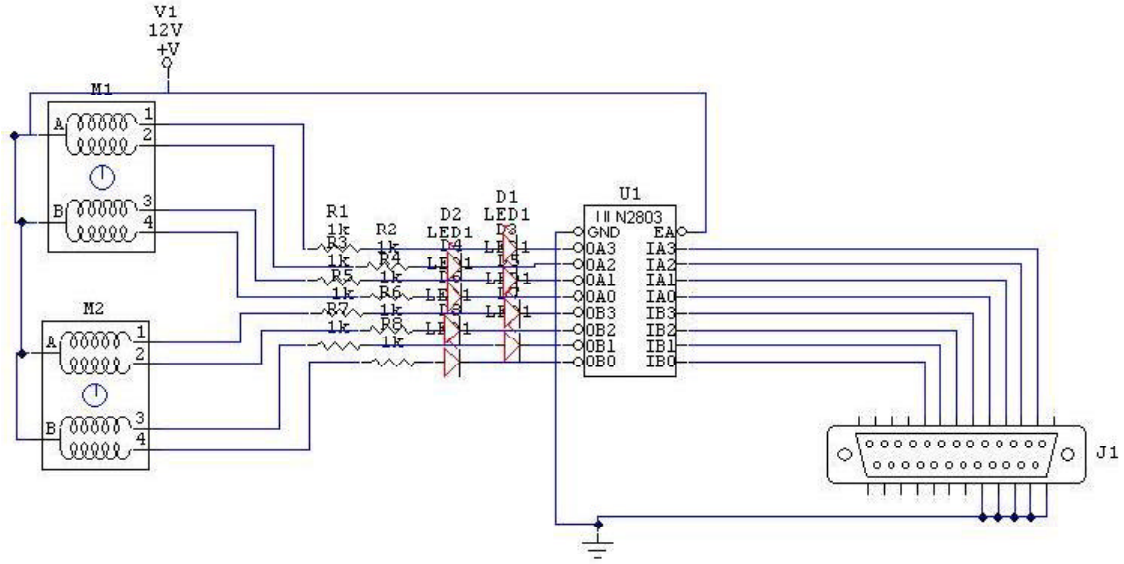

Fig. 2.5 Circuito de Salida para controlar motores.

Descripción del Componente de Software:

El objetivo del componente consiste en identificar los recipientes que carezcan de alimento, determinar cuál debe ser atendido y controlar los motores a fin de hacer llegar el alimento a los recipientes.

La comunicación con la interfaz se realiza a través del puerto paralelo, para utilizarlo se utilizan las librerías PARAPIN[8]. Estas librerías permiten la utilización de cada pin del puerto paralelo en forma independiente y a su vez definir qué tipo de función tendrá. (entrada / salida)

Como ya se ha explicitado en la etapa de diseño, se definen distintas funciones para satisfacer los distintos objetivos que componen el objetivo general.

A continuación se describirán las funciones que componen el sistema.

1.1 Captura de estado de recipientes.

Para poder obtener los valores de los sensores de alimento, es necesario poner en 0 el bit correspondiente al pin 16. Esto permite que los valores de los sensores de alimento sean reflejados en los pines definidos como entrada del puerto paralelo.

Una vez obtenidos los valores, será actualizado el vector de estados de los recipientes.

1.2 Captura de presencia de transporte.

En este caso, es necesario poner en valor 1 el bit correspondiente al pin 16.

2.1 Inserción de recipientes vacíos.

Son insertados los recipientes que presenten carencia de alimento.

La evaluación se realiza recorriendo el vector de estado de recipientes.

Se recorre la cola hasta encontrar una posición libre, en donde se inserta el recipiente que se está evaluando.

2.2 Ordenamiento de la cola de recipientes vacíos.

La cola es recorrida tantas veces como posiciones tenga. Es utilizado el método de burbuja para realizar el ordenamiento.

La prioridad del elemento desplazado es incrementada en una unidad.

Se utiliza la última posición como variable temporal para lograr el desplazamiento.

3. Atención del recipiente.

Tiene como objetivo llenar el recipiente correcto y para lograrlo está compuesto por subfunciones que se desarrollan a continuación:

8. Para ver las características principales consultar el anexo PARAPIN.

3.1 Identificación del recipiente a ser atendido y actualización de datos

Se incrementa la prioridad del primer recipiente de la cola de vacíos y se lo coloca en la variable de recipiente en atención.

3.2. Movimiento del transporte

Se envía el transporte al recipiente atendido, para lograr estacionarse en el recipiente correcto se utilizar el valor de la variable posición comparándolo con el recipiente que se encuentra en la variable de recipiente atendido.

3.3. Actualización de Datos

Una vez finalizada la atención del recipiente, se lo elimina de la cola de vacíos.

4.4. Funcionamiento.

Para utilizar el sistema es necesario verificar que se encuentren conectados todos los componentes del mismo:

- Sensores de estado de recipientes.
- Sensores de presencia del transporte.
- Cable del transporte.
- Cable de Alimentación de la placa.
- Cable del puerto paralelo a la placa.

Una vez que se haya corroborado la parte física del sistema es necesario ingresar como súper usuario (root), ubicarse en el directorio donde se encuentre el sistema y ejecutar el programa ./tfc_final. Una vez inicializado el programa, será necesario determinar un estado de ejecución. Los mismos pueden ser:

0. Sale del Sistema.
1. Ejecución Normal.
2. Ejecución monitorizada, muestra constantemente los valores de todas las variables del sistema.
3. Ejecución a definir, es una estado de ejecución que está disponible para futuras opciones (Interfaz Gráfica.).

Una vez seleccionado el modo de ejecución comienza el funcionamiento del sistema, igualmente en cualquier momento es posible cambiar el estado de ejecución ingresando el número correspondiente.

Para visualizar el código fuente, remitirse al anexo CODIGO FUENTE.

5. Conclusión

Luego de haber desarrollado el STR para mantener con alimento tres recipientes he podido identificar distintos aspectos que deben ser tenidos en cuenta al momento de diseñar e implementar sistemas de tiempo real. A diferencia de los demás sistemas, los STR hacen necesario incorporar el límite temporal al momento de definir cada aspecto del diseño. Límite que generalmente nunca es tenido en cuenta al momento de diseñar la mayoría de los sistemas informáticos.

Este condicionamiento trae asociado incrementar el conocimiento sobre cada uno de los aspectos que estén vinculados al STR en desarrollo, con el fin de lograr determinar los tiempos utilizados para cada uno de los procesos asociados.

Igualmente, al haber desarrollado el STR para mantener con alimento a los recipientes considero que si bien requiere gran conocimiento del entorno, se puede lograr generalizar interacciones con el entorno, que deriven en la definición de procesos específicos, con el fin de lograr STR de aplicación amplia. Por lo menos para problemas de una complejidad baja pero que pueden cubrir gran cantidad de las situaciones de distintos entornos.

Por lo tanto, creo que la base para continuar la expansión de los sistemas de tiempo real se encuentra en la estandarización de los procedimientos de análisis y desarrollo ya que hay estándares relacionados con los aspectos de implementación, por lo menos para sistemas de tiempo real de características similares el realizado en este trabajo. Por ejemplo, los estándares POSIX para los sistemas operativos con sus especificaciones para los sistemas de tiempo real.

De esta manera considero este trabajo como un acercamiento que permite la iniciación en el desarrollo de sistemas de tiempo real.

Glosario

C++: Fue desarrollada por Bjarme Stroustrup en los Bell Laboratories a principios de la década de los 80. C++ introduce la programación orientada al objeto en C. Es un lenguaje extremadamente poderoso y eficiente. C++ es un super conjunto de C, para aprender C++ significa aprender todo de C, luego aprender programación orientada al objeto y el uso de éstas con C++.
PARAPIN: librería que permite la utilización de los pines del puerto paralelo. Ver Anexo PARAPIN.
POSIX: Ver anexo POSIX.
S.T.R: Sistema de Tiempo Real.
Threads: Hilos de Ejecución. Ver anexo Pthreads.

Bibliografía

[GREHAN-MOOTE-CYLIAX98] « Real-Time Programming» . Rick Grehan, Robert Moote y Ingo Cyliax. Addison Wesley. Noviembre 1998.
[BIRREL89] «An Introduction to Programming with Threads». Andrew Birrel. System Research Center. Enero 1989.
[MUÑOZ03] «Sistemas de Tiempo Real». Lic. Julio José Corvalán Muñoz. www.led.uc.edu.py. 2003
[WAINER97] «Sistemas de Tiempo Real». Gabriel A. Wainer. Nueva Librería. 1997
[FAQRTS03] «Comp.RealTime NewsGroup» http://www.realtime-info.be/encyc/publications/faq/rtfaq.htm#realtime_definition

Otras Fuentes consultadas:

National Instruments:
http://digital.ni.com/worldwide/latam.nsf/web/all/D67937341172615486256B60006665B0

Laboratorio de electrónica digital. UCA Paraguay.
http://www.led.uc.edu.py/str/revision.htm

Pagina Web Juan Carlos Galarza Roca.
http://www.angelfire.com/pa2/jcgr/tecnica/ES/expansion_a_8.htm

Página web ChipDocs.
http://www.chipdocs.com

Fabricante de Circuitos Integrados.
http://www.fairchildsemi.com

Paper I / O :
http://www.ece.cmu.edu/~koopman/des_s99/i_o/index.html

Paper Real-Time Systems:
http://www.ece.cmu.edu/~koopman/des_s99/real_time/index.html