PROTOCOLO AS-INTERFACE. IMPLEMENTACIÓN DE UN NODO MAESTRO EN PC.

DANNY ALBERTO GÜIZA GARCÍA
JOSÉ RODRIGO CONTRERAS CORONEL

UNIVERSIDAD INDUSTRIAL DE SANTANDER
Facultad de Ingenierías Físico Mecánicas
Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones
Bucaramanga
2007

PROTOCOLO AS-INTERFACE. IMPLEMENTACIÓN DE UN NODO MAESTRO EN PC.

DANNY ALBERTO GÜIZA GARCÍA
JOSÉ RODRIGO CONTRERAS CORONEL

Trabajo de grado para optar por el título de
Ingeniero Electrónico

Director: MPE. Jaime Guillermo Barrero Pérez
Codirector: MIE. Omar Leonardo Núñez Gualdrón

UNIVERSIDAD INDUSTRIAL DE SANTANDER
Facultad de Ingenierías Físico Mecánicas
Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones
Bucaramanga
2007

Dedicatoria

A DIOS, por iluminar mi camino y darme la sabiduría para vencer las adversidades.

A mi madre, Eddy, por depositar su confianza en mi, y brindarme el cariño y apoyo con el cual he logrado mis metas.

A mi hermana, Adriana, por estar siempre a mi lado, compartiendo los buenos y los malos momentos.

A mi abuela, Anita, por ser siempre cariñosa, infatigable y mi modelo a seguir.

A mi tío, Luis Francisco, por su gran amistad y respaldo incondicional.

A todos mis familiares y amigos, quienes con su ayuda y compañía, realizan un gran aporte a mi vida.

Danny Alberto Güiza García

Dedicatoria

A DIOS, todopoderoso, por darme la vida y la fortaleza necesaria para derribar todas las barreras que he encontrado a mi paso.

A mis padres, Rodrigo y Mariela, quienes con su amor y paciencia han sido gestores de la culminación de esta tesis.

A mis hermanos, Ludwing y Nelson, amigos inseparables, por brindarme su ejemplo y apoyo incondicional.

A mi novia Milena, por su amor, entrega, paciencia y motivación.

A todos mis familiares, quienes directa o indirectamente aportaron su grano de arena para la culminación de este nuevo proyecto.

A mis amigos, por su confianza, apoyo desinteresado y alegrías brindadas durante el tiempo compartido.

José Rodrigo Contreras Coronel

Agradecimientos

Los autores de este proyecto hacemos mención de aquellas personas que contribuyeron al desarrollo de este trabajo y por tal razón expresamos nuestro agradecimiento a:

Profesor Jaime Guillermo Barrero Pérez, Director del proyecto, por sus aportes, orientación y total confianza.

MIE. Omar Leonardo Núñez Gualdrón, Codirector del proyecto, por su gran apoyo, gran gestión y entera disposición para que esta meta se cumpliera a cabalidad.

Profesor José Alejandro Amaya Palacio, quien por su gran colaboración fue parte vital del desarrollo de este proyecto.

Profesor Jorge Enrique Meneses Flórez, por permitir el acceso a la red *AS-Interface* ubicada en Laboratorio de Mecatrónica de la escuela de Ingeniería Mecánica.

A los Ingenieros Electrónicos Arnol Marriaga, Oscar Abreu, Rigoberto Ortiz y Diego Orduz, quienes con el avance de sus proyectos de grado hicieron grandes aportes al desarrollo de este proyecto.

A todos nuestros amigos y compañeros, que hicieron parte de este gran proceso, por sus aportes y experiencias.

Al Sr. Michael Bryant, por su valiosa gestión, al proporcionar la versión actualizada de la norma IEC 62026-2 que rige el protocolo.

Y a todos aquellos que de una u otra manera nos brindaron su apoyo y ayuda para la culminación de esta tesis.

Índice general

Índice de figuras

Índice de tablas

Resumen

Titulo: PROTOCOLO AS-INTERFACE. IMPLEMENTACIÓN DE UN NODO MAESTRO EN PC.[1]

Autores: DANNY ALBERTO GÜIZA GARCÍA, JOSÉ RODRIGO CONTRERAS CORONEL[2]

Palabras Claves: AS-Interface, Automatización Industrial, Bus de Campo, Labview, Driver de Línea, Microcontrolador, Pirámide de Automatización, Redes Industriales.

Las redes de comunicación industrial han surgido debido a la necesidad de distribuir geográficamente los elementos de medida, actuadores y dispositivos de control que interactuan entre sí para llevar a cabo una actividad de automatización y control. En vista de los altos costos de algunos de los dispositivos empleados en los buses de campo (nombre común dado a las redes industriales) existentes en el mercado, surge la necesidad de desarrollar prototipos de dispositivos de comunicación que puedan garantizar la compatibilidad con soluciones comerciales.

En este proyecto de grado se presenta la implementación de un nodo maestro en PC, basado en el protocolo *AS-Interface*, mediante la programación en LabVIEW® de la máquina de estados, las tramas de datos y las funciones principales de un maestro estándar, así como el diseño de una interfaz *hardware* cuyo componente principal es un microcontrolador que permite el establecimiento y control del proceso de comunicación con una red *AS-i* comercial. Los circuitos de transmisión y recepción de datos incluídos en el diseño, garantizan la integridad física de las señales, según los requerimientos descritos en la norma *IEC 62026-2*. Este trabajo tiene como objeto suministrar una herramienta didáctica que permita el conocimiento a plenitud de una de las tecnologías más utilizadas actualmente en la automatización de procesos en el nivel sensor/actuador, *AS-Interface*.

[1] Trabajo de Grado

[2] Facultad de Ingenierías Físico-Mecánicas.Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones. Director: Jaime Guillermo Barrero Pérez. Codirector: Omar Leonardo Núñez Gualdrón.

Abstract

Title: AS-INTERFACE PROTOCOL. PC BASED MASTER NODE IMPLEMENTATION.[3]

Authors: DANNY ALBERTO GÜIZA GARCÍA, JOSÉ RODRIGO CONTRERAS CORONEL[4]

Keywords: AS-Interface, Industrial Automation, Fieldbus, Labview, Line Driver, Microcontroller, Automation Pyramid, Industrial Network.

Industrial communication networks have appeared due the need to distribute geographically the elements of measure, actuators and control devices that interact mutually for achieve an automation and control activity. In sight the high costs of some of the devices used in the fieldbus (common name given to the industrials networks) existent on the market, appear the need to develop prototypes of communication's devices that can guarantee the compatibility with commercial solutions.

In this research work show up the implementation of a PC master node based in *AS-interface* protocol, by means programming in LabVIEW® of the states machine, the data frames and the main functions of a standard master, as well as an interface's design hardware whose main component is a microcontroller that allows the establishment and control of the communication process with an *AS-i* commercial network. The data transmission and reception circuits included in the design, guarantee the physical integrity of the signals, according to the requests described in the standard IEC 62026-2. The work object is supply a didactic tool that enables the knowledge to plenitude one of the most used technologies at present in the automation processes in the sensor/actuador level, *AS-Interface*.

[3]Research work.

[4]Faculty of Physic-Mechanical Engineering, School of Electrical, Electronic and Telecommunications Engineering. Director: Jaime Guillermo Barrero Pérez. Codirector: Omar Leonardo Núñez Gualdrón.

Capítulo 1

INTRODUCCIÓN

La *Automatización Industrial* se ha convertido en una herramienta fundamental para mejorar el rendimiento y la eficiencia de los procesos productivos en la industria moderna y sus principales objetivos son la obtención de datos al instante, el control de los procesos y la actualización de la información en forma automática, permitiendo la toma de decisiones operacionales, tácticas y estratégicas más acertadas dentro de un proceso industrial.

Realizar la conexión física entre computadores usando tarjetas de interfaz y cables no asegura poder compartir información entres los computadores de la red. Para ejecutar este trabajo, la redes deben tener un juego de reglas o normas bien definidas, a estas normas se les conoce como protocolos, los cuales permiten el intercambio de información entre dos o más dispositivos. Los protocolos se encargan de especificar el formato del mensaje y las reglas necesarias para recibir e interpretar la información que se esta intercambiando. Asimismo desarrolla muchas funciones, entre estas están: definir una dirección única para cada equipo conectado a la red, determinar como son transmitidos los datos entres los dispositivos y procesar la información cuando es recibida.

Por otra parte, los protocolos utilizados en redes industriales cuentan con caracterísicas propias, tanto en el nivel físico como en el enlace de comunicación dependiendo del nivel en que estén ubicados en la jerarquía de automatización de procesos denominada *Pirámide de Automatización*, la cuál se explicará detalladamente en el capítulo 2. Algunos de los protocolos encontrados en la industria son: *Ethernet*, *Foundation Fieldbus*, *DeviceNet*, *ControlNet*, *Profibus*, *Modbus*, *Interbus*, *CAN*[1], *AS-Interface*, entre otros. En la figura 1.1 se muestra una gráfica comparativa de los buses de campo, cualificando características como la complejidad de su estructura, manejo de volumen

[1] *Control Area Network.*

de datos, funcionalidad y costo.

Fuente: *Ver referencia [1] p2. Modificado por los autores del proyecto.*

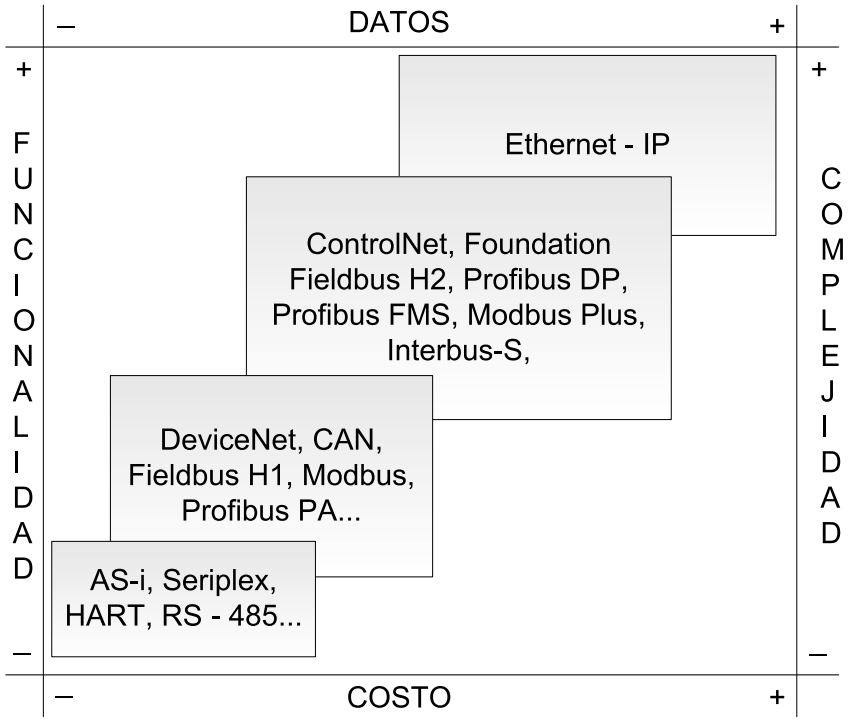


Figura 1.1: **Comparativo entre Buses de Campo.**

A nivel global el estudio de los *Protocolos de Comunicación Industrial* es considerado de gran valor en el proceso de asimilación de nuevas tecnologías de buses de campo, debido a que permite un entendimiento general del comportamiento de éstos dentro de un proceso; por ello, es vital la investigación de nuevos protocolos, haciendo énfasis en el desarrollo e implementación de diseños que adapten las arquitecturas existentes en el sector industrial.

Con este proyecto se pretende brindar un aporte académico en áreas como la *Automatización de Procesos* y la *Instrumentación Electrónica* dentro de la Universidad Industrial de Santander, haciendo uso de las tecnologías existentes en el campo de las Comunicaciones Industriales por medio de tesis de grado que a futuro permitan el estudio y la implementación de algunos protocolos en dispositivos programables cuya capacidad de integración con redes industriales comerciales permitan un entendimiento profundo de las tecnologías de buses de campo existentes en el mercado y las cuáles son utilizadas actualmente en la industria extranjera y local.

1.1. Antecedentes

En el estudio de Redes de Comunicación Industrial, existen algunas referencias de tesis de pregrado y monografías desarrolladas en la *Universidad Industrial de Santander* y específicamente en la *Escuela de Ingenierías Eléctrica Electrónica y Telecomunicaciones*, en las cuales se realizó el estudio e implementación de los protocolos *MODBUS* y *PROFIBUS*. Los trabajos encontrados son:

- Redes Industriales. Modbus y Ethernet implementados en autómatas programables TRILOGI, KOYO Y MODICOM-TELEMECANIQUE. Niño B. Carlos E. y Becerra A. Alvaro B. Tesis de Pregrado, 2002.
- Redes de Comunicaciones Industriales. Gelvez F. Julio A. Monografía Especialización en Telecomunicaciones, 2002.
- Modbus. Monitoreo de la Red empleando LabView. Carreño Sinle M. y Albarracin Pedro. Tesis de Pregrado, 2005.
- Modbus. Implementación de Procesador de Comunicaciones. Duque P. Jorge E. Tesis de Maestría, 2005.
- Modbus RTU. Implementación del Protocolo en Microcontrolador. Torres Roger. Tesis de Pregrado, 2006.
- Integración de Modbus a un sistema SCADA. Fajardo Diana. Tesis de Pregrado, 2006.
- Protocolo AS-interface: Implementación de Nodo Esclavo en un Microcontrolador. Marriaga C. Arnol R. y Abreu S. Oscar A. Tesis de Pregrado, 2006.
- Módulo de Integración de Sensores y Actuadores lógicos basado en el protocolo AS-I de Comunicaciones Industriales. Núñez G. Omar L. Tesis de Maestría, 2006.

A diferencia de algunos trabajos de grado en donde el principal objetivo es el de tomar referencias comerciales de dispositivos y analizar su configuración para su posterior puesta en marcha según los requerimientos del protocolo específico, cabe destacar que los trabajos desarrolados en la *Escuela de Ingenierías Eléctrica Electrónica y Telecomunicaciones* han sido los primeros pasos de un proceso investigativo en miras de lograr un conocimiento, análisis e implementación de sistemas embebidos de protocolos de comunicación a nivel industrial con características similares a prototipos

comerciales, siguiendo estándares internacionales.

Por otra parte la *Escuela de Ingeniería Mecánica* ha desarrollado algunos proyectos encaminados a la implementación y configuración de redes industriales comerciales los cuáles son mencionados a continuación:

- Redes Industriales de Comunicación basadas en Profibus. Puerto Jhon C. y Gómez José. Tesis de Pregrado, 1999.
- Redes de Comunicación Industrial basadas en Profibus y AS-interface. Díaz G. José N. y Martínez O. Edison. Tesis de Pregrado, 2002.
- BACnet. Protocolo de Comunicación. Archila D. Jhon F. y Barragán R. Humberto. Monografía Especialización, 2003.

A nivel nacional, es en el grupo de investigación en *Percepción y Sistemas Inteligentes* de la *Universidad del Valle*, donde se han desarrollado un gran número de trabajos relacionados con las redes de comunicación industrial, entre los cuales se destacan:

- Implementación de un esclavo con el protocolo *DeviceNet*. Arévalo B. William A. y Jaramillo T. Freddy E., 2003.
- Estudio e Implementación del Protocolo *HART* de Comunicaciones Industriales. Castillo Javier F. y Tenorio Jorge A., 2002.
- Implementación de una red *Modbus/TCP*. Ruiz O. Andrés F., 2001. Desarrollo de una red industrial con conectividad TCP/IP a través del protocolo Modbus/TCP usando el sistema embebido TINI de Dallas Semiconductor.
- Implementación de una red Inalámbrica usando el protocolo *Modbus*. Escobar Angélica M., Sánchez Hugo A., 2000.
- Interface *CAN-TII*, según el estándar *IEEE P 1451*. Agudelo Meyer A. y Fula O. Marco A., 1999.
- Driver de bus de campo *CAN* para una red de transductores inteligentes. Navarro S. Erika, 1999.
- Diseño de un procesador de aplicación con capacidad para manejo de una red *HART* basado en el estándar *IEEE P 1451*. Medina E. Carlos A. y Palacios R. Víctor J., 1998.

A nivel internacional se tiene referencia de la tesis:

- AS-I MASTER BUS CONTROLLER. Chan Poh L. Serena, Universidad de Queensland (Australia), 2003.

En éste trabajo se hizo el desarrollo parcial de un controlador maestro de bus de campo *AS-i* sobre un FPGA[2], implementando solamente una de las seis fases de la máquina de estados (fase de inclusión) que maneja el maestro y además no se realizó un adecuamiento físico de las señales que permitiera validar su funcionamiento en una red *AS-i* comercial.

1.2. Estructura del Documento

Este documento se encuentra organizado en 6 capítulos. En el capítulo 2 se presentan los conceptos básicos, relacionados con el estudio de las Comunicaciones Industriales en general.

El capítulo 3 presenta los conceptos que definen el protocolo *AS-Interface*, entre los cuáles se destacan las características físicas y lógicas, el proceso de comunicación, los tipos de transacciones, los requerimientos del receptor, el manejo de errores y las especificaciones más importantes del maestro, según los requerimientos de la norma *IEC 62026-2* [2].

En el capítulo 4 se muestra el diseño de la interfaz física del maestro, cuyo objeto principal es permitir la comunicación entre el PC y la red *AS-Interface*. En este se hace énfasis en el montaje de los circuitos de transmisión (*driver* de línea) y recepción (detección de pulsos) de datos y las consideraciones que se hicieron a la hora de escoger cada uno de los dispositivos utilizados para su funcionamiento.

El capítulo 5 presenta el desarrollo de los algoritmos implementados a nivel *software* tanto en el microcontrolador como en LabVIEW®, haciendo énfasis en la implementación de la máquina de estados del maestro, en la generación de patrones analógicos de corriente y la decodificación de pulsos detectados de la red.

El capítulo 6 se presentan los resultados y análisis del desempeño del maestro, además de registrar los valores experimentales para hacer una comparación con

[2] *Field-Programmable Gate Array*

aquellos valores estipulados en la norma.

Para finalizar, el capítulo 7 muestra las conclusiones sacadas de las pruebas hechas al maestro y las observaciones efectuadas a lo largo del desarrollo del proyecto. Asimismo, se presentan algunas sugerencias para futuros proyectos que sigan nuestra línea de investigación.

Capítulo 2

MARCO TEÓRICO

En este capítulo se exponen algunos conceptos y términos básicos, necesarios para poder hacer una apropiación certera sobre el tema que se desarrolla en el presente trabajo. Por esta razón se tratarán conceptos generales en el campo de las redes de comunicación, como el modelo de referencia OSI. Posteriormente se mencionarán temas específicos en el área de las Comunicaciones Industriales, haciendo énfasis en los buses de campo y el lugar que éstos ocupan en los niveles jerárquicos de un sistema de automatización industrial. Toda la teoría expuesta aquí tiene como objeto contextualizar el proyecto de grado para su mejor entendimiento.

2.1. Modelo de Referencia OSI

Al pensar en el intercambio digital de datos entre equipos a través de un sistema de bus, es muy importante definir la manera como se va a efectuar la transmisión de datos, el método de acceso y las informaciones relativas al establecimiento de los enlaces. Es por eso que la ISO (*International Standards Organization*) reglamentó el modelo de referencia OSI (*Open System Interconnection*), el cual define el comportamiento de un sistema de comunicaciónn a través de la descripciónn de siete capas independientes (ver tabla 2.1), con una separación estructurada que permite la modificación interna de cualquiera de estas, sin afectar el funcionamiento de las demas. Cada capa se fundamenta en la anterior, para extraer la información de control necesaria para su funcionamiento y enviar los datos a la siguiente, cumpliendo una serie de convenciones predefinidas que en conjunto conforman el protocolo [3] [1].

Este modelo es válido tanto para grandes flujos de información, como para aplicaciones más sencillas. Su implementación estricta depende del grado de complejidad de la aplicación y no se considera necesario para construir un sistema de comunicación.

Tabla 2.1: **Capas del modelo OSI.**

Capa	Identificación	Funciones
7	Aplicación	Especifica las funciones de usuario para el intercambio de variables, a través de servicios de comunicación específicos.
6	Presentación	Define la representación de los datos, la conversión del tipo de representación en un formato adecuado para el equipo y ejecuta acciones de diagnóstico.
5	Sesión	Realiza tareas de sincronización. Administra el establecimiento, disolución y vigilancia de una sesión.
4	Transporte	Realiza el establecimiento y/o disolución del enlace, y la formación, repetición y clasificación de paquetes. Revisa la integridad de los datos.
3	Red	Define el direccionamiento de paquetes y las rutas de comunicación. Realiza procesos de control de flujo.
2	Enlace de Datos	Define el método de acceso al medio, realiza la gestión de colisiones, limita los bloques de datos, y ejecuta algoritmos de detección y corrección de errores.
1	Física	Describe los niveles de voltaje y requerimientos de tiempo para la transmisión de *bits*. Realiza *Test* de errores a nivel de *bit*.

Por tal motivo en el campo de las comunicaciones industriales, la arquitectura OSI se simplifica al uso específico de tres capas [4], como se muestra en la figura 2.1:

- Nivel 1: Capa Física. Este nivel procura la transmisión transparente de *bits* a través del soporte físico, en el orden definido por el nivel de enlace. Define las características eléctricas y mecánicas (conectores o medios de enlace tipo *hardware*) de la línea de transmisión, las señales de control que determinan la temporización y el orden de transmisión, y realiza un diagnóstico de errores a nivel de *bit*. Entre los estándares más usados en este nivel, se encuentran las interfases RS-232 y RS-485.

- Nivel 2: Capa de Enlace de Datos. Este nivel tiene como función asegurar la transmisión de la cadena de *bits* entre los dos sistemas involucrados en el proceso de comunicación, apoyándose en un medio físico de conexión. Este nivel

Fuente: *Autores del proyecto.*

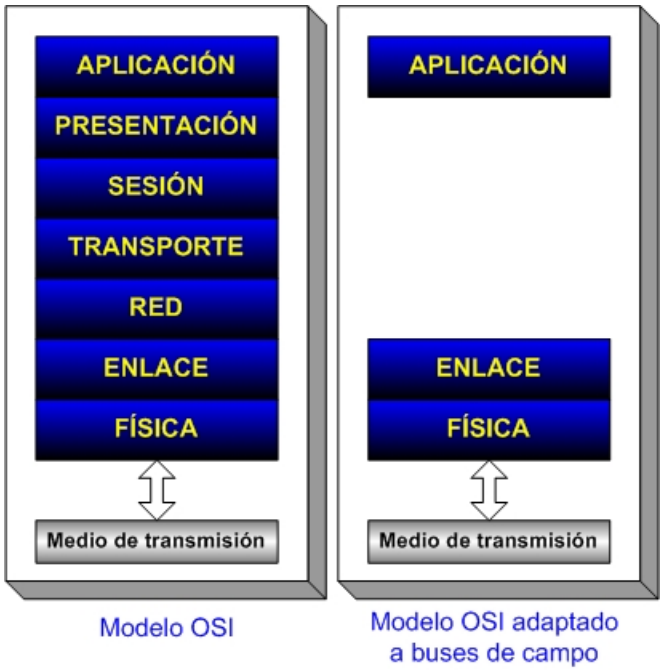


Figura 2.1: **Modelo OSI adaptado a buses de campo.**

es el encargado de formar las tramas de envío añadiendo datos de control y ejecutar los métodos de direccionamiento, detección y recuperación de errores, reenvío de tramas perdidas y regulación de tráfico. En redes locales procura el acceso exclusivo al medio definiendo los subniveles MAC (*Medium Access Control*) y LLC (*Logic Link Control*). Este ultimo aplica generalmente a la norma IEEE 802.2, aunque en sistemas de bus de campo se utilizan métodos de acceso considerablemente modificados debido a las características de tiempo real exigidas para dicha aplicación.

- Nivel 7: Capa de Aplicación. Este nivel se encarga de proporcionar un entorno que facilite el entendimiento entre usuarios de distintas máquinas digitales a nivel temático, sin importar medios, ni protocolos de comunicación. Comprende los servicios específicos de enlace con las diferentes aplicaciones de comunicación.

2.2. Comunicaciones Industriales

La estructura de un sistema de automatización emplea diversos equipos y protocolos de comunicación, aplicables de acuerdo a las necesidades del proceso industrial, lo

cual hace necesario la división o jerarquización de tareas en niveles caracterizados por parámetros tales como: recepción y transmisión de datos, acondicionamiento de señales y control de variables. Esta división es mostrada en la figura 2.2 y es conocida como *Pirámide de Automatización*. A continuación se hace una descripción de cada uno de los niveles.

Fuente: *Autores del proyecto.*

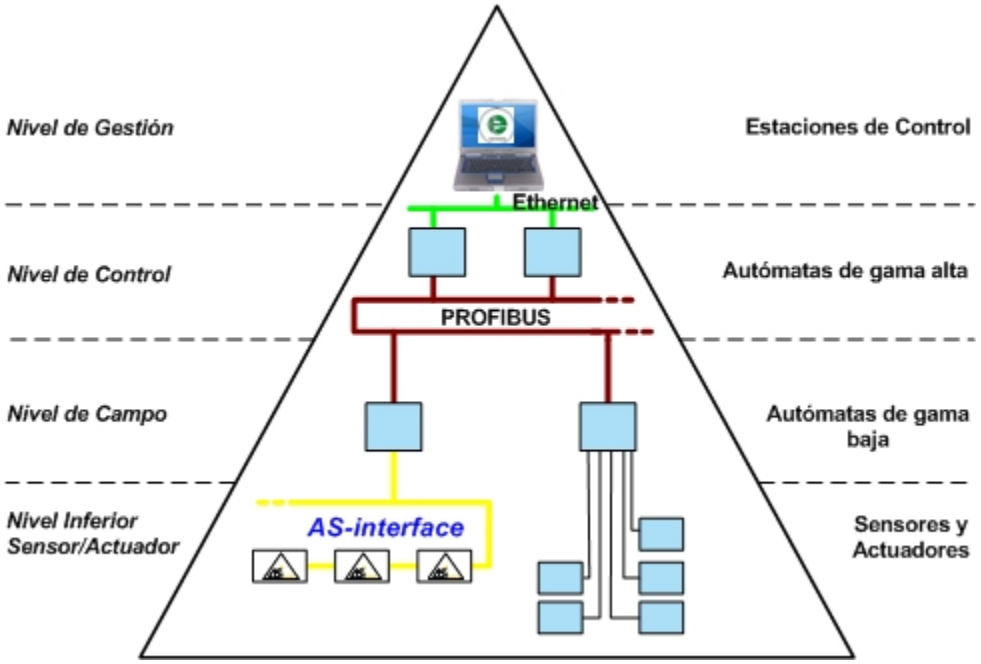


Figura 2.2: **Pirámide de automatización.**

- **Nivel Sensor/Actuador:** también llamado nivel de instrumentación, esta compuesto por dispositivos de medición (sensores) y de mando (actuadores). Es el nivel mas bajo de la pirámide y en el se encuentran los componentes directamente implicados en el proceso de producción, donde los sensores son los encargados de medir variables tales como temperatura, presión, nivel de líquidos; y los actuadores, los elementos que ejecutan órdenes provenientes de sistemas de control, siendo algunos ejemplos las bandas transportadores, los brazos mecánicos, las válvulas y los motores. En este nivel se encuentra el campo de aplicación del protocolo *AS-i*, demostrando un gran desempeño ante el manejo de las variables del proceso.

- **Nivel de Control (nivel de campo):** en este nivel se encuentran los elementos capaces de controlar los actuadores y sensores del nivel anterior. Estos dispositivos suelen ser autómatas programables o equipos de aplicación específica basados en

un microprocesador como los PLCs (Controladores Lógicos Programables) o los RTUs (Unidades terminales remotas). Los dispositivos de este nivel de control junto con los del nivel inferior actuador/sensor poseen autonomía suficiente para realizar procesos productivos por sí mismos, convirtiéndose en subredes o islas automatizadas. A este nivel pertenecen protocolos tales como *Profibus* y *Modbus*.

- **Nivel de Supervisión (nivel de planta):** en este nivel se sitúan autómatas de gama alta, computadores o sistemas de visualización, capaces de gestionar y supervisar todos los dispositivos de control existentes en la fábrica. Estos componentes son interconectados por medio de redes de tipo LAN *(Local Area Network)* y permiten la implementación de entornos SCADA, con los cuales se puede tener una imagen virtual de la planta, mediante el uso de pantallas de computador o visualizadores industriales, donde se muestren las posibles alarmas, fallos o alteraciones en cualquiera de los procesos que se llevan a cabo.

- **Nivel de Gestión (nivel de fábrica):** en este nivel no es relevante el control y supervisión de los procesos, siendo de gran importancia toda la información adquirida por medio de la gestión con niveles inferiores de una o varias plantas. Dicha información se refiere a las variables de producción, llevando a la realización de estadísticas acerca de los costos y tiempos de fabricación, calidad, estrategias de ventas, y en general, disponer de datos que permitan a los niveles directivos la toma de decisiones oportunas para la optimización en el funcionamiento de la planta. Las redes empleadas en el nivel son de tipo LAN y WAN *(Wide Area Network)* bajo protocolo *Ethernet*.

2.2.1. Bus de Campo

Un Bus de Campo es una red digital de comunicación serial, multipunto, bidireccional, compartida por diferentes elementos de campo (controladores, transductores, actuadores y sensores), que permite la transferencia de datos e información de control entre estos elementos primarios de automatización, control y monitoreo, con elementos de más alto nivel tales como los *DCS*[1] y los *SCADA*[2].

Con este tipo de buses se pretende bajar los costos de montaje y mantenimiento, facilitar la instalación de aplicaciones en tiempo real, permitir la transmisión serie sobre un bus digital de datos con capacidad de interconectar controladores con todo tipo de dispositivos de entrada-salida de sencillo manejo. Según la cantidad de datos

[1] *Distributed Control System*
[2] *Supervisory Control and Data Acquisition*

a transmitir se dividen en buses de alto nivel, buses de dispositivos (unos pocos bytes a transmitir) y buses actuador/sensor (se transmiten datos a nivel de bit), pero en ningún caso llegan a transmitir grandes bloques de información. La figura 2.3 muestra las bondades al implementar buses de campo en instalaciones industriales.

Fuente: *Ver referencia [5] p40. Modificado por los autores del proyecto.*

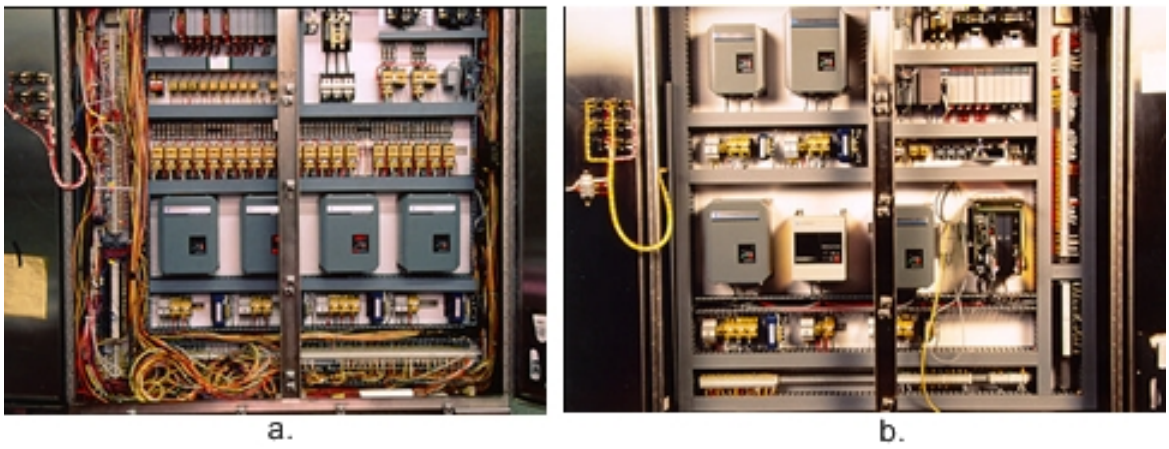


Figura 2.3: **Instalación industrial: a) sin usar buses de campo, b) con buses de campo.**

2.3. Comunicaciones por Líneas de Potencia (PLC)

En ésta sección se presenta algunas consideraciones importantes acerca de la tecnología PLC (*Power Line Communications*), la cual hace posible la transmisión de datos, voz y video sobre líneas de tensión, existente dentro una industria o domicilio. El actual estado de PLC permite la transmisión a velocidades de hasta 200 *Mbps*, lo que facilita la transformación de la red eléctrica en una autentica red de banda ancha, capaz de prestar gran cantidad de servicios ofrecidos por los operadores de telecomunicaciones a nivel global.

El uso del cableado eléctrico como soporte físico para la transmisión de información, además de energía, no es una idea nueva. Hasta hace poco tiempo la función de la tecnología PLC se había limitado al monitoreo y control de las líneas eléctricas y la transmisión de las lecturas de los contadores, es decir aplicaciones que no requerían un gran ancho de banda para su correcto funcionamiento. Durante finales de los años noventa los avances tecnológicos realizados permiten alcanzar velocidades de trans-

misión de Megabits, razón por la cual se plantea la posibilidad real de utilizar la red eléctrica como redes de acceso. Las ventajas que ofrece la tecnología PLC son las siguientes:

- Despliegue sencillo y rápido. El despliegue de la tecnología PLC es muy rápido y sencillo, porque utiliza infraestructura ya instalada (Los cables eléctricos).
- Servicio PLC desde diferentes lugares. La tecnología PLC permite conectarse a Internet y/o hablar por teléfono desde los enchufes eléctricos, ofreciendo la posibilidad de navegar y/o hablar de diferentes lugares de la empresa.
- Instalación simple y rápida. Instalación simple y rápida (solo es necesario conectar un MODEM PLC), y no requiere obras ni cableado.
- Multitud de nuevos servicios. Puede suministrar múltiples servicios con la misma plataforma tecnológica IP (un solo MODEM permite el acceso a Internet a alta velocidad y telefonía, así como diversos servicios a distancia como Demótica, TV interactiva, Teleseguridad, etc.).
- Conexión permanente. Proporciona una conexión a Internet permanente (las 24 horas del día) y sin interrupciones.

Con todo esto, las mayores ventajas de PLC apuntan a su disponibilidad mundial, efectividad del costo y facilidad de instalación.

La tecnología PLC, fué llevada al protocolo de comunicaciones *AS-interface* para hacer posible el envío de datos a la red por medio de la línea de tensión que alimenta los diversos componentes del bus de campo, permitiendo así una considerable disminución en el cableado, y por supuesto la instalación de la red y su posterior mantenimiento, de una forma más fácil y segura.

Capítulo 3

PROTOCOLO *AS-INTERFACE*

AS-interface[1] es un protocolo de comunicación industrial, que posee la ventaja de transmitir información sobre la misma línea de alimentación de la red (tecnología PLC[2]), permitiendo la comunicación con elementos tanto binarios como analógicos y la transferencia de datos y parámetros de forma serial, por medio de mensajes digitales de duración corta y fija. Este protocolo surgió hace aproximadamente 15 años con el fin de ofrecer soluciones de automatización con facilidades de implementación y mantenimiento a bajo costo, capacidad de detección de fallos y manejo de funciones de diagnóstico en aquellos sistemas que se encuentren en el nivel Sensor/Actuador donde normalmente se necesitan pocos bits de información para su funcionamiento. *AS-i* es un bus de campo definido por el estándar europeo EN50295 [6] y el estándar americano IEC 62026-2.

3.1. Características Generales

El protocolo *AS-i* es un bus de campo desarrollado inicialmente por *Siemens*, para la interconexión de actuadores y sensores binarios, pero la arquitectura se extendió al manejo de entradas/salidas analógicas. A nivel físico, la red puede adoptar cualquier tipo de topología, estructuras en bus, árbol o estrella. Las especificaciones del protocolo *AS-i* están abiertas a dominio público bajo la regulación de la IEC[3] en el estándar 62026-2, con la designación *Low-voltage switchgear and controlgear - Controller-device interfaces (CDIs) - Part 2: Actuator Sensor Interface (AS-i)* y actualmente es

[1] *Actuator Sensor Interface (AS-i)*

[2] *Power Line Communications.*

[3] *International Electrotechnical Commission.*

soportada por la organización independiente *AS-International Association*[4].

Entre las principales características de este sistema se encuentran:

- Velocidad de transmisión fija de 166,67 *Kbps*.
- Como medio físico de transmisión, emplea un cable bifilar sin apantallamiento, que permite tanto la alimentación del bus, como el transporte de datos. Presenta un sistema de conexión tipo vampiro (auto regenerador) que evita errores de polaridad ante la incorporación o eliminación de dispositivos.
- Para la transmisión de datos realiza procesos de codificación *Manchester II* y modulación *APM*[5].
- La longitud máxima de cada segmento es de 100 metros. Sin embargo se pueden instalar repetidores que permiten la unión de hasta tres segmentos (300 metros de longitud máxima).
- Permite la interconexión de un máximo de 31 esclavos, donde cada dispositivo habilita el manejo de 4 I/O digitales.
- En operación normal se establece un tiempo de ciclo máximo de $5ms$, para la consulta de los esclavos y añade dos ciclos extras para operaciones de administración del bus, como detección de fallos y control de pausas.

3.2. Interfases del protocolo *AS-i*

El estándar IEC 62026-2 especifica tres tipos de interfases que hacen parte del protocolo *AS-Interface*, las cuales se muestran en la figura 3.1 y se definen a continuación:

- **Interfaz 1.** En esta interfaz se conectan los esclavos con los actuadores, sensores y otros dispositivos y elementos que sean compatibles con AS-Interface. Además, en esta se puede verificar el comportamiento de las entradas y salidas los diferentes esclavos.
- **Interfaz 2.** Aquí se suministran todos los requerimientos físicos, lógicos y mecánicos para el intercambio de datos y la distribución de potencia hacia la red. Aquí se ubican las señales de información codificadas, las transacciones AS-i y la administración de potencia a la red.

[4] www.as-interface.net
[5] Alternate Pulse Modulation

Fuente: *Autores del proyecto.*

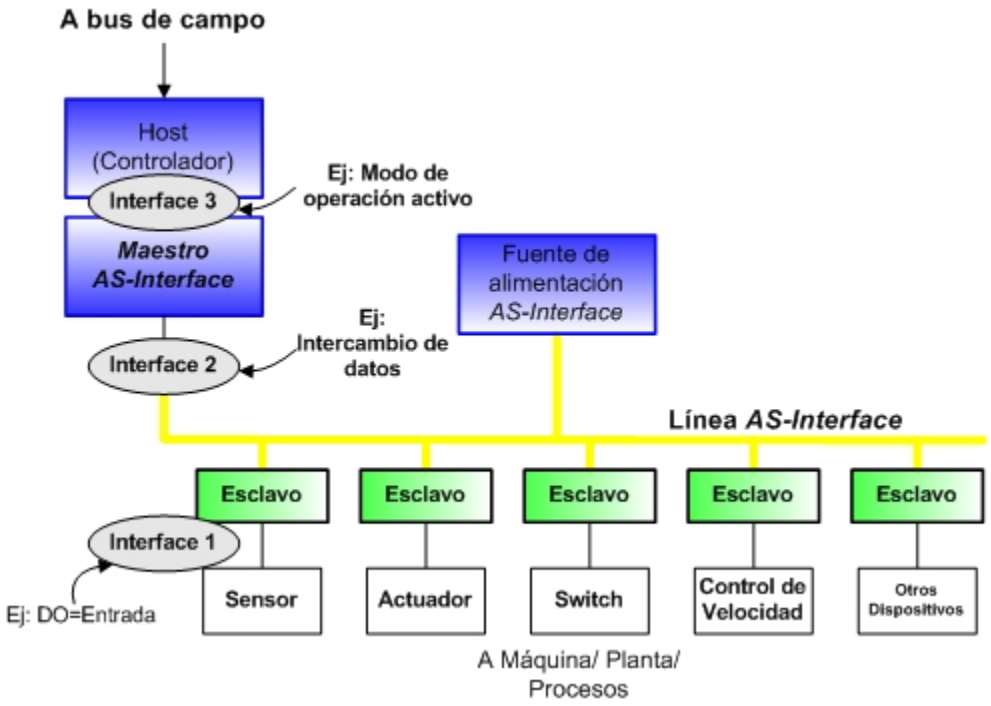


Figura 3.1: **Interfases del protocolo *AS-i*.**

- **Interfaz 3.** Esta es la interfaz entre el host (controlador) y el maestro AS-i, es decir, aquí se suministran todas las funciones usadas por el host para acceder al maestro y así efectuar el envío y recepción de datos cíclicos a y desde los esclavos.

3.3. Señal Física

La rata de transmisión de 166,67 *kbps* del bus *AS-i* define un tiempo de bit, $T_{BIT} = 6\mu s$, el cual no debe tener una desviación superior a $\pm 0\,.\,2\,\%$. Cada transacción se caracteriza por una petición de maestro y su respectiva respuesta de esclavo, con *bits* fijos de inicio y fin. Estos mensajes son codificados en formato *Manchester II*, donde un "0" lógico genera un flanco negativo de señal, un "1" lógico genera un flanco positivo de señal y el periodo de inactividad se toma como un nivel lógico alto. Esta señal codificada, es modulada con un esquema *APM*, representado por pulsos de corriente con amplitudes entre 55 mA y 68 mA, los cuales generan pulsos de voltaje tipo Sin^2 con amplitudes de $\pm 2V$ sobre el nivel de DC, dado el comportamiento inductivo de la línea y el circuito de desacople que ofrece la fuente AS-i. En la Figura 3.2 se muestran las señales típicas del sistema AS-i [2].

La forma de onda de la corriente puede ser representada determinísticamente de la

Fuente: *Ver referencia [1] p.22.Modificado por los autores del proyecto.*

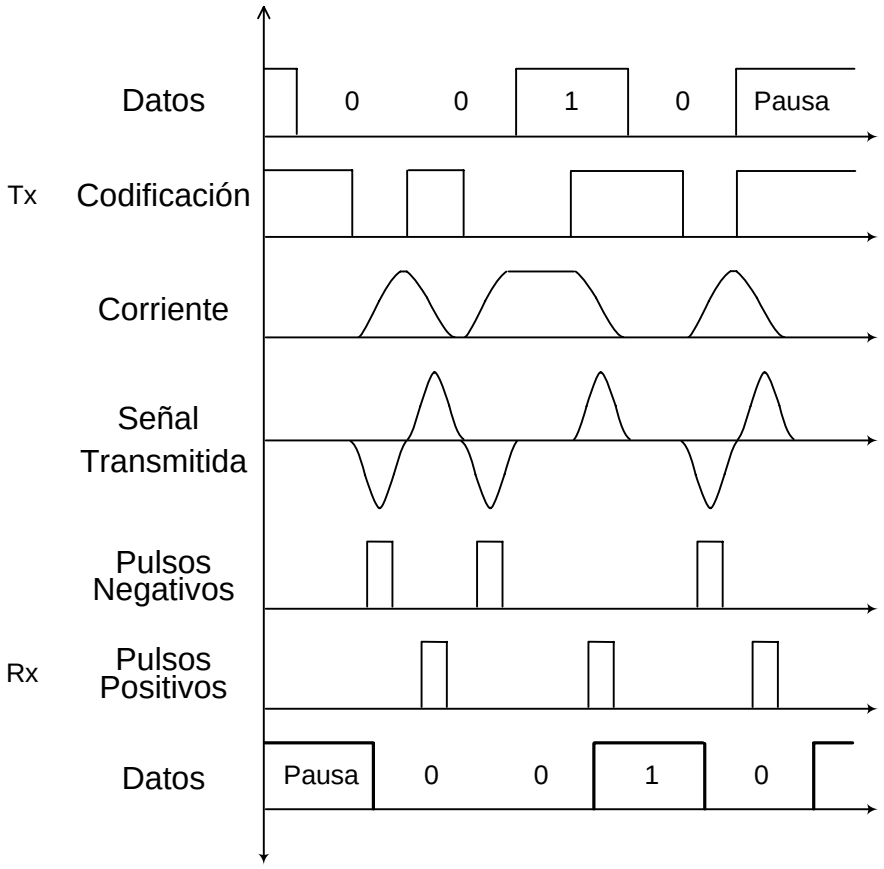


Figura 3.2: **Señal *AS-i.***

siguiente forma:

Durante un flanco negativo,

$$I_{AS-i}(t) = I_{send}\left[\frac{t}{3\mu s} - \frac{1}{2\pi}Sin\left(\frac{2\pi}{3\mu s}t\right)\right] \tag{3.1}$$

y durante un flanco positivo,

$$I_{AS-i}(t) = I_{send}\left[1 - \frac{t}{3\mu s} + \frac{1}{2\pi}Sin\left(\frac{2\pi}{3\mu s}t\right)\right] \tag{3.2}$$

Debido a las inductancias de desacople la forma de onda de la corriente genera un pulso de voltage negativo en cada flanco positivo y un pulso de voltage positivo en cada flanco negativo. Una expresión para describir el comportamiento ideal de esta señal de voltage en el bus AS-i es:

$$V_{AS-i}(t) = \pm V_{send}\left[Sin^2\left(\frac{2\pi}{6\mu s}t\right)\right] \tag{3.3}$$

3.4. Proceso de Comunicación

AS-Interface es un sistema de comunicación *Maestro-Esclavo* que está compuesto por un maestro simple y por un máximo de 31 esclavos (62 con direccionamiento extendido) que poseen una dirección única en el rango de 1 a 31 ($1A/1B$ *a* $31A/31B$ en modo extendido). Esta dirección es llamada *dirección de operación* y es almacenada en memoria no-volátil, permitiendo que sólo aquellos esclavos con una dirección de operación específica sean capaces de responder ante las peticiones del maestro. La dirección "*cero*" es utilizada durante el cambio de una dirección de esclavo y esta es almacenada en memoria volátil.

El intercambio de datos entre el maestro y el total de los esclavos en la red *AS-i* es efectuado empleando un esquema *Cyclic Polling*[6] como el mostrado en la figura 3.3. En éste se puede observar que una transacción inicia con una petición del maestro, quien espera una respuesta desde el esclavo dentro de un rango de tiempo definido; si el maestro no recibe una respuesta válida desde el esclavo dentro de este tiempo, entonces interpretará esto como una respuesta negativa. El maestro puede retransmitir la petición una vez más y luego de recibir una respuesta válida, iniciando así la siguiente transacción después de que la pausa de envío ha transcurrido.

Fuente: *Ver referencia [2] p.24. Modificado por los autores del proyecto.*

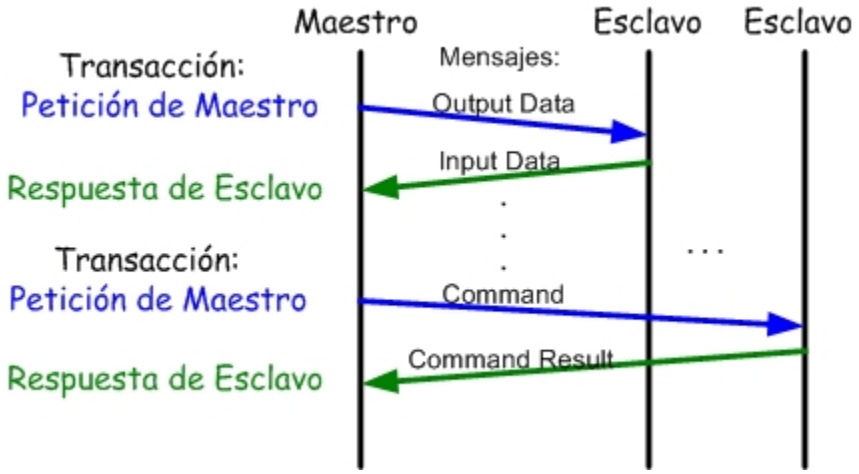


Figura 3.3: **Proceso de encuestamiento cíclico.**

Todos los tiempos que se especifican en esta sección son tomados de las señales sobre la línea *AS-Interface* localizadas sobre las terminales del maestro [2]. Un ejemplo de estos tiempos es enseñado en la figura 3.4.

[6] Encuestamiento cíclico

Fuente: *Autores del proyecto.*

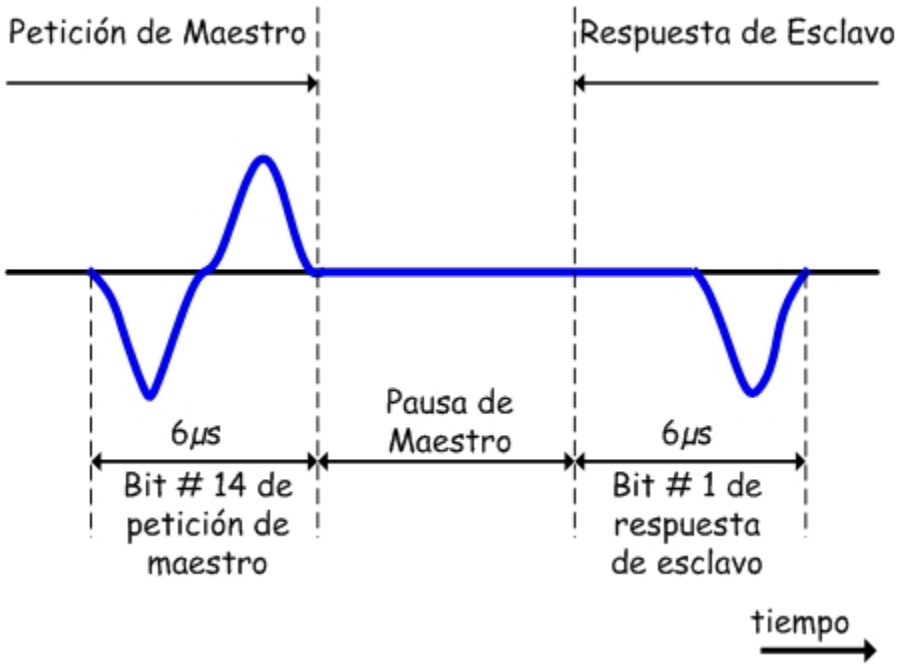


Figura 3.4: **Medida de la pausa del maestro.**

La petición del maestro y la petición del esclavo inician con un "*cero*" como primer bit. Una transacción está dividida en dos acciones (petición del maestro y respuesta del esclavo) y dos intervalos de tiempo (pausa de maestro y pausa enviada) como se observa en la figura 3.5. A continuación se hace una breve descripción de los tiempos de una transacción *AS-Interface*, los cuáles fueron tomados en cuenta para la implementación del maestro:

- *Master Request*: Envío de un mensaje desde el maestro hacia un esclavo.
- *Master Pause*: Durante este tiempo el esclavo procesa la función pedida, genera la respuesta respectiva e inicia este tiempo para enviarsela al maestro. Un esclavo comenzará su respuesta dentro de un periodo de tiempo entre $2T_{BIT}$ y $5T_{BIT}$, después del fin de la petición de maestro. De igual forma el maestro será capaz de aceptar el comienzo de una respuesta de esclavo dentro de un periodo de tiempo entre $12\mu s$ y $63\mu s$ despues del fin de su petición.
- *Slave Response*: Envío de los datos del esclavo al maestro.
- *Slave Pause*: Déspues de recibida la respuesta del esclavo, habrá un periodo mínimo durante el cual no ocurrirá la siguiente transmisión. Esta pausa es equivalente

Fuente: *Ver referencia [2] p.24. Modificado por los autores del proyecto.*

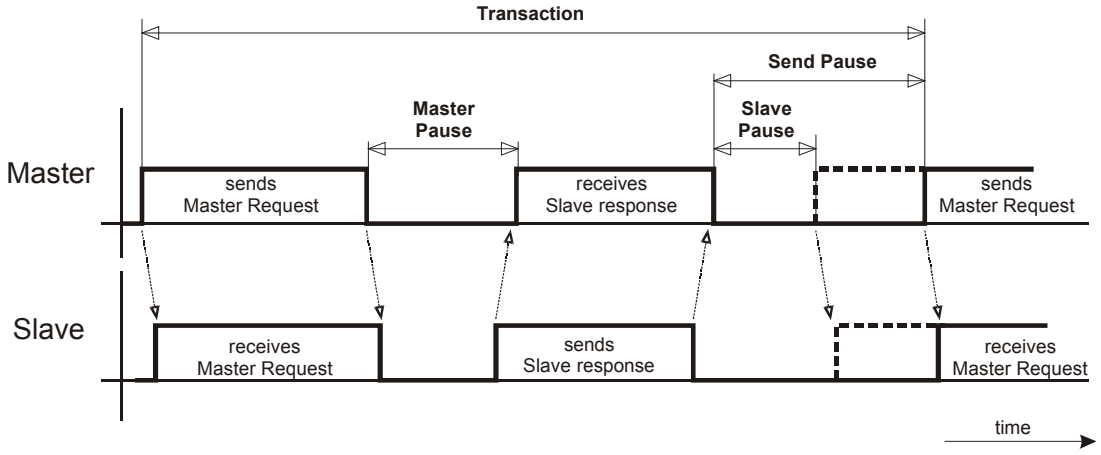


Figura 3.5: **Transacción *AS-i*.**

a un periodo de tiempo entre $1.5T_{BIT}$ y $2T_{BIT}$. El esclavo será capaz de aceptar el comienzo de una nueva petición de maestro déspues de una pausa de $6\mu s$.

- *Send Pause*: Después de recibida la respuesta del esclavo habrá un periodo mínimo durante el cual no ocurrirá la siguiente transmisión. En operación normal el tiempo de esta pausa será de un *Slave Pause* en caso de tener más de 30 transacciones por ciclo. En el caso de tener 30 o menos transacciones, el *Send Pause* puede ser prolongado a un tiempo máximo de $500\mu s$.

- *Slave Response Time-out*: En caso de no recibir respuesta del esclavo en este intervalo de tiempo, el maestro finalizará la transmisión o la reenviará. Este periodo corresponde a la ventana de tiempo de $11T_{BIT}{}^{+0\mu s}_{-3\mu s}$.

Para llevar a cabo este proceso de comunicación el maestro elabora una petición de 14-*Bits* y el esclavo una respuesta de 7-*Bits*, de longitud corta y tamaño constante, tal como se muestra en la figura 3.6. Estos datagramas poseen los campos definidos en la tabla 3.1.

Fuente: *Autores del proyecto.*

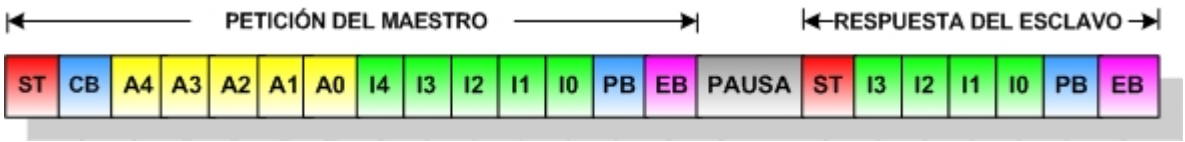


Figura 3.6: **Estructura del datagrama *AS-i***

Tabla 3.1: **Descripción de los campos del datagrama *AS-i*.**

Bit	Descripción
ST	*Bit* de inicio de datagrama. Fijo a "0".
CB	*Bit* de Control. 0 = Transmisión de Datos/Parámetros. 1 = Transmisión de Comandos.
A4 - A0	Dirección del Esclavo.
I4	*Bit* de Información. 0 = Petición de Datos. 1 = Petición de Parámetros.
I3 - I0	*Bits* de Información.
PB	*Bit* de Paridad. Paridad Par.
EB	*Bit* de fin de datagrama. Fijo a "1".

3.5. Tipos de Transacciones

El sistema *AS-i* esta en capacidad de procesar dos tipos fundamentales de transacciones; las transacciones simples o sencillas y las transacciones combinadas [2].

1. ***Transacción Simple***. En la tabla 3.2 se muestran los cuatro tipos de transacciones simples, las cuales definen las funciones de intercambio de datos, parametrización, administración de la red y diagnóstico.

Tabla 3.2: **Tipos de transacciones *AS-i* simples.**

Tipo	Descripción
Data Exchange	Entrega y/o recibe los patrones de *bits* hacia/desde los puertos de salida/entrada del dispositivo esclavo.
Write Parameter	Entrega los patrones de *bits* a los puertos de parámetros del dispositivo esclavo.
Addres Assignment	Asignación de dirección no-volátil (1...31) a un dispositivo esclavo para reemplazar la dirección zero.
Commands	*Read_ identification_ code* *Reserve_ R1* *Read_ I/O_ configuration* *Read_ status* *Reset_ slave Delete_ address*

2. ***Transacción Combinada***. Transacciones definidas para la transferencia de más de 4-*Bits* netos de información, estableciendo un conjunto adicional de reglas.

Para identificar esclavos que son capaces de realizar transacciones combinadas se utilizan los *ID-Codes*[7] o *Extended ID-Codes*. En la tabla 3.3 se presentan las diferentes alternativas disponibles en el estándar para la transferencia de valores y funciones especiales.

Tabla 3.3: **Tipos de transacciones *AS-i* combinadas.**

Tipo	Esclavo	Maestro	Descripción
I	S-7.3	M.3	Entradas - Salidas de 16-*Bits*.
I	S-7.4	M.3	Dispositivos de campo complejos.
II	S-7.5.5	M.4	Combinación de dispositivos de campo.
II	S-B.A.5	M.4	Comunicación serial en dispositivo de campo.
III	S-7.A.7	M.4	4I/4O en modo de direccionamiento extendido.
III	S-7.A.A	M.4	8I/8O en modo de direccionamiento extendido.
IV	S-7.A.8	M.4	Entradas de 16-*Bits* en direccionamiento extendido.
IV	S-7.A.9	M.4	Entradas duales de 16-*Bits* en modo extendido.
V	S-6.0	M.4	Entradas - Salidas de 16-*Bits* de alta velocidad.
Safety	S-7.B	–	Cualquier esclavo de entada para trabajo seguro.

- Tipo I: Este tipo de transacción está relacionada con la transmisión de variables analógicas.
- Tipo II: Este tipo de transacción usa el mecanismo de transferencia de datos de *AS-i* para construir un canal de comunicación serial *full duplex* para la transferencia de *bits* entre el maestro y el esclavo. Esta puede ser usada para en el manejo de sensores y actuadores analógicos, dispositivos de campo con parámetros variables y el reemplazo de interfases de $4 - 20mA$.
- Tipo III: Este tipo de transacción usa el mecanismo de transferencia de datos de *AS-i* para construir canales de comunicación *full duplex* de *4-Bits* u *8-Bits* entre el maestro y el esclavo, en el modo de direccionamiento extendido. Esta transacción es usada específicamente en el manejo de teclados, torres visualizadoras de señal, válvulas terminales y sensores/actuadores de *8-Bits*.
- Tipo IV: Este tipo de transacción usa el mecanismo de transferencia de datos de AS-i para construir un canal de comunicación sencillo o dual de *16-Bits* para

[7] Código de identificación del esclavo

transferencia de datos desde el esclavo al maestro en el modo de direccionamiento extendido. Este tipo es montado para el uso de sensores de *16-Bits* de canal sencillo o dual y el reemplazo de interfases de $4 - 20mA$.

- Tipo V: Este tipo de transacción usa el mecanismo de transferencia de datos de AS-i para construir canales de comunicación *full duplex* rápida, de *8-Bits*, *12-Bits* o *16-Bits* para transferencia de datos desde el maestro al esclavo usando 2, 3 o 4 direcciones estándar. Este tipo es montado para el uso de sensores/actuadores de *16-Bits* y alta velocidad, y el reemplazo de interfases de $4 - 20mA$ por lazos de control.

3.6. Requerimientos del Receptor

En un sistema *AS-i* real tanto las formas de onda como las amplitudes descritas en la sección 3.4, estarán influenciadas por las propiedades físicas de la línea. Por tal motivo los receptores deben ser capaces de detectar pulsos dispersos como los mostrados en la figura 3.7, los cuales deben cumplir con las siguientes características mínimas:

Fuente: *Ver referencia [7] p.19. modificado por los autores del proyecto.*

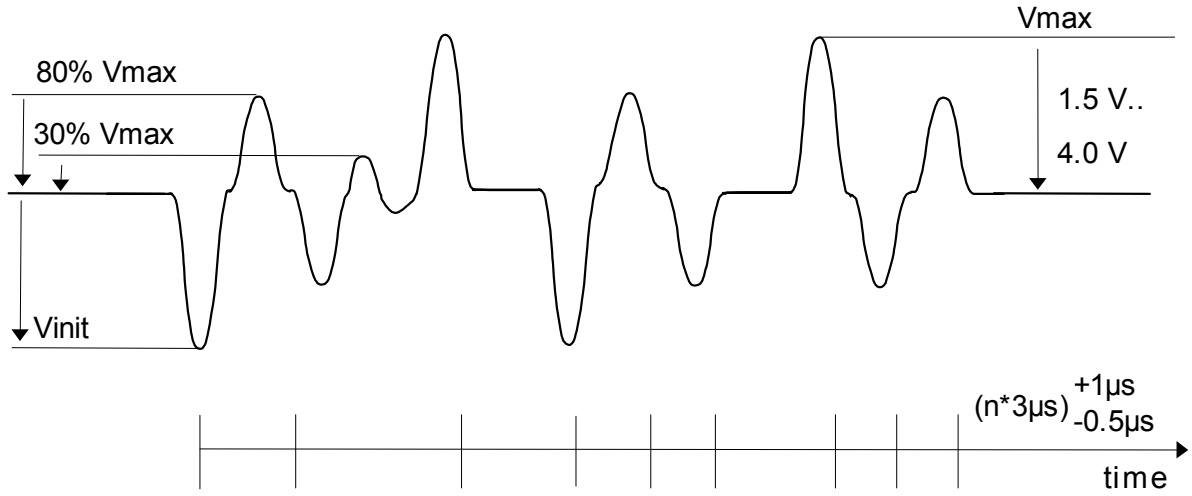


Figura 3.7: **Requerimientos del Receptor *AS-i*.**

- La máxima amplitud del pulso V_{max} del mensaje puede variar entre $1.5\ V_{pico}$ y $4\ V_{pico}$. Las diferencias de amplitud entre peticiones consecutivas del maestro no variarán para una misma configuración, es decir, los valores mostrados en la figura 3.7 representan el peor caso en el cual se realicen diferentes configuraciones

y ubicaciones de las estaciones esclavas sobre la línea. Finalmente, para una configuración uniforme, la máxima variación de V_{max} entre dos respuestas de esclavo ubicados en diferentes posiciones de la línea será en una escala de $1:1.5$.

- La amplitud de un pulso válido dentro de un mensaje puede variar desde el 65 % al 100 % de la amplitud máxima V_{max}. De acuerdo a versiones previas de la norma el rango de variación puede ser desde el 80 % al 100 %.
- Los pulsos validos que deben ser aceptados por el receptor comienzan en una ventana de tiempo de $(3\mu s \cdot n)^{+1.0\mu s}_{-0.5\mu s}$ en relación al pulso inicial V_{init}.
- Los pulsos fuera de una ventana de tiempo de $(3\mu s \cdot n)^{+1.6\mu s}_{-0.8\mu s}$ deben ser rechazados por el receptor. Estas desviaciones pueden ocurrir debido a la combinación de diferentes efectos como cargas capacitivas sobre la línea, o desviaciones en la frecuencia del oscilador del transmisor o del receptor.
- Los pulsos de ruido (noise) o rizado (ringing) de hasta un 30 % de V_{max} no deben perturbar la recepción del mensaje.
- La máxima desviación del tiempo nominal de *bit* $(3\mu s)$ debe ser menor o igual a ± 0.1 % para el maestro y menor o igual a ± 0.2 % para el esclavo.

3.7. Detección de Errores

La detección de errores es un proceso necesario para ignorar cualquier datagrama que haya sido deteriorado por efectos del ruido sobre la línea. Cualquier petición del maestro[8] o respuesta de esclavo[9] sobre la línea AS-i será sometido a un chequeo por parte del receptor, ante la posible ocurrencia de algunos de los siguientes errores de transmisión [2]:

- *Start_ bit_ error*: se produce cuando el pulso inicial que sigue a una pausa no es de polaridad negativa. Este *bit* es la referencia del proceso de decodificación.
- *Alternating_ error*: se genera cuando dos pulsos consecutivos no son de polaridad diferente.
- *No_ information_ error*: Los pulsos de cualquier petición o respuesta deben ser detectados dentro de una ventana de tiempo de $(3\mu s \cdot n)^{+1.0\mu s}_{-0.5\mu s}$ después del pulso inicial, donde n=26 para el maestro y n=12 para el esclavo.

[8] *master request.*
[9] *slave response.*

- *Parity_ error*: se produce cuando la suma de todos los *bits* de información de un datagrama no cumple con el tipo de paridad par estipulado por la norma.
- *End_ bit_ error*: se genera cuando el pulso final del datagrama no es de polaridad positiva. Este *bit* corresponde al tiempo ($6\mu s \cdot n$), donde n=13 para el maestro y n=6 par el esclavo.
- *Length_ error*: error producido cuando se detecta algún tipo de señal durante los periodos de pausa.

3.8. Maestro *AS-Interface*

Esta sección describe los requerimientos generales de un maestro *AS-Interface*, teniendo en cuenta el perfil de Maestro referenciado en [8], el cual clarifica los modos de funcionamiento disponibles dependiendo del número de esclavos conectados a la red los cuáles pueden ser 31 (modo estándar) o 64 (modo extendido) [2]. Para este proyecto fué definido el modo estándar, debido a que la red *AS-i* suministrada no posee más de 5 esclavos.

El maestro *AS-i* es el encargado de recoger los datos de la red y de enviarlos al controlador correspondiente, y viceversa. Asimismo organiza el tráfico de datos y, en caso necesario, coloca los datos de los sensores y actuadores a disposición del *host*[10] o de un sistema de bus superior (por ejemplo, *Profibus*), a través de pasarelas *DP/AS-Interface*[11]. Aparte de la correspondiente consulta sobre el estado de las señales de los esclavos, el maestro también es capaz de transmitir parámetros de configuración a los esclavos, o supervisar la red constantemente y suministrar datos de diagnóstico.

El maestro tiene una estructura de capas compuesta de tres partes funcionales como la mostrada en la figura 3.8, las cuáles son definidas a continuación:

Fuente: *Autores del proyecto.*

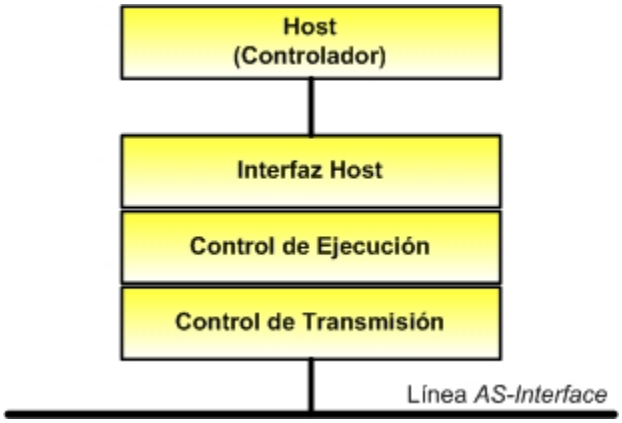


Figura 3.8: **Estructura de capas de un maestro *AS-Interface*.**

- ***Interfaz Host:*** actúa como la interfaz lógica entre el maestro y *host*. El *host* utiliza las funciones suministradas por esta interfaz para comunicarse con el sistema *AS-Interface*.

[10] Control Programable y Automático de Procesos
[11] visite *www.as-interface.net*

- ***Control de Ejecución:*** responsable del inicio del maestro y del sistema *AS-Interface*, del intercambio de datos cíclico, de los comandos y de las funciones de control de ejecución.
- ***Control de Transmisión:*** responsable de la transmisión física de las peticiones del maestro así como de la retransmisión automática de peticiones en caso de fallas.

3.8.1. Listas Locales para el Control de Ejecución

El *Control de Ejecución* requiere de algunas listas para desempeñar la comunicación entre el maestro y el esclavo y para suministrar medios de diagnóstico al sistema. Los datos usados para describir la configuración deseada de *AS-Interface* o para configurar esclavos son almacenados permanentemente. A continuación se describen los cuatro grupos más importantes de listas que son utilizadas en todas las fases del Control de Ejecución.

1. **Imágenes de Datos de Entradas/Salidas en el maestro:** estos datos son leídos o escritos desde o hacia los esclavos.
   - *Imagen de Datos de Entrada (***IDI):** este arreglo contiene la última copia actual de los datos recibidos desde las entradas de todos los esclavos activos. La entrada de datos de esclavos inactivos es llevada a cero (0).
   - *Imagen de Datos de Salida (***ODI):** este arreglo contiene el dato a ser transmitido a los esclavos activos en el siguiente intercambio de datos.
   - *Imagen de Datos de Entradas Analógicas (***AIDI):** este arreglo contiene la última copia actual de los datos recibidos desde las entradas de todos los esclavos activos utilizando transacciones combinadas tipo 1 a 5. La entrada de datos de esclavos inactivos es llevada a $7FFF$.
   - *Imagen de Datos de Salidas Analógicas (***AODI):** este arreglo contiene los datos a ser transmitidos cíclicamente a los esclavos activos con salidas utilizando transacciones combinadas tipo 1 a 5.
2. **Configuración de Datos Imágen en el Maestro.**
   - *Datos Imágen de Configuración (***CDI):** este arreglo contiene la copia actual de la configuración de entradas/salidas y el código de identificación (ID, ID1, ID2) de todos los esclavos, determinado por la lectura de estos datos desde

el esclavo. El dato de configuración de esclavos inactivos es activado a sus valores por defecto (todos los bits en 1).

- *Datos de Configuración Permanente (***PCD):** este arreglo contiene la configuración proyectada de entradas/salidas y los códigos de identificación (ID, ID1, ID2) de todos los esclavos determinado por la configuración local del maestro usando la función *Set_ Permanent_ Configuration* o por la función *Store_ Actual_ Configuration*. Los datos de configuración permanente de los esclavos que no son proyectados serán llevados a sus valores por defecto (todos los bits en 1) y ese dato será almacenado en una memoria no-volátil.

3. **Parámetro Imagen en el Maestro.**

   - *Parámetro Imágen (***PI):** este arreglo contiene la copia actual del parámetro de salida de todos los esclavos activos determinado por las peticiones del último *Write_ Parameter* a los esclavos. El dato *PI* de esclavos inactivos es llevado a valores por defecto (F_{Hex}).
   - *Parámetro Permanente (***PP):** este arreglo contiene los parámetros configurados de todos los esclavos, determinado por la configuración local del dispositivo maestro usando la función *Set_ Permanent_ Parameter*. Estos datos serán almacenados en una memoria no-volátil. Después del encendido del maestro, el *Control de Ejecución* copia todos los valores del parámetro permanente al arreglo de parámetro imágen.

4. **Listas de Esclavos en el Maestro.**

   - *Listas de Esclavos Detectados (***LDS):** en esta lista un bit es colocado para cada esclavo que es detectado por el maestro (a través de la operación inicial o fase de inclusión).
   - *Lista de Esclavos Activos (***LAS):** en esta lista un bit es colocado para cada esclavo que ha sido activo durante la operación de inicio o la fase de inclusión.
   - *Lista de Esclavos proyectados (***LPS):** esta lista contiene los esclavos esperados en el sistema *AS-i*. Esta lista es configurada por la función *Set_ LPS* o por configuración local del dispositivo maestro utilizando la función *Store_ Actaul_ Configuration*. Este dato será almacenado en una memoria no-volátil.
   - *Lista de Fallas Periféricas (***LPF):** esta lista contiene las fallas periféricas de todos los esclavos activos. Este dato será almacenado en memoria volátil.

3.8.2. Máquina de Estados del Maestro

Durante las fases de transmisión, el *Control de Ejecución* envía peticiones al *Control de Transmisión*, las cuales pueden ser realizadas solamente después de la aceptación de la peticiones efectuada previamente. La figura 3.9 muestra el resumen del diagrama de estados del maestro.

Fuente: *Autores del proyecto.*

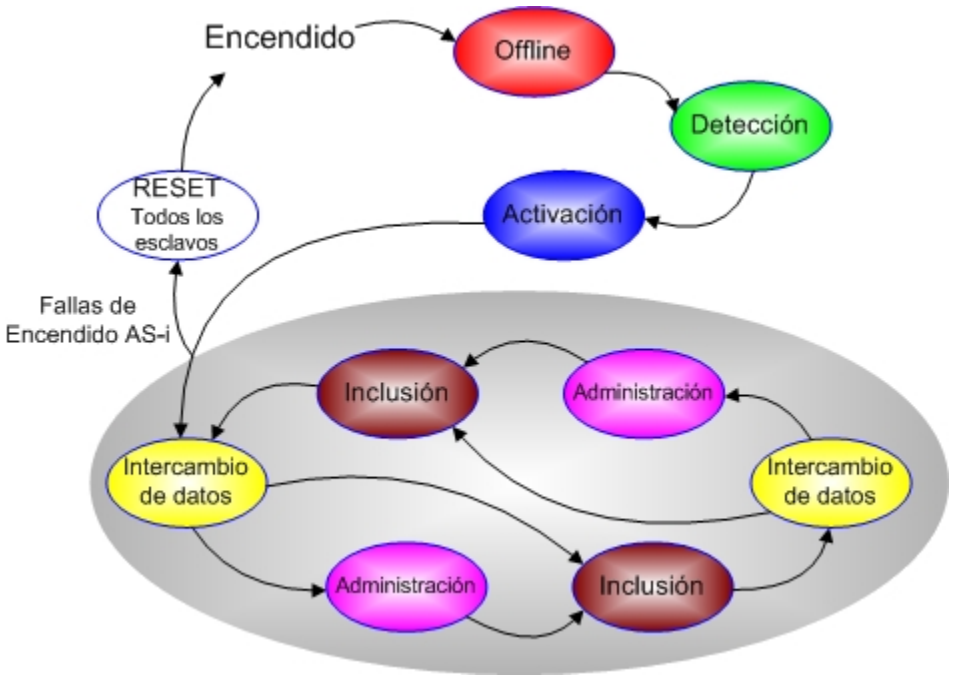


Figura 3.9: **Diagrama de Estados de un Maestro *AS-i*.**

Después que el maestro es accionado, este procesa el inicio de las listas necesarias *(Fase Offline)*. Después de esto, este busca esclavos en el rango de direcciones completa *(Fase de Detección)*. La siguiente fase *(Fase de Activación)* es responsable de la activación de los esclavos conectados lo cuál es desempeñado dependiendo del modo de operación del maestro.

- En caso de Modo Protegido: todos los esclavos detectados y pre-configurados serán activados por el maestro.
- En caso de Modo Configuración: todos los esclavos detectados serán activados por el maestro.

El maestro inicia el intercambio de datos cíclicamente con todos los esclavos activados *(Fase de Intercambio de Datos)*. Al final de cada ciclo este puede enviar un comando

a un esclavo específico *(Fase de Administración)* y actualiza la información de fallas periféricas de los esclavos activos o busca nuevos esclavos enviando un comando a las direcciones de esclavo libres *(Fase de Inclusión)*. En caso de no responder, el maestro inicia el siguiente ciclo para intercambio de datos. Si hay una respuesta de esclavo, el esclavo será activado dependiendo del modo operacional durante el siguiente ciclo *AS-Interface* y será incluido en intercambio de datos normal.

Capítulo 4

DISEÑO DE HARDWARE

En este capítulo se realiza una descripción general del diseño e implementación del *hardware*, incluyendo los criterios de selección de cada uno de los dispositivos electrónicos implementados. Además, se presenta cada una de las etapas que conforman la interfaz física realizada, iniciando con una explicación a nivel de diagrama de bloques de la estructura general abordada para el diseño del maestro y posteriormente se presentará en detalle la implementación de cada una de las etapas, junto con los dispositivos utilizados para su correcto funcionamiento.

El diseño de la interfaz física se realizó teniendo en cuenta la capacidad de transmisión y recepción de información que debe poseer el maestro a través de la red, cumpliendo con los requerimientos mínimos para el establecimiento de una comunicación *AS-i* y de acuerdo a las funciones básicas referenciadas en la norma IEC 62026-2. Estas funciones fueron definidas por etapas, como se describe a continuación:

- **Etapa de control:** basada en el uso de un microcontrolador, es la encargada de realizar el análisis de los datos en los procesos de transmisión y recepción del PC a la Red y viceversa.
- **Etapa de recepción de datos:** encargada de recibir las señales analógicas provenientes de la línea *AS-i* y convertirlas en pulsos para su posterior procesamiento.
- **Etapa de transmisión de datos:** encargada de transformar la señal de tensión proveniente de la etapa de control en señales de corriente[1], para su posterior introducción en la línea *AS-i*, y de esta forma poder enviar las peticiones realizadas por el maestro a los esclavos de la red.

[1]La transmisión de datos debe ser realizada a través de una señal de corriente de acuerdo a lo establecido por el protocolo *AS-i*.

- **Etapa de alimentación:** capaz de extraer la potencia de la línea *AS-i* para entregarla a las etapas mencionadas anteriormente y así asegurar el correcto funcionamiento del sistema.

La figura 4.1 muestra el diagrama de bloques general del maestro *AS-i*, con las etapas descritas anteriormente, donde se aprecia la conexión del bus al maestro y la manera como es enlazado internamente hacia los bloques de transmisión, recepción y alimentación.

Fuente: *Autores del proyecto.*

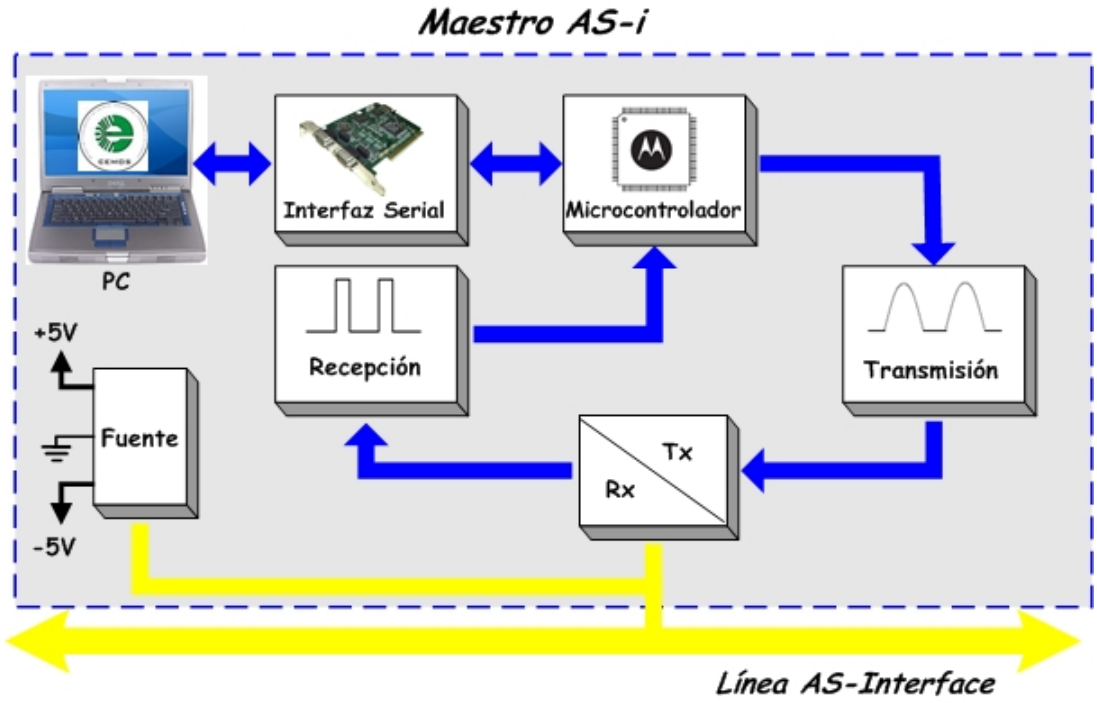


Figura 4.1: **Diagrama de bloques del Maestro *AS-i*.**

Una vez que las peticiones del maestro son procesadas en el PC con el software de desarrollo *LabView*, los datos son llevados por medio del puerto serie a la etapa de control la cuál se encarga de generar la transmisión de la información, por medio de señales de voltaje con la forma de onda de corriente especificada en la norma IEC 62026-2 a la etapa de transmisión para luego inyectar la corriente a través de la línea *AS-i*. Posteriormente, al recibir una respuesta desde el esclavo, el bloque de recepción será habilitado para llevar la información proveniente de la red a la etapa de control y desde esta etapa serán llevados los datos al PC para su procesamiento y visualización en *LabView*.

Además, se cuenta con una etapa de alimentación, encargada de suministrar los voltajes de alimentación a las demás etapas internas que componen al maestro. Allí las componentes de la señal proveniente de la línea de transmisión *AS-i* que contengan información serán eliminadas a través de un filtro pasa bajos, obteniendo así una tensión DC, que posteriormente será adecuada de acuerdo a los requerimientos de los circuitos internos que utilizaremos.

A continuación se dará una explicación detallada del diseño de cada bloque, a excepción de la etapa correspondiente a la interfaz serial la cuál será tratada en el capítulo 5. Además se efectuará una breve descripción de los dispositivos que fueron utilizados para su implementación.

4.1. Etapa de Control

Esta etapa permite realizar el análisis de los datos provenientes tanto del PC, como de la red, llevando un orden específico en las transmisiones y recepciones, tal y como se explica la sección 5.1, la cual trata detalladamente la programación del microcontrolador.

Para efectuar el análisis y la comunicacion de datos entre la red y el PC, se cuenta con 3 dispositivos fundamentales, los cuales son un microcontrolador Motorola MC9S12E128 [9], una tarjeta de comunicaciones PCI[2] a Serial *Quatech DSC-100*, y un conversor *TTL*[3] a *RS-232* de *Texas Instruments* SN75C3221 [10].

Para poder realizar la comunicación serial entre el PC y el microcontrolador, se buscó un dispositivo que garantizara por lo menos 2 veces la velocidad de transmisión del protocolo *AS-i*, es decir, $166,67$ *Kbps*. En un comienzo se empleó una tarjeta de comunicación USB-Serial, referenciada en la subsección 5.2.2, la cual al momento de realizar las pruebas garantizaba una velocidad superior a 1Gbaud, con la desventaja de que introducía retardos en la transmisión, debido a que manejaba buffers de mínimo 128 bytes, generando un tiempo de espera determinado para la captura total del buffer. Esta alternativa no fue tomada, ya que las características de transmisión *AS-i*, exigen una petición de maestro con una respuesta casi instantánea del esclavo y no da lugar al manejo de buffers de información.

La solución encontrada fue la adquisición de una tarjeta PCi-Serial, que cumple

[2] *Peripheral Component Interconnect.*
[3] *Transistor-Transistor Logic.*

con los requerimientos de la comunicación, cuyo funcionamiento se encuentra descrito también en la subsección 5.2.2.

La recepción y la transmisión de datos entre el microcontrolador y el PC se realizan mediante la tarjeta *QUATECH DSC-100*, la cual en sus puertos de entrada y salida maneja niveles *TTL*, diferentes a los *RS-232* manejados por la SCI[4] del microcontrolador. Mediante el uso del circuito integrado SN75C3221, se logró efectuar la conversión de niveles *TTL* a *RS-232* y viceversa, garantizando la integridad de las señales a una velocidad de transmisión de $460,8$ *Kbps*.

El microntrolador utilizado para el desarrollo de la aplicación, es el MC68HC9S12E128 de 16 *bits*, de la familia HCS12 de *Motorola*. Este cuenta con 128 *Kbytes* de memoria EEPROM[5], 8 *Kbytes* de memoria RAM[6], y herramientas de gran importancia en la implementación del maestro, como 3 módulos SCI y 2 módulos DAC. La MCU[7] funciona a una velocidad de 22,1184 MHz, con ayuda del PLL[8] interno y de un oscilador externo de 4,9152 MHz. La programación se realiza por medio de una interfaz USB, a través del sofware Codewarrior [11].

Cabe destacar que se realizó el diseño del sistema de desarrollo del microcontrolador, una vez adquirido el core MC68HC9S12E128. La tarjeta implementada es mostrada en la figura 4.2, y está diseñada solo con los periféricos necesarios para el funcionamiento de la aplicación.

4.2. Etapa de Recepción de Datos

Esta etapa se encarga de recibir los patrones analógicos de voltaje provenientes de la línea *AS-i*, los cuales fueron mencionados previamente en la sección 3.3 como aquellos niveles de tensión que pueden ser representados como una función seno al cuadrado para cada pulso y cuya aparición es el resultado de la corriente que circula por la red y la impedancia característica de la misma. Posteriormente, dichos pulsos son convertidos a niveles lógicos para ser llevados a los puertos del microcontrolador en la etapa de control y así poder realizar el procesamiento adecuado de los datos que van a ser enviados al PC.

[4] *Serial Communications Interface.*
[5] *Electrically Erasable Programmable Read Only Memory.*
[6] *Random Access Memory.*
[7] *Microcontroller Unit.*
[8] *Phase Looked Loop.*

Fuente: *Autores del proyecto.*

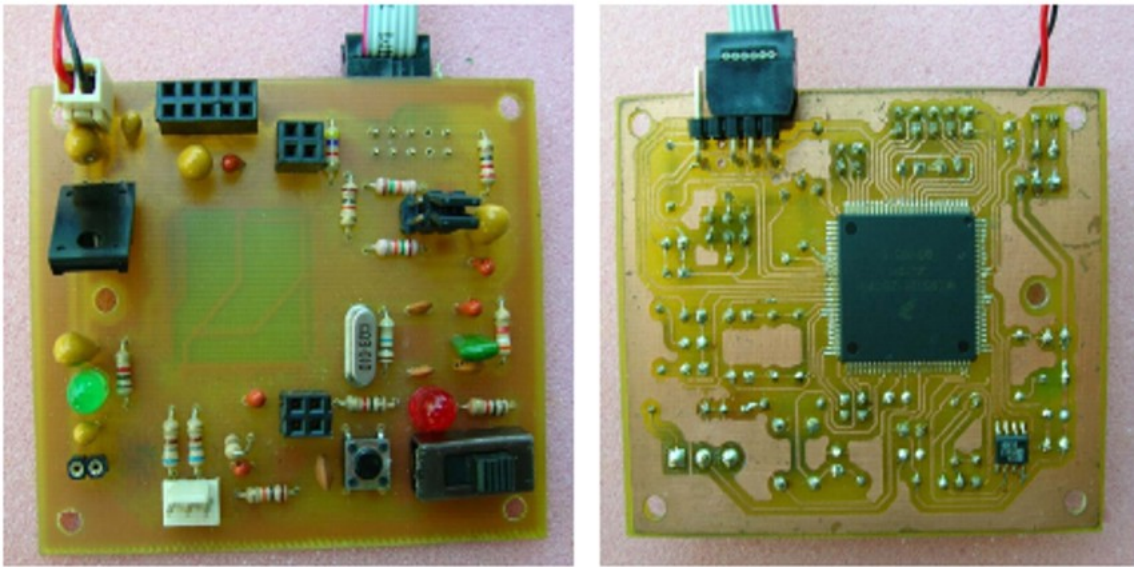


Figura 4.2: **Sistema de desarrollo. (a) Cara superior. (b) Cara posterior.**

Para el buen desarrollo del proceso de recepción se diseñó el circuito de detección de pulsos mostrado en la figura 4.3, el cuál está dividido en cuatro sub-etapas las cuáles serán explicadas brevemente a continuación.

Fuente: *Autores del proyecto.*

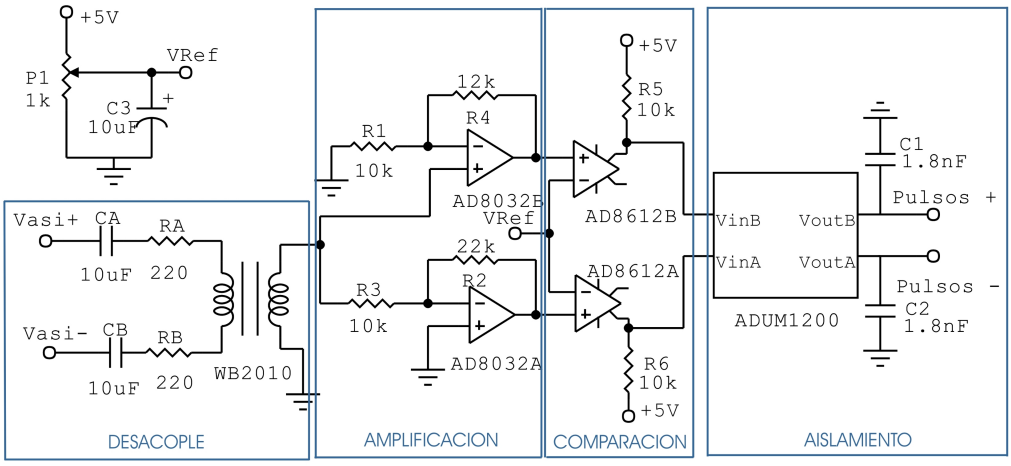


Figura 4.3: **Etapas del circuito receptor.**

En la primera sub-etapa se implementó un circuito de desacople en corriente continua ($R_A C_A$ $R_B C_B$) y una etapa de aislamiento con el transformador WB2010 [12]

con una relación de 1:1 y una corriente de saturación del núcleo de 250 mA, cuyo fin es extraer la señal mensaje del bus.

El patrón de voltaje obtenido anteriormente es llevado a la segunda sub-etapa que es la encargada de la amplificación de éstas señales, con el objeto de mejorar la resolución y así facilitar el proceso de comparación que se observará en la siguiente sub-etapa. La señal es amplificada a través de un circuito que utiliza una configuración no inversora, encargada de la señal que será utilizada para obtener los pulsos positivos, y otro circuito cuya configuración inversora permitirá la obtención de los pulsos negativos posteriormente. Para tal fin se utilizó el circuito integrado $AD8032$ [13], que se caracteriza por tener dos amplificadores operacionales de propósito general y una respuesta en frecuencia acorde con las características de la señal, mostradas en el capitulo 2. Los valores de resistencia fueron escogidos para llegar a una ganancia de 2.2 V/V en las dos etapas de amplificación, con el fin de obtener señales con un rango máximo menor a 10 V_{p-p}, que sería el valor en el cual se saturarían los amplificadores operacionales. Cabe destacar que entre mayor sea la amplitud de la señal entregada por cada amplificador, el proceso de comparación se facilitará considerablemente.

La tercera sub-etapa que es la de comparación fué desarrollada utilizando el circuito integrado $AD8612$ [14], que consta de dos comparadores de alta velocidad, con un retardo de 4 ns a ± 5 V, y con un ancho de banda de 100 MHz. Aquí se hace la discriminación de los pulsos analógicos positivos y negativos para convertirlos en niveles lógicos (pulsos digitales) que serán enviados al microcontrolador. Las señales tomadas de la etapa de amplificación son comparadas con un nivel de referencia que en nuestro caso es la menor amplitud pico que deben tener los pulsos analógicos para ser válidos y cuyo valor es aproximadamente 1 V_p de acuerdo a lo recomendado en [7]. Su implementación fue realizada con un divisor resistivo, utilizando un potenciómetro (trimmer), para la obtención del nivel de voltaje requerido.

Para finalizar, la última sub-etapa efectúa la adecuación de los pulsos por medio del aislador digital ADUM1200 [15], que al ser saturado mejora la forma de los pulsos obtenidos en la sub-etápa anterior y aisla en señal los puertos de entrada del microcontrolador de las salidas del comparador.

4.3. Etapa de Transmisión de Datos

En esta sección se presenta la implementación de la transmisión de datos desde el maestro hacia la red *AS-i* comercial, la cual está ubicada en el laboratorio de mecatrónica de la escuela de ingeniería mecánica. Luego de varios estudios al protocolo y con la colaboración del grupo de investigación *CEMOS*, se logró el diseño de una fuente de corriente con el fin de generar los pulsos analógicos a través de la línea de transmisión como se observó en el capítulo 3, especialmente en la figura 3.2. Como se ha podido evidenciar, estos niveles de tensión se logran inyectando una señal de corriente que junto con la impedancia de la red forman los pulsos analógicos que harán posible la comunicación con los diferentes esclavos.

El proceso de generación de los pulsos analógicos inicia con la inyección de una señal emitida por el microcontrolador, especialmente por la salida del módulo conversor digital-analógico (DAC) del mismo, la cual fué creada según las normas especificadas para las tensiones y los tiempos que debe manejar esta señal referenciados en [2] [7]. La programación y otros detalles son mencionados en el capítulo 5 de forma completa.

Luego de efectuada la programación, se obtuvieron valores en los niveles de tensión a la salida del DAC de 0 V a 2 V y tiempos de bit de aproximadamente 6μs como lo especifican las normas que rigen el protocolo *AS-i*. En vista de que la máxima corriente que puede circular a través de la salida del DAC del microcontrolador, es de 40 mA, según su hoja de datos, fué vital la incorporación de un circuito de acople (LMV821) [16] entre el microcontrolador y la fuente de corriente, el cual fué configurado en la tarjeta de programación del microcontrolador como un circuito seguidor de corriente, y cuyo fin es el de ofrecer una protección contra sobre-corrientes al puerto de salida del DAC para evitar daños en el microcontrolador.

Una vez logrado el acople de la señal proveniente del microcontrolador, se continuó con la implementación de una fuente de corriente controlada por tensión, capaz de alimentar una carga diferencial, la cual representa la impedancia vista por el maestro a través de la línea *AS-i*. Además, ésta fuente de corriente posee la característica especial de presentar pocas variaciones en la salida a pesar de los posibles cambios en su carga. Teniendo en cuenta las anteriores consideraciones se decidió escoger una topología de fuente de corriente *Howland* diferencial o complementaria, la cual es mostrada en figura 4.4.

Fuente: *Autores del proyecto.*

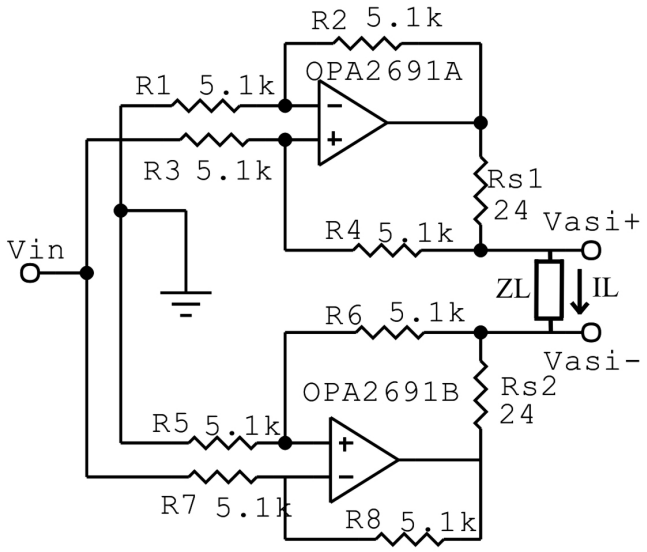


Figura 4.4: **Etapas del circuito transmisor.**

La fuente *Howland* diseñada es balanceada y se construye a partir de una fuente diferencial que cumple con las condiciones mostradas en la ecuación [4.1] y cuya primordial característica es la independencia de la corriente de salida respecto a la carga lo que hace que se mantenga el flujo de corriente así la impedancia de salida sufra variaciones

$$\frac{R_2}{R_1} = \frac{R_4}{R_3} = \frac{R_6}{R_5} = \frac{R_8}{R_7} \tag{4.1}$$

La función de transferencia de la configuración diferencial depende del cumplimiento simultaneo de las ecuaciones [4.2] y [4.3], las cuales hacen referencia la configuración no inversora y a la configuración inversora respectivamente.

$$I_L = \frac{V_{in}}{R_{s1}} \tag{4.2}$$

$$I_L = \frac{-V_{in}}{R_{s2}} \tag{4.3}$$

Las condiciones expresadas anteriormente implican que R_{s1} debe ser igual a R_{s2} para asegurar que la corriente que entra a la carga sea la misma corriente que sale de ella, garantizando así el carácter complementario de la fuente implementada.

Cuando se mantiene un nivel de voltaje fijo a la entrada de la fuente *Howland*, la corriente de salida dependerá únicamente de R_{s1} y R_{s2}. En la implementación del maestro *AS-i*, se manejó un nivel de tensión a la salida del DAC del microcontrolador con un valor de 2 V_{p-p} y se ajustaron las resistencias R_{s1} y R_{s2} a 24 Ω con el fin de generar una corriente de 55 mA, siendo un valor aceptable entre los niveles establecidos por el protocolo referenciado en [2] [7]. Además, se utilizaron resistencias de precisión R_n de una tolerancia del 1 % y un valor de 5,1 $k\Omega$, con el fin de hacer posible el balance adecuado del circuito. Este valor fue escogido en base a un análisis de resultados experimentales que se obtuvieron al manejar diversas cargas a la salida de la fuente de corriente. Con valores más elevados disminuye la resistencia de salida al aumentar el margen de precisión de las resistencias y con resistencias de menor valor disminuye la resistencia de entrada del circuito.

El circuito integrado escogido para esta implementación debía tener un par de amplificadores operacionales que contaran con características fundamentales tales como la capacidad de corriente de salida, las cuales en nuestro caso deben ser superiores a 70 mA; un *slew rate* del amplificador que debe ser superior a 1,05 $V/\mu s$, que es el valor obtenido al ser calculado con una señal en la salida con las características descritas en el capítulo 3. Con un *slew rate* superior a este valor se garantiza un funcionamiento adecuado del circuito. Además, los amplificadores deben poseer un ancho de banda superior a 167 kHz cuando este se encuentre realimentado y una alimentación dual de ± 5 V.

Una vez estudiadas las condiciones mencionadas anteriormente, y luego de verificar los amplificadores disponibles en el mercado, se escogió el $OPA2691$ [17], debido a su buen desempeño y al cumplimiento que éste hace de los requerimientos necesarios para la implementación de la fuente de corriente.

4.4. Etapa de Alimentación

Esta etapa fué desarrollada con el fin de realizar un aislamiento de los niveles de señal de información de los circuitos de regulación de tensión; para ello se contó con dos inductancias de 1,8 mH, dispuestas para que actúen como un camino de alta impedancia para la señal proveniente de la red y como un corto circuito para los niveles de corriente contínua. Además, por medio del circuito mostrado en la figura 4.5 se logró una reducción en el nivel diferencial de tensión proveniente de la red cuyo

valor aproximado es de 30 V, estableciendo a su vez una referencia a tierra para todos los dispositivos utilizados en el *hardware* del Maestro. La disminución a los niveles de voltaje utilizados por los diversos dispositivos usados durante el diseño del *hardware* (± 5 V) se desarrolló en primera instancia por medio del regulador $LM317$ con el cual se logró disminuir la tensión a 10 V, con el cual se alimenta el circuito inversor de tensión que se presenta más adelante. Para lograr el nivel de referencia positiva de 5 V se empleó el regulador $LM7805$.

Fuente: *Autores del proyecto.*

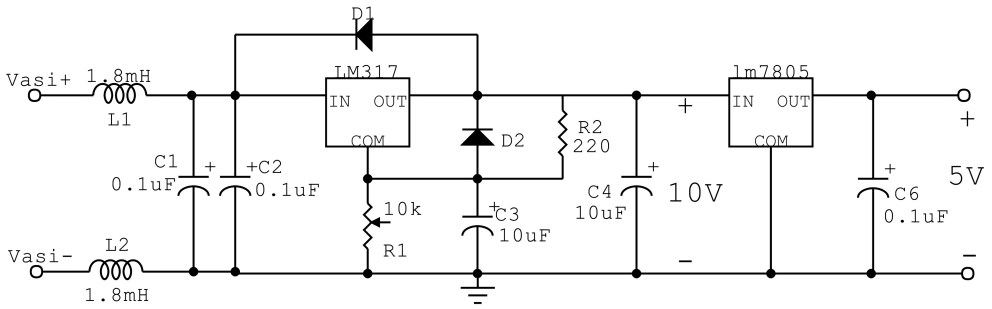


Figura 4.5: **Etapa de extracción de potencia de la red *AS-i*.**

Para una correcta polarización de los circuitos integrados que se utilizaron en nuestra aplicación, se necesitaba una referencia negativa de voltaje (-5 V), que hasta el momento no existía. Para ello fue necesario recurrir a un inversor de tensión cuyo fin es el de lograr este nivel negativo, a partir de un valor de tensión positiva. Con este fin se utilizó el $LT1054$ [18], que es un conversor de voltaje a partir de la conmutación de condensadores, configurado como se muestra en la figura 4.6. La tensión obtenida fue una referencia de -5 V, utilizando una tensión de entrada de 10 V. Cabe anotar que la tensión obtenida a plena carga de los requerimientos del maestro es $-4{,}9$ V, sin que esto afecte el funcionamiento adecuado de los circuitos que la utilizan.

Como resultado del diseño de la interfaz de comunicación del Maestro, se puede observar un circuito capaz de enviar y recibir datos desde y hacia la red AS-i comercial utilizada como el mostrado en la figura 4.7, el cual consta de una entrada para la señal proveniente del bus y cuatro puertos para comunicación externa con dispositivos de entrada-salida, para efectuar el intercambio de información digital. El funcionamiento lógico del Maestro se muestra de forma más detallada en el capítulo 5 donde claramente se puede apreciar la programación tanto en el microcontrolador como en el programa

Fuente: *Autores del proyecto.*

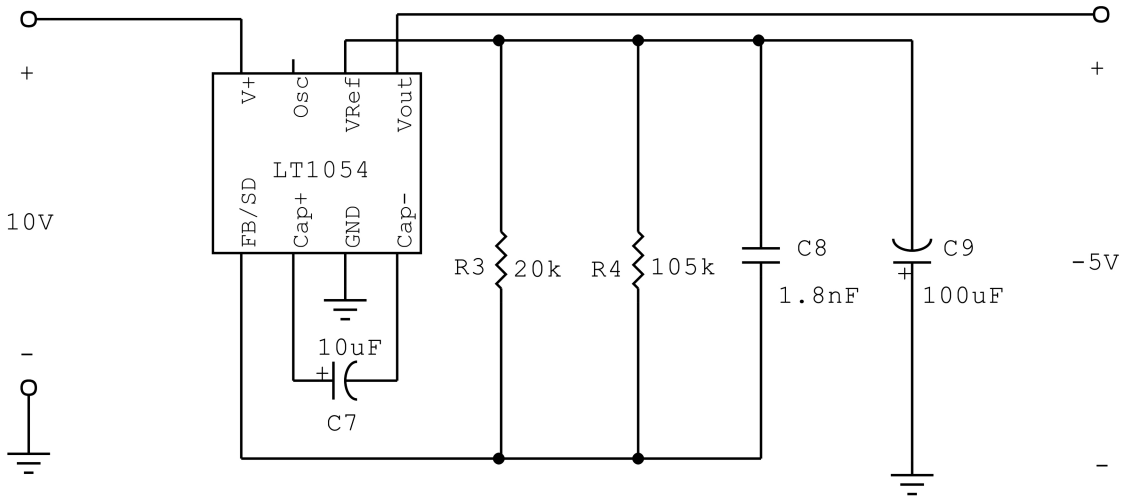


Figura 4.6: **Etapa de inversión de voltaje.**

LabVIEW® que hacen posible la transmisión y recepción de datos con la red.

Fuente: *Autores del proyecto.*

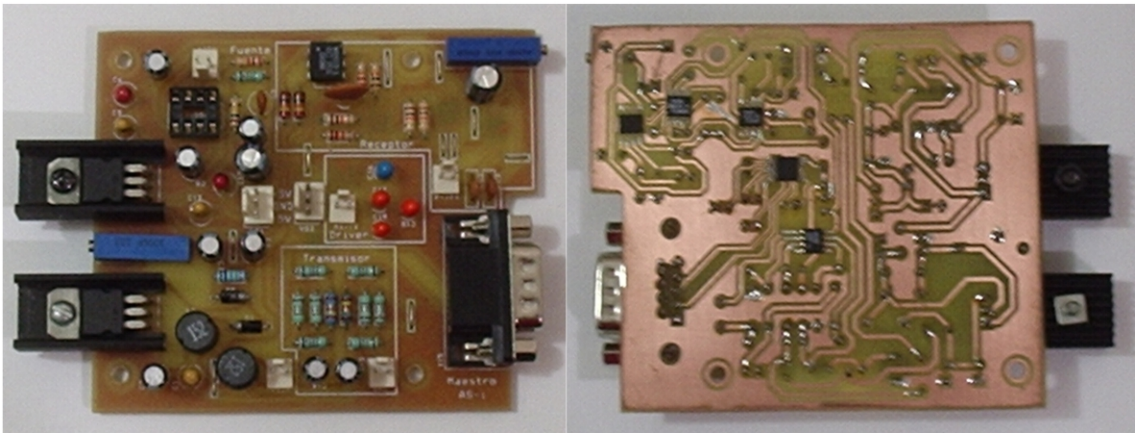


Figura 4.7: **Interfaz de comunicación del maestro.**

Capítulo 5

DISEÑO DE SOFTWARE

En este capitulo se describe de manera general la programación del software del sistema, tanto en el microcontrolador, mediante los lenguajes C y *assembler*, como en el PC por medio del lenguaje de programación G[1] de LabVIEW®[2].

5.1. Software del microcontrolador

En esta sección se describe la programación de la máquina de estados implementada en el microcontrolador, que permite el intercambio de datos entre la red *AS-i* y el computador. Esta programación cuenta con cinco etapas principales mostradas en la figura 5.1, elaboradas en el software *Codewarrior*, el cual realiza la compilación mediante lenguaje C, C++ y *Assembler*, siendo el lenguaje C y *Assembler* las alternativas utilizadas [19] [11].

Cada una de las fases es creada a manera de función, es decir, se crea un subprograma digitado al final del programa principal, el cual puede ser llamado en cualquier momento.

A continuación se hace una breve descripción de cada una de las etapas de la máquina de estados, con el fin de facilitar la comprensión de toda la programación implementada en el microcontrolador.

5.1.1. Inicialización de Puertos

En esta sección se realiza la configuración de los periféricos pertenecientes al microcontrolador que se utilizan en la comunicación entre la red y el PC. Estos dispositivos son el DAC *(Digital to Analogue Converter)*, los *timers*, la SCI *(serial communica-*

[1] Lenguaje de programación usado para crear programas basados en diagramas de bloques

[2] *Laboratory Virtual Instrument Engineering Workbench*

Fuente: *Autores del proyecto.*

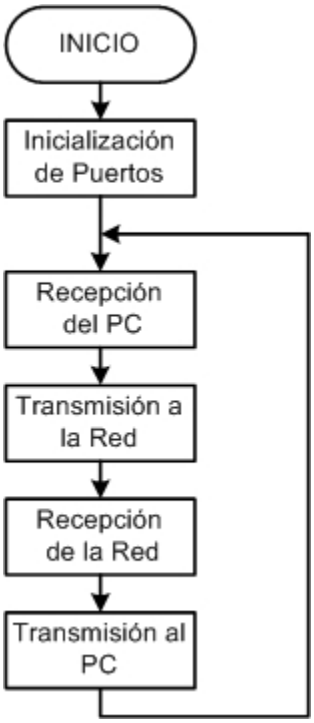


Figura 5.1: **Diagrama de flujo de la programación en el microcontrolador.**

tions interface) y los GPIO *(General Purpose Input Output)* que serán descritos en las siguientes etapas.

5.1.2. Recepción del PC

En esta etapa entra a funcionar la SCI o interfaz serial de comunicación, la cual permite el envió y recepción de datos digitales con otros dispositivos de una forma serial, bidireccional y asíncrona. Este periférico funciona enviando tramas de 10 *bits*, los cuales hacen referencia a un *bit* de inicio de valor cero, un *bit* final de valor uno y 8 *bits* de datos. De este modo se reciben dos *bytes* enviados desde el PC que contienen cada uno 7 *bits* de información relacionados a los 14 *bits* totales que componen la trama de maestro. Los dos *bytes* son manipulados y organizados dentro de una registro de 16 *bits* llamado datoRedTx, que permite su almacenamiento para su posterior utilización en la siguiente etapa.

5.1.3. Transmisión a la Red

Como ya se ha visto en la parte correspondiente al hardware de la sección 4.3, para realizar el envío de datos a la red, se inyecta una señal de corriente a la etapa

de transmisión, esta señal es característica del protocolo *AS-i* y es creada por medio del conversor digital-analógico del que dispone el microcontrolador. Debido a que el circuito seguidor interno del DAC tiene un *slew rate* de $2V/\mu s$ y para maximizar la rata de envió de datos, la amplitud de la señal de salida se escogió a una amplitud de $2V$. Los datos son enviados con una frecuencia aproximada de 4,33 MSPS[3], con un total de 26 muestras por tiempo de *bit*.

Los bits almacenados durante la etapa anterior en el registro datoRedTx y que corresponden a la trama del maestro, son convertidos uno a uno a señales de corriente según su valor digital y su posición dentro del arreglo. Se analiza el dato presente y el dato futuro para saber cual debe ser el arreglo característico de niveles a enviar al DAC. La tabla 5.1 muestra los valores presente y futuro con su respectivo tipo de señal y la figura 5.2 los 4 tipos de señales de corriente según los niveles digitales presente y futuro.

Tabla 5.1: **Relación de bits con el tipo de señal de corriente.**

Presente	Futuro	Tipo de Señal
1	1	PNEG
1	0	DPOS
0	1	DNEG
0	0	NPOS

Fuente: *Autores del proyecto.*

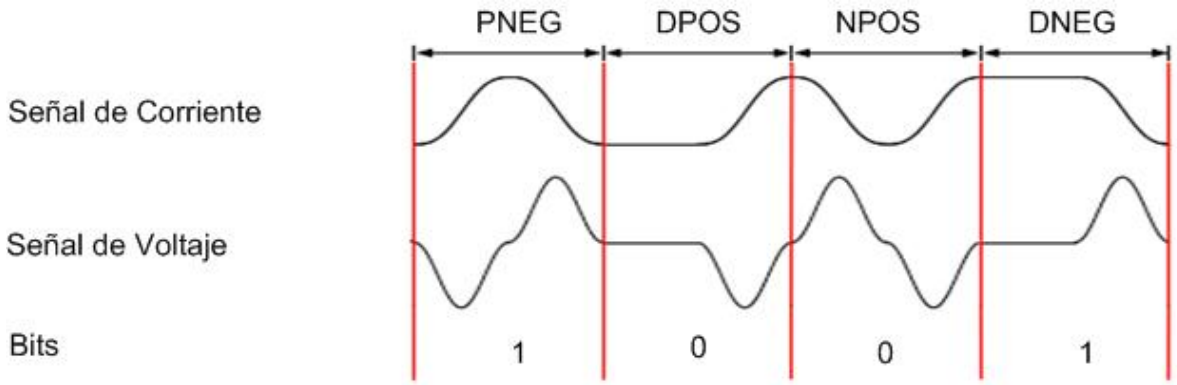


Figura 5.2: **Tipos de Señales de Corriente.**

El diagrama de flujo del programa de transmisión es mostrado en la figura 5.3, en la cual se puede observar que se comienza enviando el bit SB *(Start bit)*, por defecto

[3] *Mega Samples Per Second*

un 0 digital, y no necesita análisis previo para su envío, ya que este siempre va a tener la misma forma de onda sea cual fuere la transacción. Después de esto el programa está definido para enviar los 13 *bits* restantes pertenecientes a la trama de maestro, realizando el análisis ya mencionado anteriormente en el que se tiene en cuenta, tanto el *bit* presente como el futuro. El registro de almacenamiento se va desplazando hacia la izquierda para ir tomando el *bit* más significativo a ser enviado. El programa entra en una iteración de análisis y envío hasta el momento en que un contador determina que efectivamente se han enviado los 14 *bits*.

Fuente: *Autores del proyecto.*

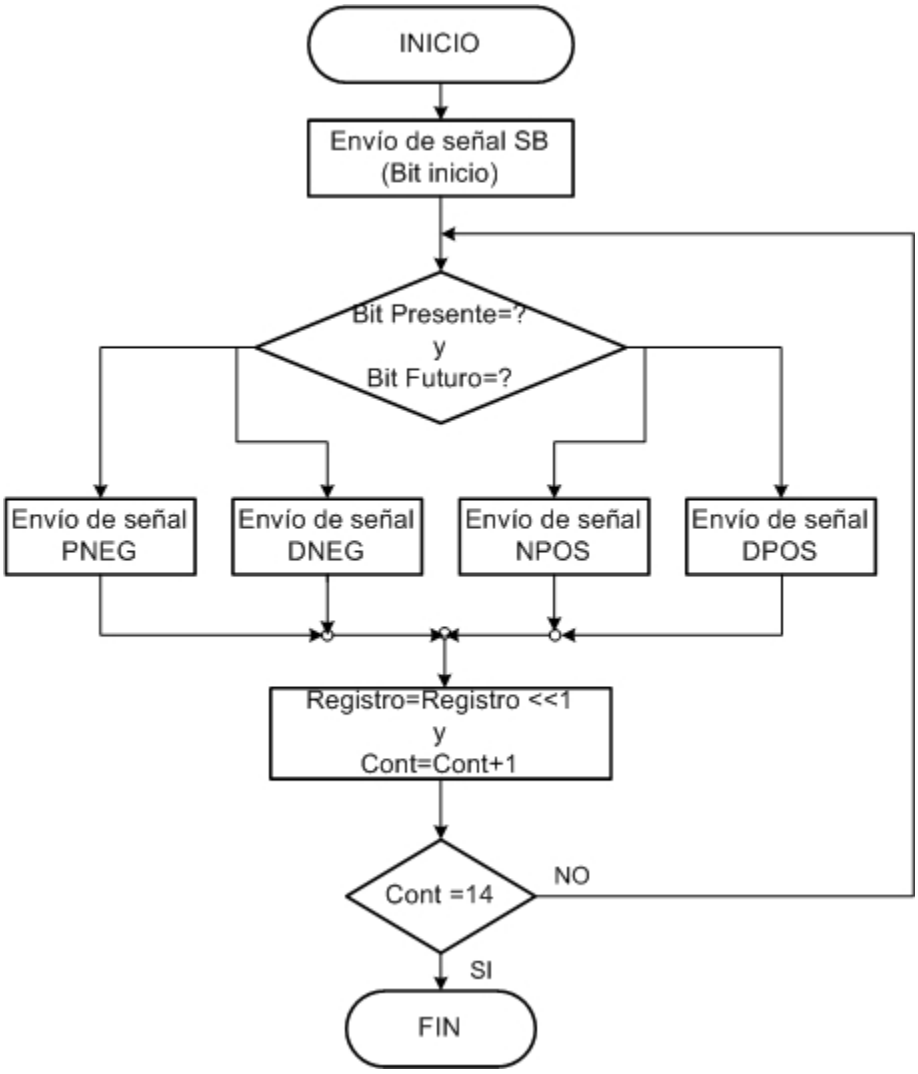


Figura 5.3: **Diagrama de flujo de la transmisión a la red.**

5.1.4. Recepción de la Red

En la programación de la recepción de la red, tiene un papel muy importante el uso de los *timers*, los cuales se pueden configurar de dos maneras, como *input capture*, para capturar los pulsos provenientes del *hardware* de recepción, y como *output compare*, para comparar el registro de tiempo interno del microncontrolador con tiempos de espera necesarios en el proceso de captura. Cabe resaltar que una vez puesto en marcha un temporizador como comparador de tiempo, este se reinicia cada vez que se cumple su tiempo de configuración.

En la figura 5.4, se muestra el diagrama de flujo referente a esta etapa, que comienza con un tiempo de espera de aproximadamente $50\mu s$, en donde si se ha cumplido este tiempo y no se ha presentado la captura de algún pulso negativo, se procede a escribir en el registro de envío al computador el numero 1. Este número es interpretado por el programa en LabVIEW®, como una ausencia o error en la contestación por parte del esclavo encuestado.

Si por el contrario durante el transcurso del conteo de los $50\mu s$ se hace la captura de algún pulso negativo, inmediatamente el programa entra a procesar y capturar los restantes pulsos que forman una respuesta de esclavo.

El primer pulso negativo capturado no es tenido en cuenta para el resto del proceso, ya que corresponde a un 0 digital, es decir, es el SB (bit de comienzo) de la trama de esclavo. El programa a continuación habilita un nuevo temporizador, que sirve para realizar la captura de pulsos cada $3\mu s$, en donde para este primer pulso simplemente se espera que termine dicho tiempo, para luego entrar a capturar uno nuevo. Si el nuevo pulso recibido es positivo, se desplaza el registro RxRed una posición hacia la izquierda, para introducir un nuevo *bit* con valor 1. Si por el contrario el pulso recibido es negativo, simplemente se desplaza el registro una posición hacia la izquierda tal y como si simplemente se estuviera introduciendo un 0.

Este proceso se repite hasta el momento en que se hayan capturado y almacenado en el registro RxRed los 7 *bits* que conforman la trama completa de esclavo.

Fuente: *Autores del proyecto.*

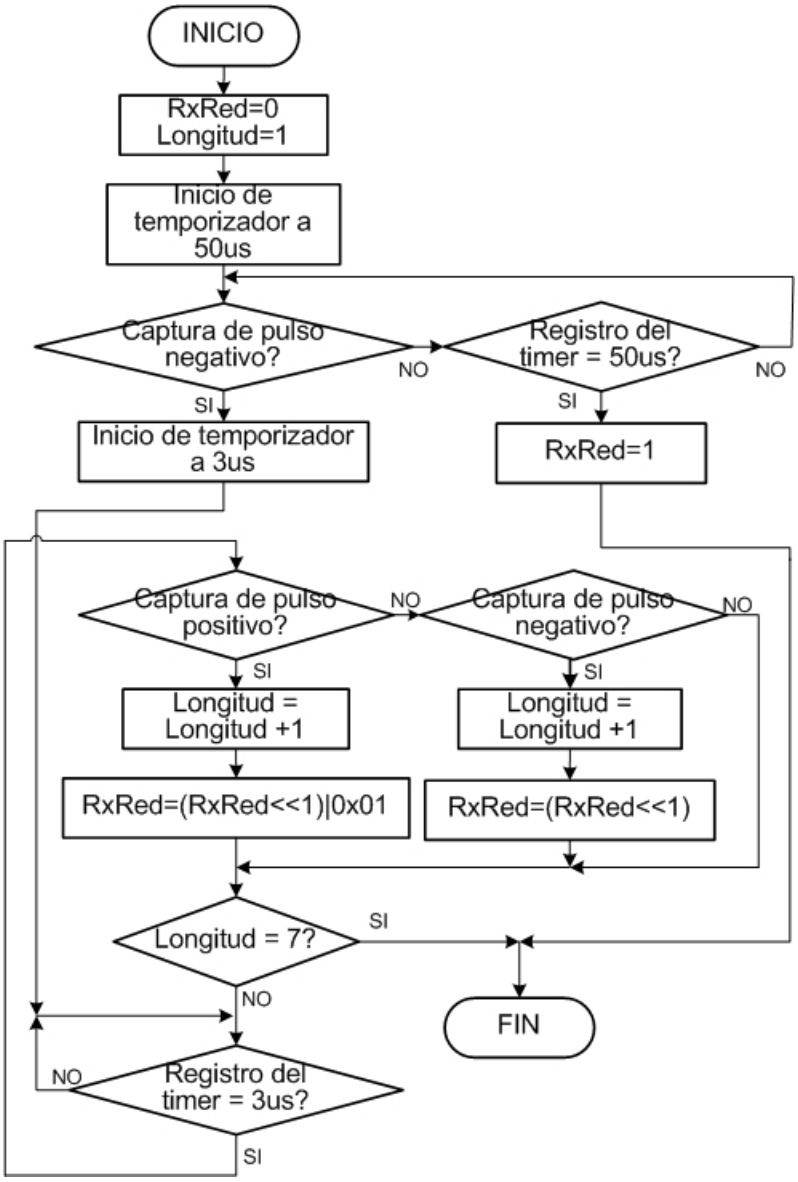


Figura 5.4: **Diagrama de flujo de la recepción de la red.**

5.1.5. Transmisión al PC

Para la transmisión al computador se vuelve a emplear la sci, esta vez transmitiendo un *byte*, el cual contiene el registro RxRed con la información de la respuesta de esclavo recibida en la etapa anterior. Finalmente se forma un ciclo repetitivo en el que el microcontrolador sirve de intermediario en la comunicación entre el computador y la red.

5.2. Implementación del Maestro *AS-Interface* en LabVIEW®

En esta sección se hace una explicación general de la manera como fueron construidas las tramas de datos características del protocolo *AS-Interface*, así como el diseño de las principales fases de control de ejecución, incluyendo la captura de las tramas respuesta (que son entregadas por el esclavo consultado) empleando una interfaz desarrollada en LabVIEW® [20].

5.2.1. Módulo de Visualización del Maestro (Panel Frontal)

En el panel frontal mostrado en la figura 5.5 se enumeran algunos de los controles e indicadores principales, que se utilizan para efectuar la comunicación con dispositivos de la red *AS-Interface* (esclavos). A continuación se dará una breve explicación de cada uno de ellos:

Fuente: *Autores del proyecto.*

Figura 5.5: **Panel Frontal del Maestro *AS-i*.**

1. **Arranque del Maestro *AS-Interface*.** Este control sirve para iniciar el envío de datos desde el maestro luego que LabVIEW® es iniciado en modo *RUN*.

2. **Configuración de Puerto Serial.** En esta parte del Panel Frontal se puede elegir el puerto serie del PC (COM1, COM2,..., COM8, COM9, etc.) que se va a utilizar para el envío y recepción de datos desde y hacia la red. Además, en el diagrama de bloques mostrado en la figura 5.6 se permite configurar la velocidad de transmisión del puerto, que en nuestro caso está fijada en un máximo de 460,80 *kbps* gracias a que éste es la máximo *baud rate* permitido por la tarjeta PCI-Serial que se adquirió y cuya descripción se hará más adelante en esta sección. Las demás opciones de configuración como la paridad (ninguna), el numero de bits de datos (8 por defecto), el bit de parada (1) y el *time-out* [4], son constantes.

3. **Visualización de las Fases del Maestro.** En esta etapa se puede observar cómo se ejecutan cada una de las fases del maestro utilizando *leds* (indicadores) que sirven para hacer un monitoreo del comportamiento de la red. Más adelante se hará una breve explicación de cómo se ha implementado cada una de las fases y las tramas del protocolo de comunicación correspondientes a cada una de ellas.

4. **Banderas.** Las banderas son utilizadas para señalar el estado del maestro y sirven para verificar algunas funciones específicas del maestro, las cuáles serán explicadas posteriormente.

5. **Listas y Datos manejados por el Maestro.** En esta parte de la interfaz con el usuario, se pueden observar algunas listas utilizadas por el maestro para llevar un control de aquellos esclavos que están activos. Además, se puede manipular algunos intercambios de datos con los esclavos de la red *AS-interface*. A medida que se vaya explicando cada una de las fases en la siguiente sección, se pueden observar que listas y tramas son activadas dependiendo de la fase en la que se encuentre el maestro.

6. **Créditos.** Aquí se hace el reconocimiento de aquellas personas que han hecho posible el desarrollo de este proyecto de grado.

5.2.2. Tarjeta PCI-Serial

La comunicación serial entre el PC y la interfaz SCI del microcontrolador, se hizo a través de la tarjeta PCI-Serial (Quatech DSC-100) mostrada en la figura 5.7, la cuál

[4] Tiempo de llegada de bytes al puerto serie

Fuente: *Autores del proyecto.*

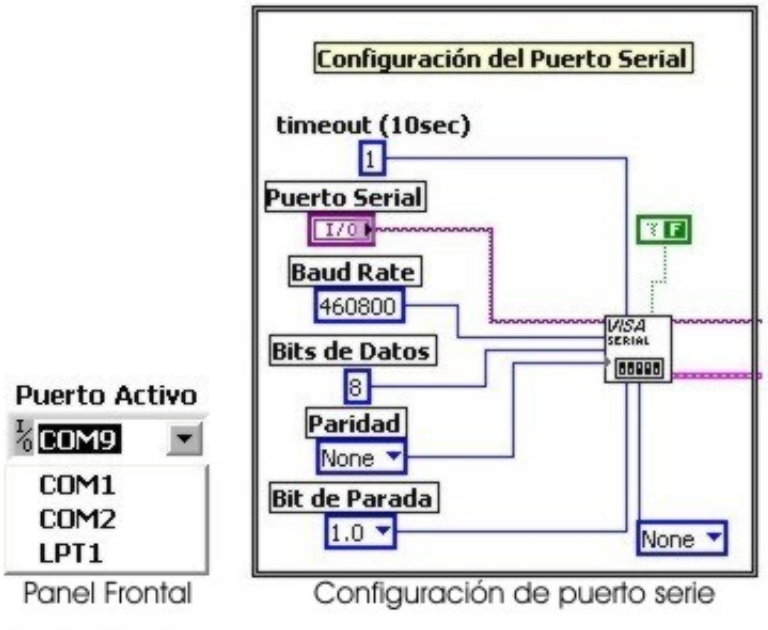


Figura 5.6: **Configuración de Puerto Serial.**

permitió hacer el envío y recepción de datos a una velocidad mucho mayor que la obtenida por defecto de un puerto serial común del PC (125 *Kbps*). Esta tarjeta nos permitió contar con las siguientes ventajas:

- Dos puertos independientes RS-232 con conectores DB-9.
- Velocidad de transmisión de hasta 460,80 *Kbps*, suficiente para hacer un buen procesamiento de datos en el microcontrolador sin que se tuvieran problemas de tiempos.
- Conexión *plug-and-play*, es decir, el sistema la reconoce sin necesidad de instalar drivers.
- Es soportada bajo los sistemas operativos Windows 95/98/Me/NT/2000/XP y Linux.

Cabe destacar que se hicieron pruebas anteriores utilizando el puerto USB del PC con la tarjeta mostrada en la figura 5.8, pero surgieron problemas en los tiempos entre peticiones al ser del orden de milisegundos, lo cuál no era conveniente teniendo en cuenta que se quería que éstos fueran del orden de los microsegundos debido a las exigencias de la norma *AS-i*. Por eso se decidió buscar en el mercado una alternativa económica

Fuente: *Autores del proyecto.*

Figura 5.7: **Tarjeta PCI-Serial usada para el envío y recepción de datos.**

que nos permitiera lograr un equilibrio entre la velocidad de transmsión y los tiempos entre envío de tramas de datos.

Fuente: *Autores del proyecto.*

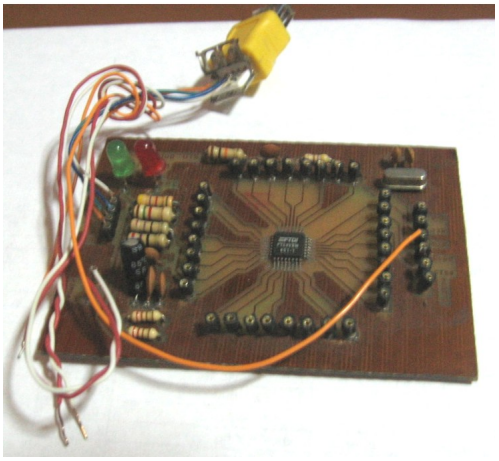


Figura 5.8: **Tarjeta USB utilizada para las primeras pruebas del maestro.**

5.2.3. Implementación de las Fases del Maestro

En esta sección se explicará brevemente cada una de las fases del maestro *AS-interface* y la implementación de cada una de ellas en LabVIEW®, teniendo en cuenta las Listas de Control, las tramas de datos enviadas y las posibles respuestas de esclavo visualizadas.

Fase *Offline*

De acuerdo a la figura 5.9 la fase *Offline* arranca después de activar las banderas *Master Power ON* (MPO), *Offline*, *AS-Interface Power Fail* (APF), Set_Permanent_Configuration, Set_LPS, y luego de cambiar desde un modo configuración a un modo protegido. Las siguientes listas, arreglos y banderas que son iniciadas en la fase, se muestran a continuación en la Tabla 5.2:

Tabla 5.2: **Listas, arreglos y banderas del maestro que son iniciadas durante la Fase *Offline*.**

Ítem a ser inicializado	**Valor**
Lista de Esclavos Detectados (LDS)	Todos los bits 0
Lista de Esclavos Activados (LAS)	Todos los bits 0
Lista de Fallas Periféricas (LPF)	Todos los bits 0
Imágen de Datos de Entrada (IDI)	Todos los bits 0
Imágen de Datos de Salida (ODI)	Todos los bits 1
Imágen de Parámetro (PI)	Parámetros proyectados después de la activación del maestro sin cambios
Imágen de Datos de Configuración (CDI)	Todos los bits 1
Bandera *Config_OK*	Puesta a 0

La Imágen de Datos de Salida (ODI) es iniciada en F_{Hex} debido a que ésta es precisamente la trama de bits por defecto de la salida de los esclavos. Desde el punto de vista del controlador, este estado de inactividad es visto como 0 porque la Imágen de Datos de Salida es invertida por la función de control de ejecución denominada *Write_ODI*.

Luego de iniciada la Fase Offline, la bandera *Offline_ready* es activada y todos los esclavos son llevados a 0 con un *broadcast*. Posterior a la inicialización, todas las funciones de control de ejecución serán habilitadas y la bandera *Offline_ready* se

Fuente: *Ver ref [2]. Modificado por los autores del proyecto.*

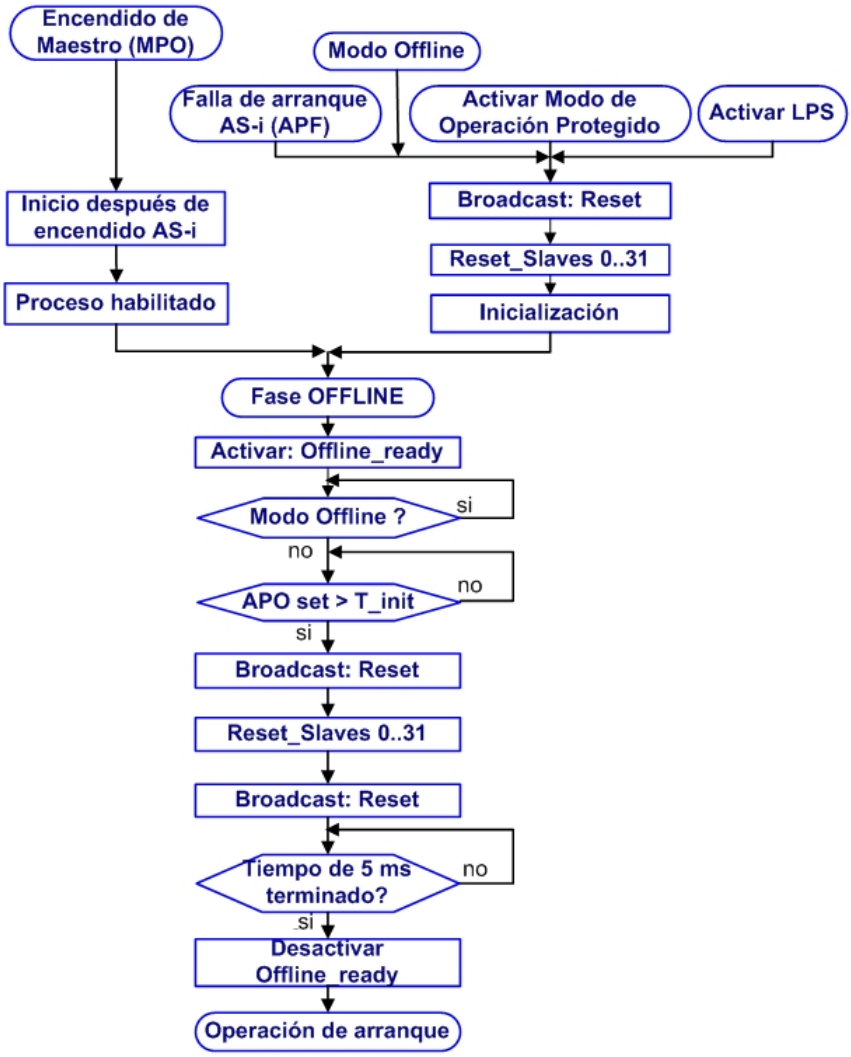


Figura 5.9: **Fase *Offline***

activará.

La bandera *AS-Interface Power On* (APO) representa un intervalo de tiempo (ver figura 5.10) y se activará cuando haya transcurrido más de un tiempo inicial (T_init) igual a 1 segundo. Esta es escogida para asegurar que todos los esclavos conectados han sido localmente inicializados y que han sido habilitados para enviar y recibir datos. Después que este T_init ha expirado, el control de ejecución llevará a cero a todos los esclavos por una petición *broadcast*.

La primera fase está compuesta por las primeras 9 estructuras *case* como se puede ver en el diagrama de bloques del programa principal implementado en LabVIEW®.

Fuente: *Autores del proyecto.*

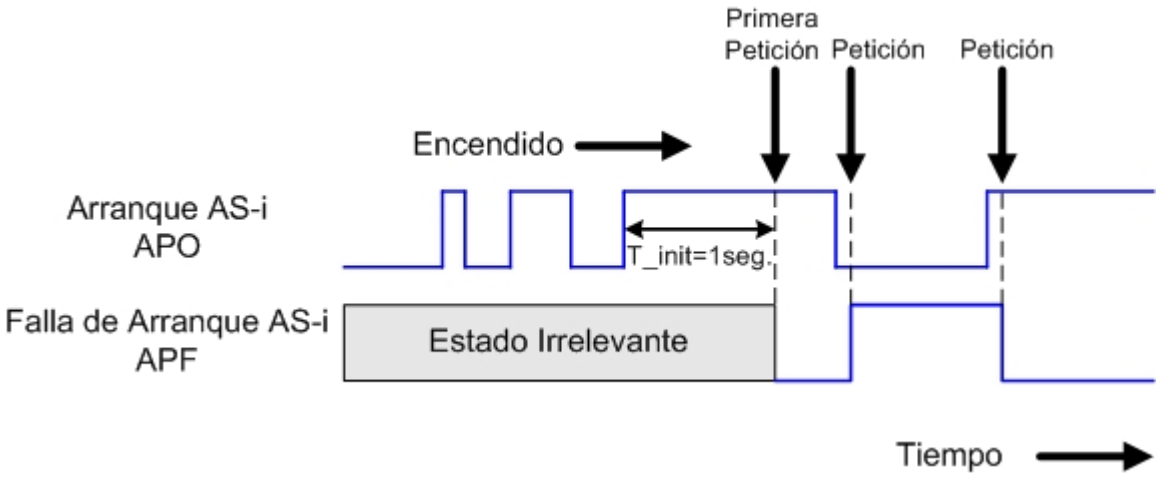


Figura 5.10: **Indicación de falla en el arranque *AS-i*.**

En esta fase del maestro se hace el envío de las primeras peticiones del maestro, las cuáles son: *Broadcast(Reset)* y *Reset_Slave*[5]. A continuación se muestra cómo se ha hecho la manipulación de los datos con el fin de facilitar el envío de los mismos al microcontrolador.

En la figura 5.11(a) se puede observar la trama *AS-i* normal para una petición como *Broadcast(Reset)*. Para efectos de la posterior manipulación de los datos de dirección e información, ésta es invertida para que dichos datos queden de forma ascendente, es decir, para que vayan desde A0 hasta A4 en el caso de la dirección y de I0 a I4 en el caso de la información. En la figura 5.11(b) se muestra cómo es convertida la trama en dos *bytes* que son enviados en forma hexadecimal y que son sacados por el puerto serie, teniendo en cuenta el bit de inicio y el bit final que por defecto se ubican en la trama visualizada en el puerto.

En el siguiente esquema (ver figura 5.12), se puede observar cómo fué construído el datagrama en LabVIEW®, además de cómo se enviaron los datos al puerto serie de la tarjeta PCI-Serial. Cabe aclarar que los datos que llegan al microcontrolador son llevados a su orden original para su posterior envío a la red *AS-interface*. También se puede apreciar como es calculada la paridad en todos los datagramas utilizados en las fases del maestro.

En el caso del datagrama de 14 *bits Broadcast(Reset)* se puede observar que fué dividido en dos partes de 7 *bits*, que junto a otros bits colocados por defecto forman

[5]las tramas de datos se han dejado con los nombres originales utilizados en la norma

Fuente: *Autores del proyecto.*

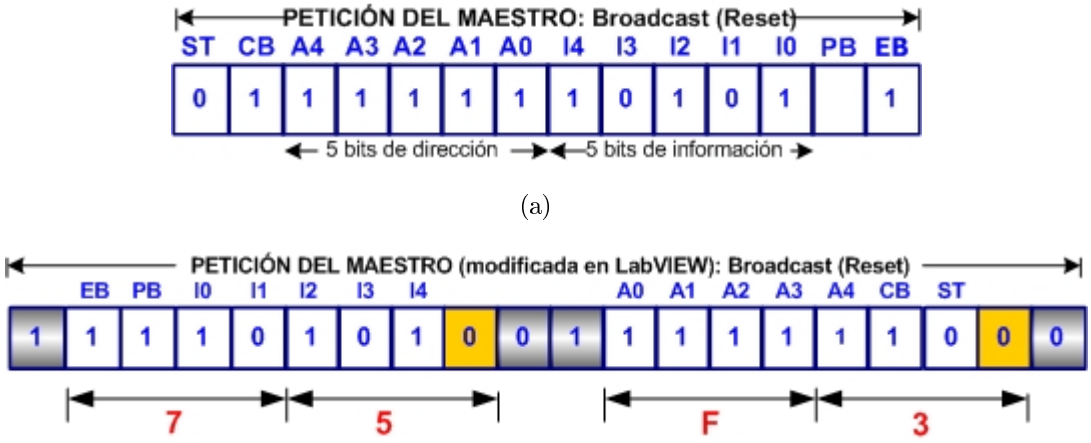


Figura 5.11: **Datagrama de la petición *Broadcast(Reset).***

Fuente: *Autores del proyecto.*

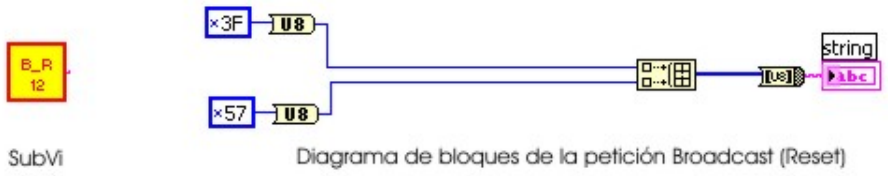


Figura 5.12: **Diagrama de bloques de la petición *Broadcast(Reset).***

una única trama de datos que es visualizada de la siguiente forma: un *bit* inicio puesto en uno que viene por defecto, siete bits de datos de la trama *AS-i*, un bit llevado a cero que es puesto para completar los 8 bits de datos enviados desde el programa y un último bit llevado a cero por defecto también. En total son 20 *bits* que pueden ser visualizados en el osciloscopio en el mismo orden que aparece en la figura 5.11(b).

A continuación se muestra en la figuras 5.13(a) y 5.13(b) el segundo datagrama utilizado en la fase *OFFLINE* denominado *Reset_ Slave*, cuyo direccionamiento cíclico permite encuestar a todos los esclavos para llevarlos a sus valores por defecto y así poder iniciar el proceso de comunicación. Luego se observa cómo se construyó la trama teniendo en cuenta el cambio variable del bit de paridad, el cuál fué calculado gracias al SubVi denominado Validar_Paridad (V_P)que es mostrado en la figura 5.14.

Fuente: *Autores del proyecto.*

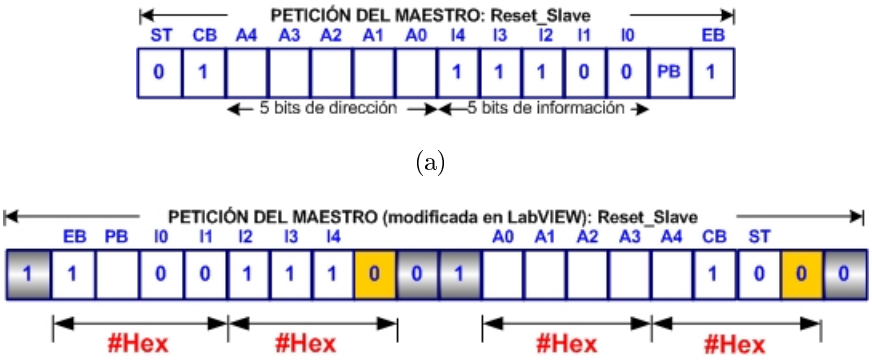


Figura 5.13: **Datagrama de la petición *Reset_ Slave*.**

Fuente: *Autores del proyecto.*

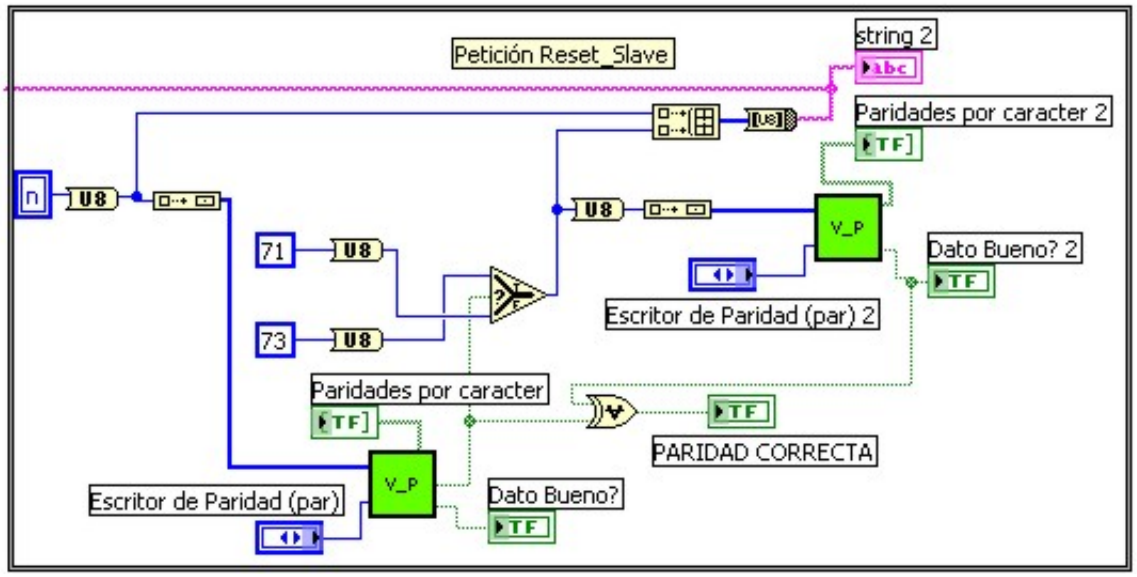


Figura 5.14: **Diagrama de bloques de la petición *Reset_ Slave*.**

Fase de Detección

En esta fase el maestro busca los esclavos conectados a la línea *AS-i* como lo muestra la figura 5.15 y estos son ubicados en la Lista de Esclavos Detectados (LDS) como lo muestra la figura 5.16. La configuración de Entradas/Salidas y los códigos de identificación son almacenados en la Imagen de Datos de Configuración (CDI) como se observa en la figura 5.17.

Fuente: *Ver ref [2]. Modificado por los autores del proyecto.*

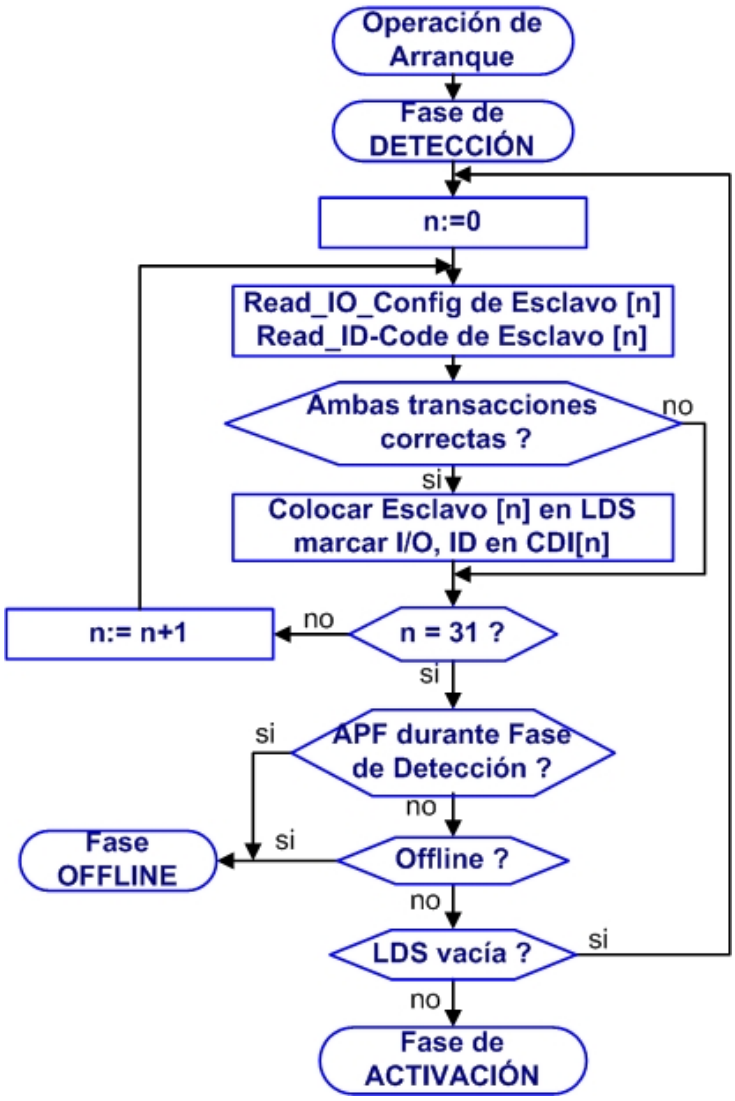


Figura 5.15: **Fase de Detección.**

El control de ejecución busca al menos una vez para cada esclavo en el rango de direcciones y efectúa una petición que se utiliza para leer los datos de configuración de Entradas/Salidas (Petición *Read_I/O_Configuration*) y una petición para leer el código de identificación (Petición *Read_ID_Code*).

- Si no se recibe una respuesta válida a la peticiones Read_I/O_Configuration y Read_ID_Code, la búsqueda continuará con la siguiente dirección de esclavo.
- Si por el contrario, el control de ejecución recibe ambas respuestas válidas a las peticiones antes mencionadas, entonces el esclavo será incluido en la Lista de Esclavos Detectados (LDS). Asimismo, los datos de configuración de En-

Fuente: *Autores del proyecto.*

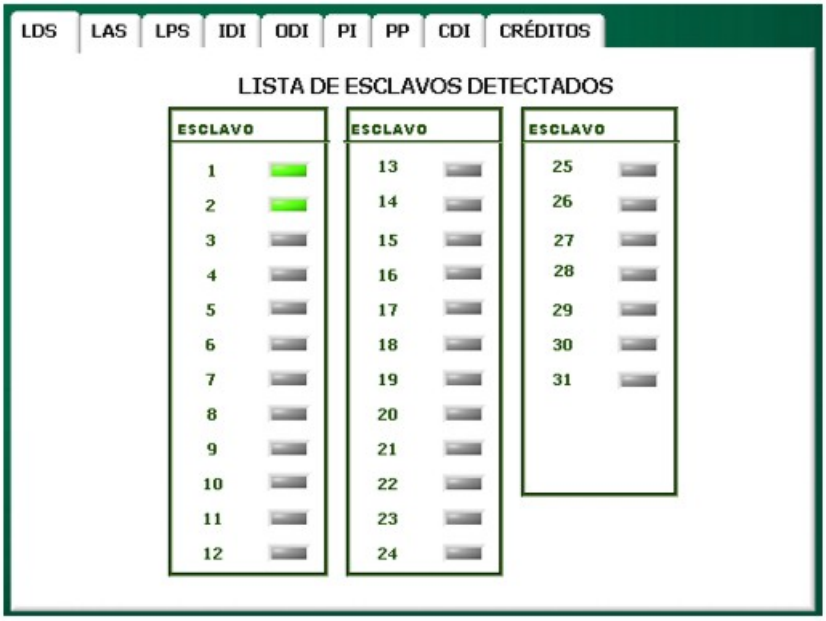


Figura 5.16: **Lista de Esclavos Detectados (LDS).**

Fuente: *Autores del proyecto.*

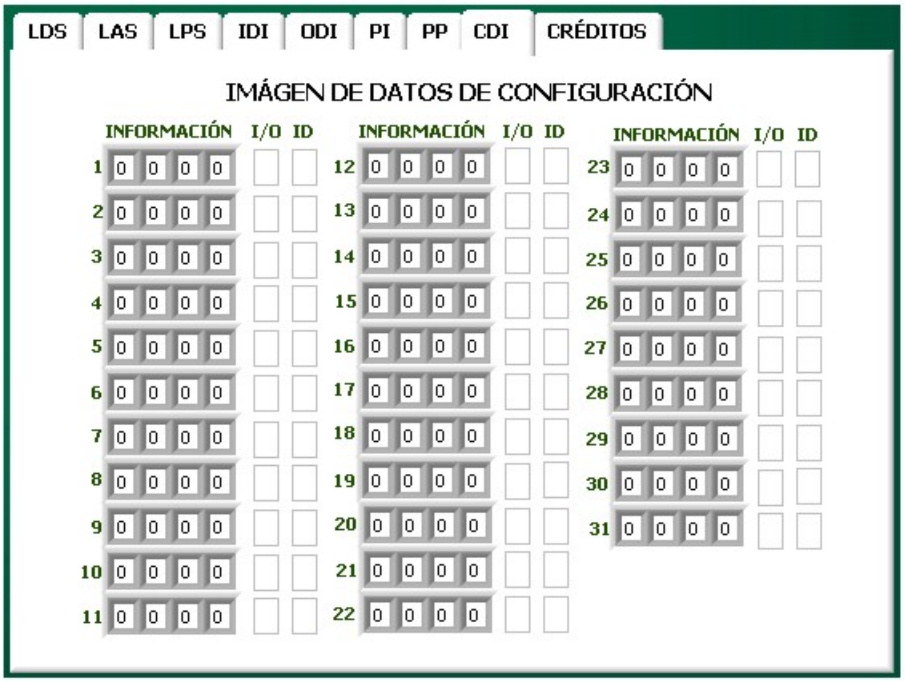


Figura 5.17: **Imagen de Datos de Configuración (CDI).**

tradas/Salidas y el código de identificación de los esclavos serán almacenados en la Imagen de Datos de Configuración (CDI).

Después que se han encuestado todas las direcciones de esclavo de la manera descrita anteriormente, el control de ejecución chequea los contenidos de la Lista de Esclavos Detectados (LDS) y de la bandera *AS-interface Power Fail* (APF). Si APF está activa durante la Fase de Detección, el control de ejecución cambia a la Fase *Offline*. Si LDS está vacía, entonces se retoma la búsqueda de nuevos esclavos repitiendo la Fase de Detección. De otra manera el control de ejecución cambia a la Fase de Activación.

En cuanto a la implementación en LabVIEW® de esta fase, se puede decir que está desarrollada desde el *case* 10 al *case* 20 y que la componen las peticiones de maestro Read_I/O_Configuration y Read_ID_Code las cuáles son mostradas con sus respectivos diagramas de bloques en las figuras 5.18(a), 5.18(b), 5.19, 5.20(a), 5.20(b) y 5.21.

Fuente: *Autores del proyecto.*

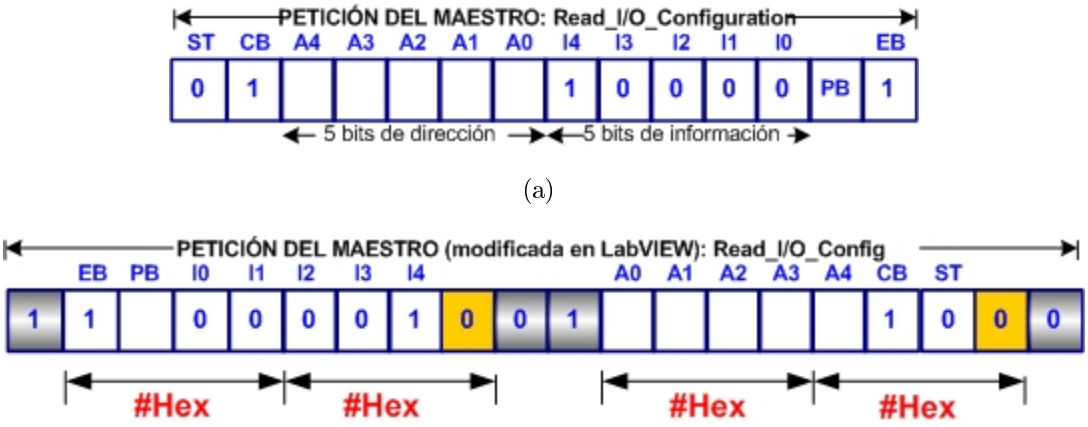


Figura 5.18: **Datagrama de la petición *Read_ I/O_ Configuration*.**

Fase de Activación

Los esclavos encontrados en la Fase de Detección son activados escribiendo el parámetro desde la Imagen de Parámetros (PI) a los esclavos. La activación de los esclavos depende del Modo de Operación, ya sea Protegido o de Configuración como lo muestra la figura 5.22. Además, un esclavo que no está activado no acepta ninguna petición de intercambio de datos.

En el Modo Protegido solo aquellos esclavos que sean miembros de la Lista de

Fuente: *Autores del proyecto.*

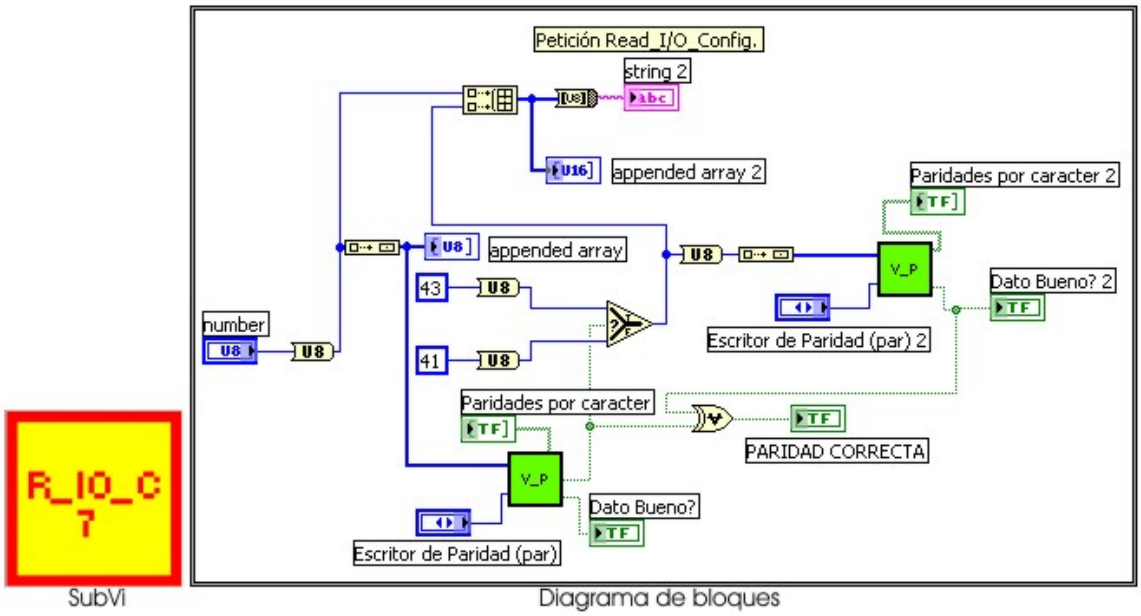


Figura 5.19: **Diagrama de bloques de la petición *Read_ I/O_ Config.***

Fuente: *Autores del proyecto.*

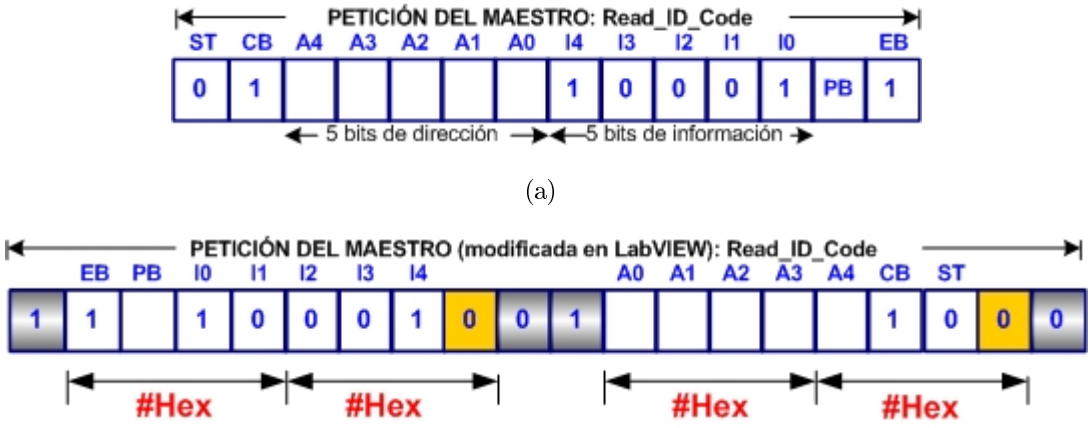


Figura 5.20: **Datagrama de la petición *Read_ ID_ Configuration.***

Esclavos Detectados (LDS) y la Lista de Esclavos Proyectados (LPS) (ver figura 5.23) y, aquellos cuya Imagen de Datos de Configuración (CDI) tengan igual valor a el Dato de Configuración Permanente (PCD) serán activados. Por otra parte en el Modo de Configuración, todos los esclavos, excluyendo el esclavo con dirección cero, los cuáles son miembros de la Lista de Esclavos Detectados (LDS) serán activos.

Fuente: *Autores del proyecto.*

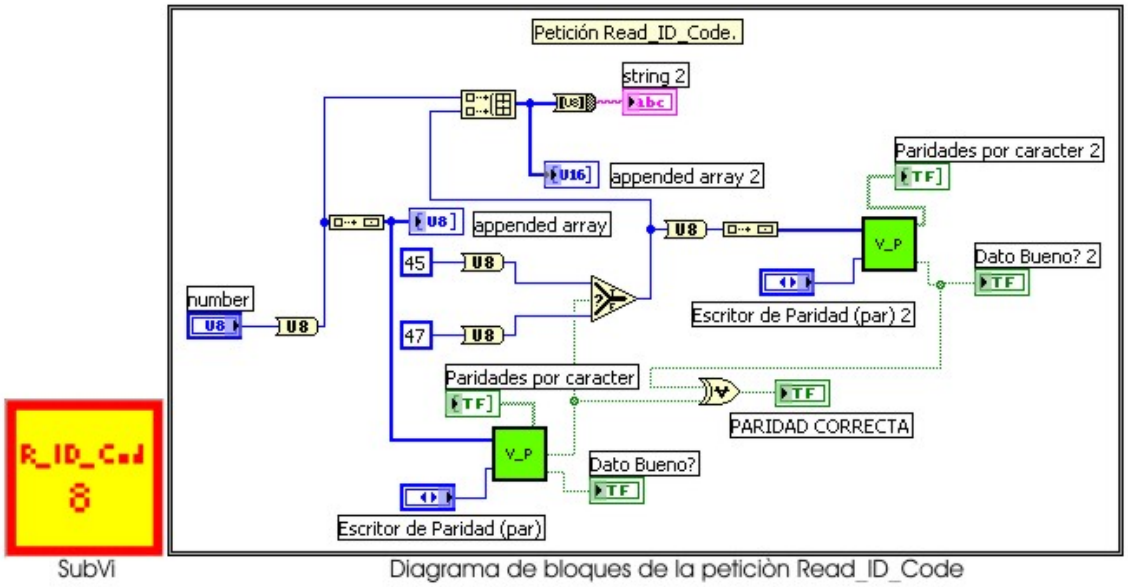


Figura 5.21: **Diagrama de bloques de la petición *Read_ID_Code*.**

La activación de un esclavo específico es efectuada escribiendo los valores de parámetros a un esclavo específico desde la Imagen de Parámetros (PI) con la petición de maestro denominada *Write_Parameter*.

Si la respuesta a la petición *Write_Parameter* es positiva (*Data_OK*), el control de ejecución comprueba la bandera *Data_Exchange_Active*.

Si la bandera *Data_Exchange_Active* es activada, el contenido del elemento Imagen de Datos de Salida (ODI) que se muestra en la figura 5.24 del esclavo será enviado al maestro al utilizar la petición de maestro *Data_Exchange* cuya trama y diagrama de bloques implementado en LabVIEW® se muestran en las figuras 5.25(a), 5.25(b) y 5.26. Si el dato de respuesta recibida desde el esclavo es válido (*Data_OK*), entonces los valores recibidos se mueven a la Imagen de Dato de Entrada (IDI) y la entrada para el esclavo en la Lista de Esclavos Activos (LAS) es almacenada como se puede ver en la figura 5.27. Si el dato de respuesta recibido es inválido, entonces el esclavo es borrado de la Lista de Esclavos Detectados (LDS). Si la bandera no es activada, la entrada para el esclavo en la Lista de Esclavos Activos (LAS) será activada.

Si la respuesta a la petición *Write_Parameter* es negativa (*Data_NOK*), entonces el esclavo es borrado como un miembro de la Lista de Esclavos Detectados (LDS).

Fuente: *Ver ref [2]. Modificado por los autores del proyecto.*

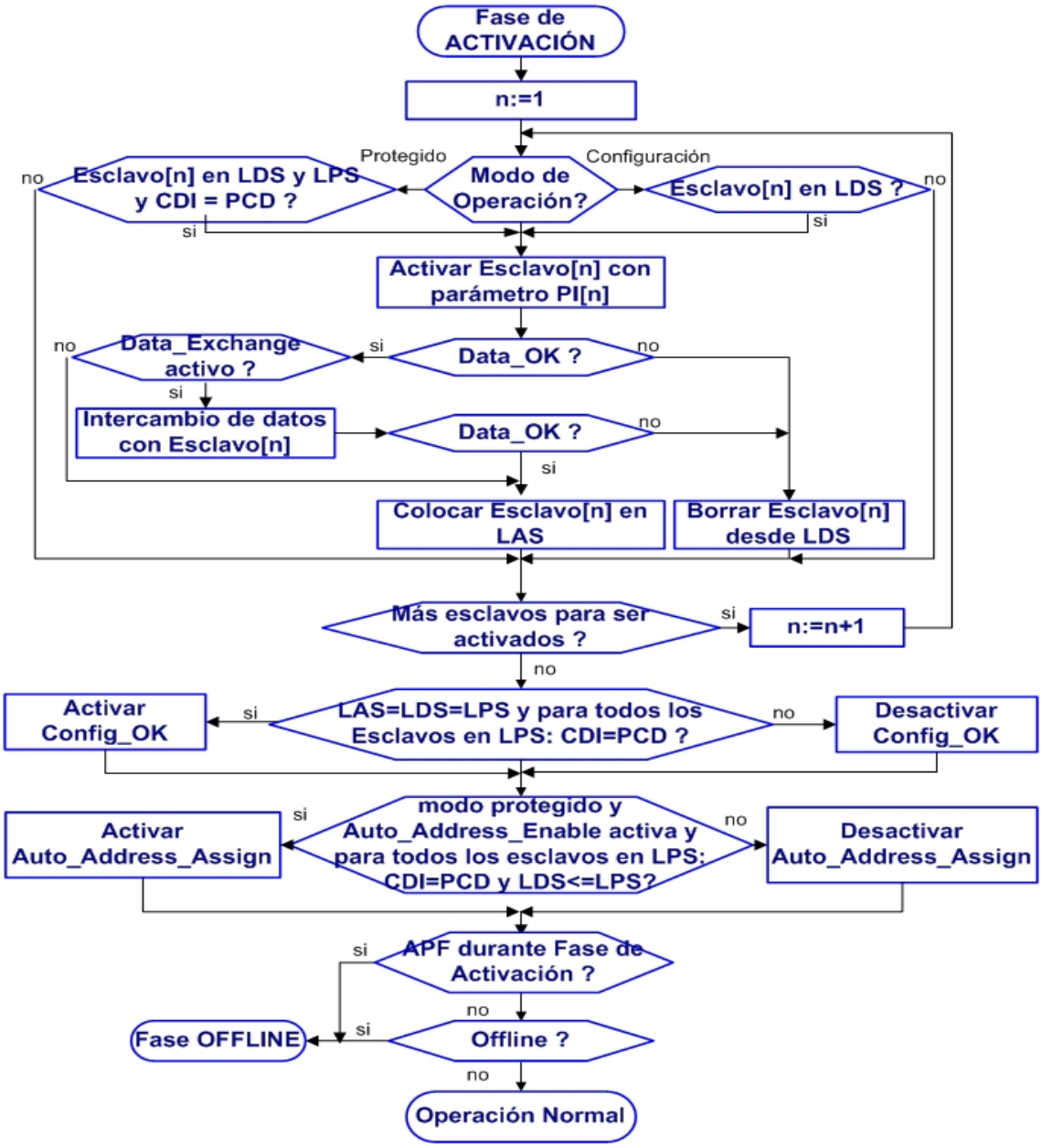


Figura 5.22: **Fase de Activacion.**

Antes de dejar la Fase de Activación las banderas *Config_OK* y *Auto_Address_Assign* son actualizadas.

$\Longrightarrow$ La bandera *Config_OK* es activada si:

El contenido de la Lista de Esclavos Proyectados (LPS) y la Lista de Esclavos Detectados (LDS) y la Lista de Esclavos Activos (LAS) son idénticos. Y si el Dato de Configuración Permanente (PCD) es idéntico con la Imagen de Datos de Configuración (CDI) para todos los esclavos de la Lista de Esclavos Proyectados (LPS).

Fuente: *Autores del proyecto.*

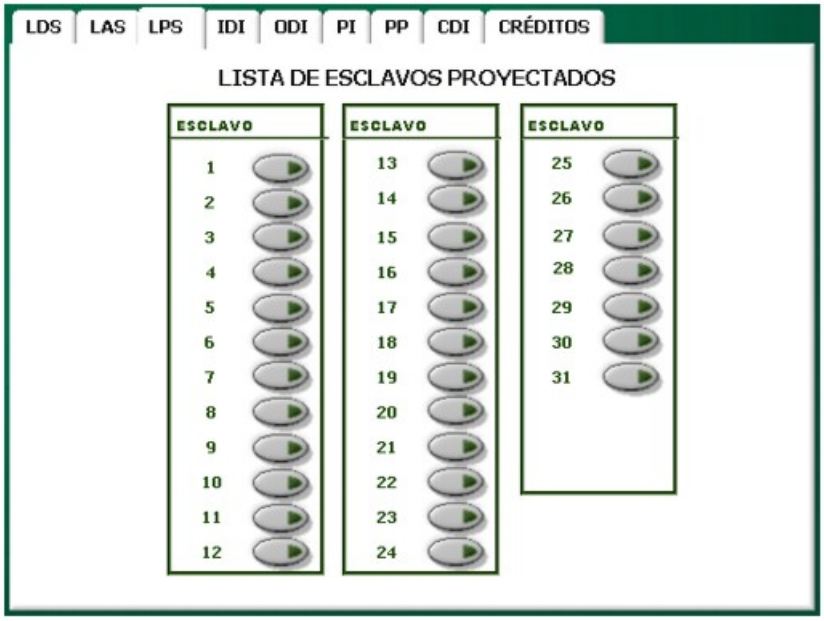


Figura 5.23: **Lista de Esclavos Proyectados (LPS).**

Fuente: *Autores del proyecto.*

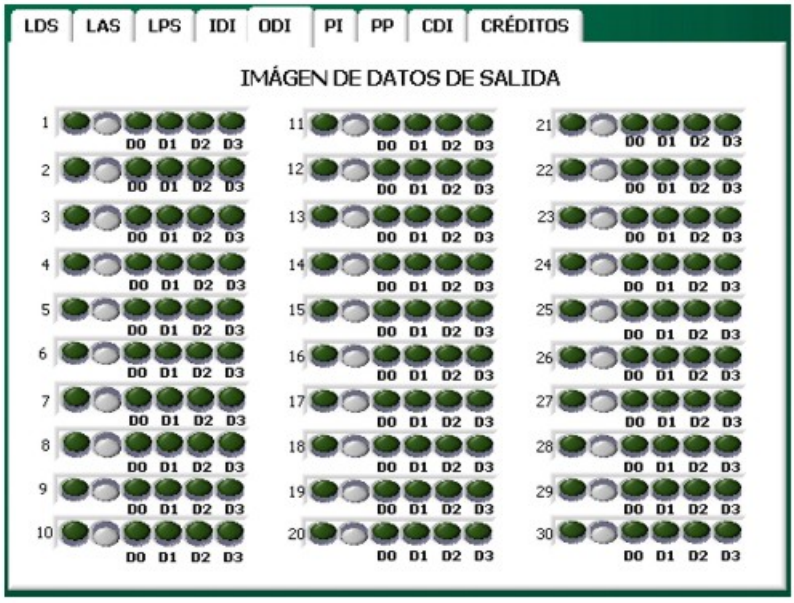


Figura 5.24: **Imágen de Datos de salida (ODI).**

$\Longrightarrow$ La bandera *Auto_ Address_ Assign* es actualizada en Modo Protegido solamente y es activada si:

El bit *Auto_ Adress_ Enable* está activo. Asimismo si la Lista de Esclavos Detectados

Fuente: *Autores del proyecto.*

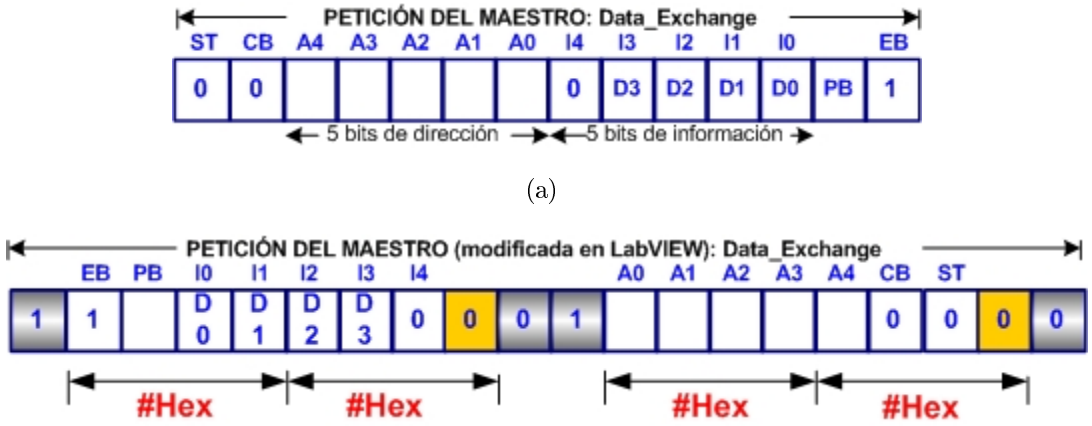


Figura 5.25: **Datagrama de la petición *Data_ Exchange*.**

Fuente: *Autores del proyecto.*

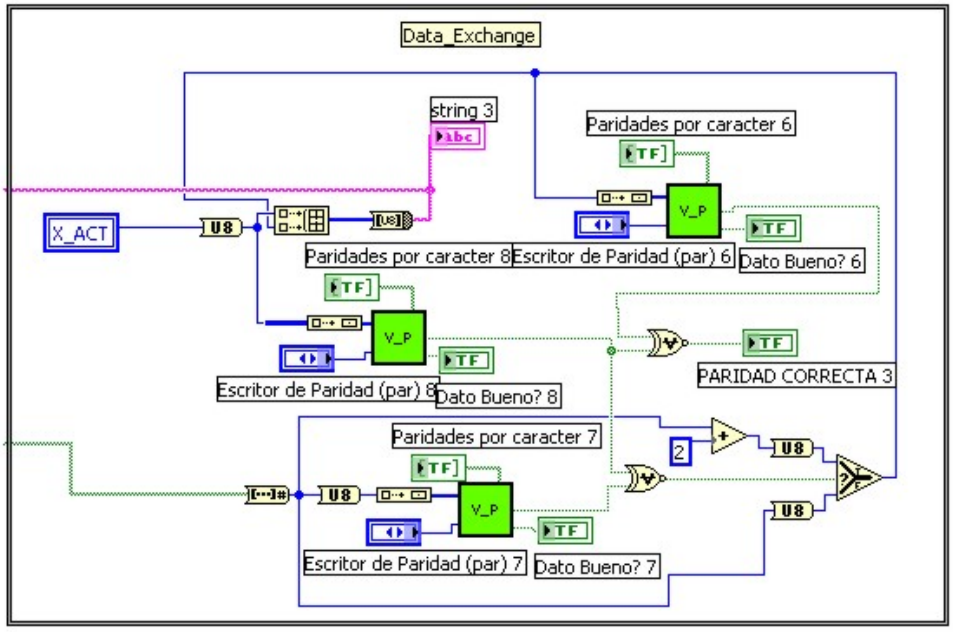


Figura 5.26: **Diagrama de bloques de la petición *Data_ Exchange*.**

(LDS) contiene un número de esclavos que es menor o igual al número de esclavos ubicados en la Lista de Esclavos Proyectados (LPS), y si para todos los esclavos excepto para el esclavo cero de la Lista de Esclavos Detectados (LDS) el Dato de Configuración Permanente (PCD) es idéntico a la Imagen de Dato de Configuración (CDI).

$\Longrightarrow$ La bandera *Auto_ Address_ Assign* se desactiva para cerrar la asignación de direcciones automática y si:

Fuente: *Autores del proyecto.*

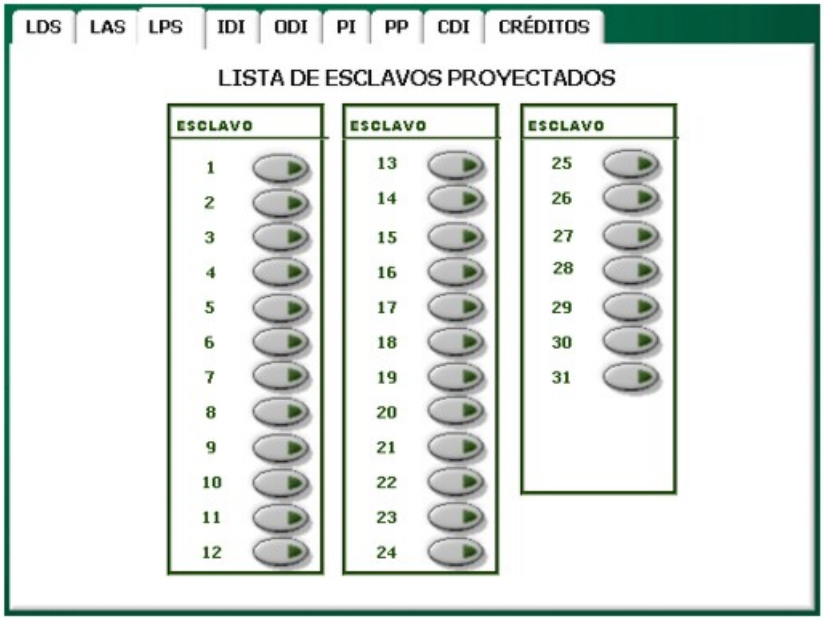


Figura 5.27: **Lista de Esclavos Activos (LAS).**

El bit *Auto_Adress_Enable* se desactiva ó si la Lista de Esclavos Detectados (LDS) contiene más esclavos que la Lista de Esclavos Proyectados (LPS) ó si, para al menos un esclavo excepto el esclavo cero de la Lista de Esclavos Detectados (LDS), el Dato de Configuración Permanente (PCD) no corresponde con la Imagen de Dato de Configuración (CDI).

Si *AS_Interface_Power_Fail* (APF) permanece inactiva durante la Fase de Activación, el control de ejecución cambia a la etapa de Intercambio de Datos.

Por último es de vital importancia mencionar que esta fase va desde el *case* 21 hasta el 41 en la máquina de estados implementada en el programa.

Fase de Intercambio de Datos

Esta fase se puede observar en su totalidad entre el *case* 42 y el *case* 61 del programa total implemantado para el manejo de la máquina de estados del protocolo *AS-Interface*. Previamente al Intercambio de Datos, el control de ejecución probará la bandera *Data_Exchange_Active*.

Si es VERDADERA (Activa), entonces el control de ejecución enviará el contenido

Fuente: *Ver ref [2]. Modificado por los autores del proyecto.*

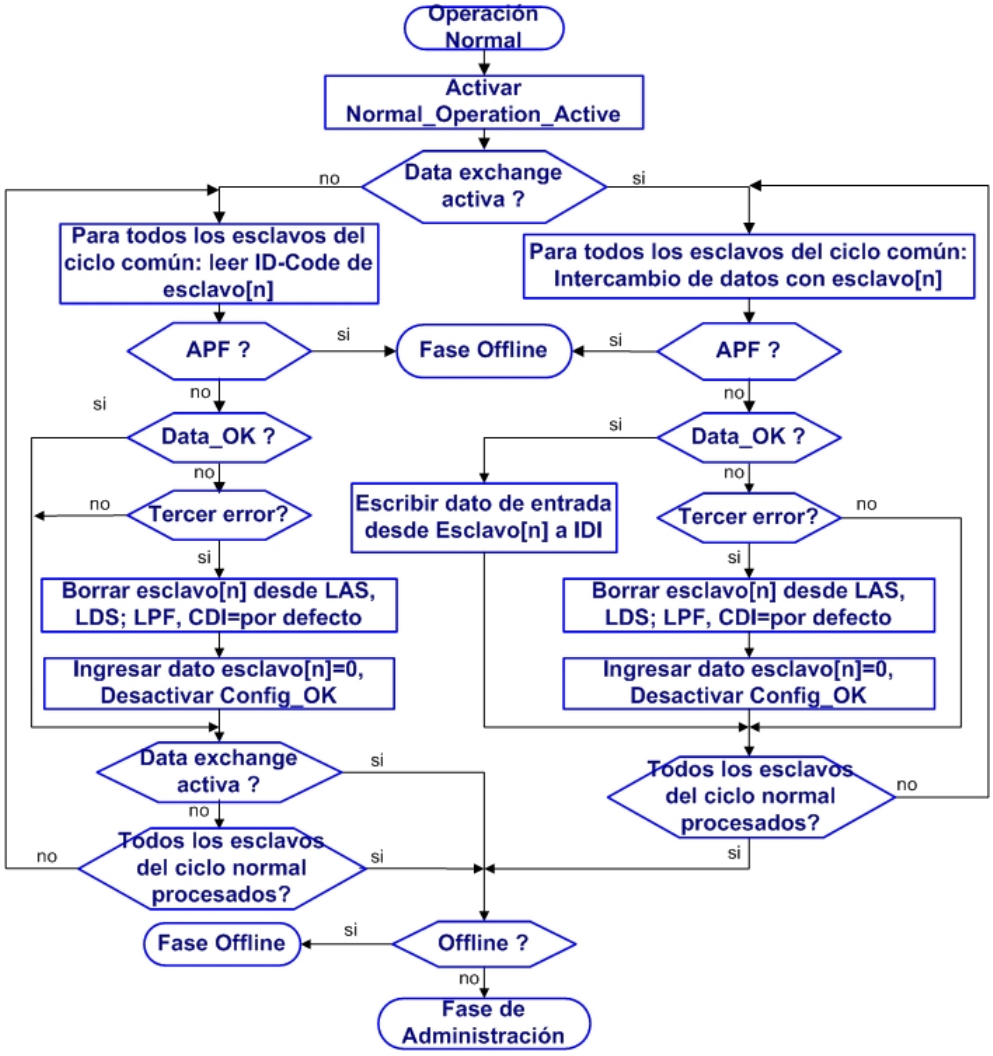


Figura 5.28: **Fase de Intercambio de Datos.**

del elemento Imagen de Datos de Salida (ODI) por medio de la petición de maestro *Data_Exchange* a todos los esclavos que sean miembros de la Lista de Esclavos Activos (LAS), iniciando con el esclavo con la dirección más baja y en orden de menor a mayor. Además, el esclavo direccionado responderá con el dato actual desde el registro de transmisión del esclavo.

Si el dato de respuesta recibido no es válido (*Data_NOK*) en tres fases de intercambios de datos consecutivas, entonces el esclavo direccionado será removido de la Lista de Esclavos Activos (LAS) y de la Lista de Esclavos Detectados (LDS), y los elementos de la Imagen de Datos de Configuración (CDI) y la Lista de Fallas Periféricas (LPF) son activados con sus valores por defecto. Asimismo los elementos del arreglo de la Imagen

de Dato de Entrada (IDI) como se observa en la figura 5.29 serán llevados a 0, y por consiguiente la bandera *Config_ OK* será desactivada como lo muestra la figura 5.28.

Fuente: *Autores del proyecto.*

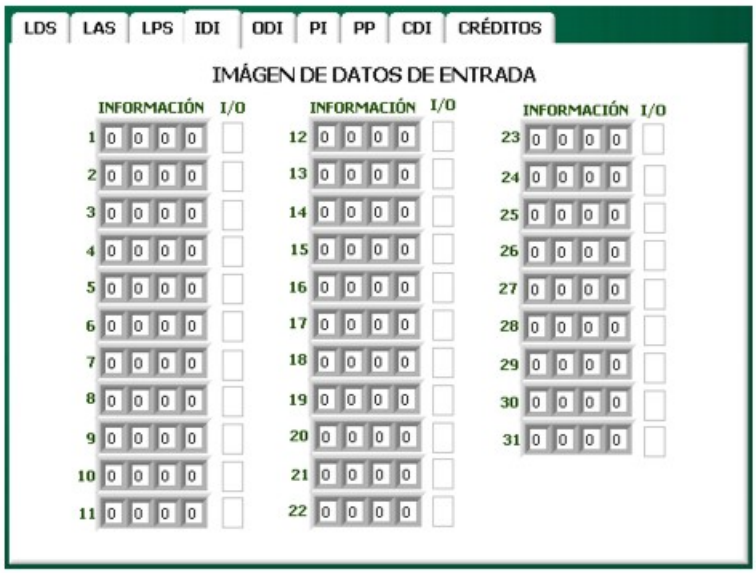


Figura 5.29: **Imágen de Datos de Entrada (IDI).**

Si el dato de respuesta recibido es válido (*Data_ OK*), entonces los valores recibidos son llevados a la Imagen de Datos de Entrada (IDI).

Si la bandera *Data_ Exchange_ Active* es FALSA, el control de ejecución no enviará una petición de Intercambio de datos, peroverifica si todos los esclavos marcados en la Lista de Esclavos Activos (LAS) están activos, leyendo los códigos de identificación de todos los esclavos de la Lista de Esclavos Activos (LAS) en orden ascendente con la petición de maestro *Read_ ID_ Code*. Antes de enviar la enterior petición, se verifca la bandera *Data_ Exchange_ Active*. Si esta bandera es activada, entonces el control de ejecución inmediatamente cambiará a la Fase de Administración.

Si el control de transmisión responde en tres fases de intercambio de datos consecutivas con *Data_ NOK*, entonces este esclavo es borrado de la Lista de Esclavos Activos (LAS) y de la Lista de Esclavos Detectados (LDS). Los elementos de la Imagen de Datos de Configuración (CDI) y la Lista de Fallas Periféricas (LPF) para este esclavo serán llevados a sus valores por defecto, y los elementos de la imágen de Datos de Entrada (IDI) son llevados a 0 y la bandera *Config_ OK* es desactivada.

La bandera *Data_ Exchange_ Active* puede ser utilizada para iniciar la sin-

cronización entre el maestro y el controlador. También se usa para deshabilitar el intercambio de datos durante el inicio del controlador.

Si la función de transmisión detecta APF durante la Fase de Intercambio de Datos, entonces el control de ejecución cambiará a la Fase *Offline* sin llegar al fin de la Fase de Intercambio de Datos que esté en curso.

Después del proceso de la Fase de Intercambio de Datos, el control de ejecución cambiará a la Fase de Administración.

Fase de Administración

Durante la Fase de Administración (ver figura 5.30), los comandos pueden ser emitidos para que soporten las funciones remotas listadas en la tabla 5.3. Además, los comandos emitidos durante esta fase son controlados por la administración, y si no hay peticiones para un intercambio de comandos, esta fase cambia directamente a la Fase de Inclusión. Esta fase fué implementada entre el *case* 62 y el *case* de la máquina de estados total implementada en LabVIEW®.

Tabla 5.3: **Funciones remotas del maestro.**

Funciones remotas (Intercambio de mensajes sobre la línea *AS-i*)
Write_ Parameter
Address_ Assign
Comandos:
Reset_ Slave
Delete_ Address
Read_ I/O_ Configuration
Read_ ID_ Code

Fase de Inclusión

Durante la Fase de Inclusión el control de ejecución lleva una petición al control de transmisión. La inclusión de un esclavo necesita de cuatro a siete (dependiendo del perfil del maestro) peticiones de maestro que son procesadas. De acuerdo al perfil del maestro que no soporta códigos de identificación extendidos y fallas periféricas, la Fase de Inclusión puede manejarse en cuatro ciclos consecutivos como lo muestra la figura 5.31 y es manejada en la programación de la máquina de estados implementada desde

Fuente: *Ver ref [2]. Modificado por los autores del proyecto.*

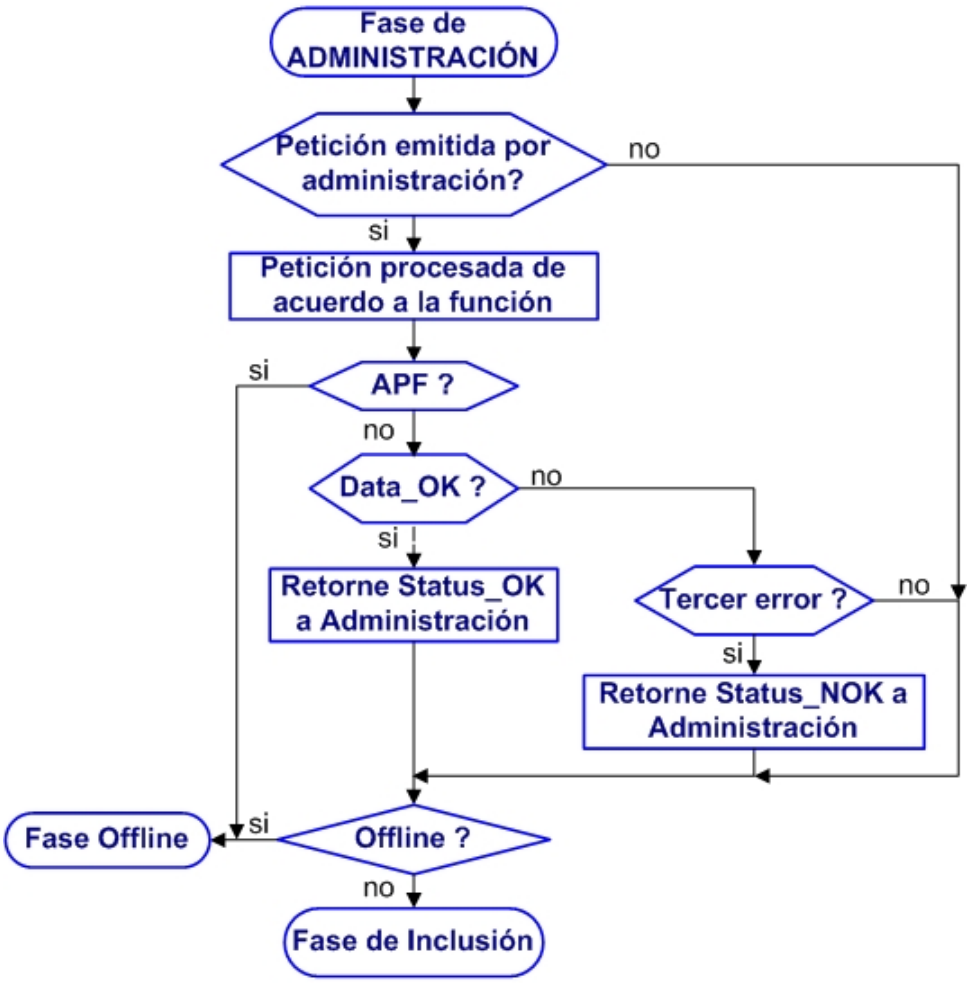


Figura 5.30: **Fase de Administración.**

el *case* 71 al *case* 106.

Para finalizar la implementación del Maestro *AS-Interface* en LabVIEW® cabe aclarar que se trata de respetar cada una de las fases del protocolo según la norma. Las modificaciones hechas a éstas, son efectuadas teniendo en cuenta que el maestro no sufra ningún cambio que altere el intercambio de datos con los esclavos de la red, es decir, que cumpla con las principales características mencionadas en la norma IEC-62026-2 [2].

Fuente: *Ver ref [2]. Modificado por los autores del proyecto.*

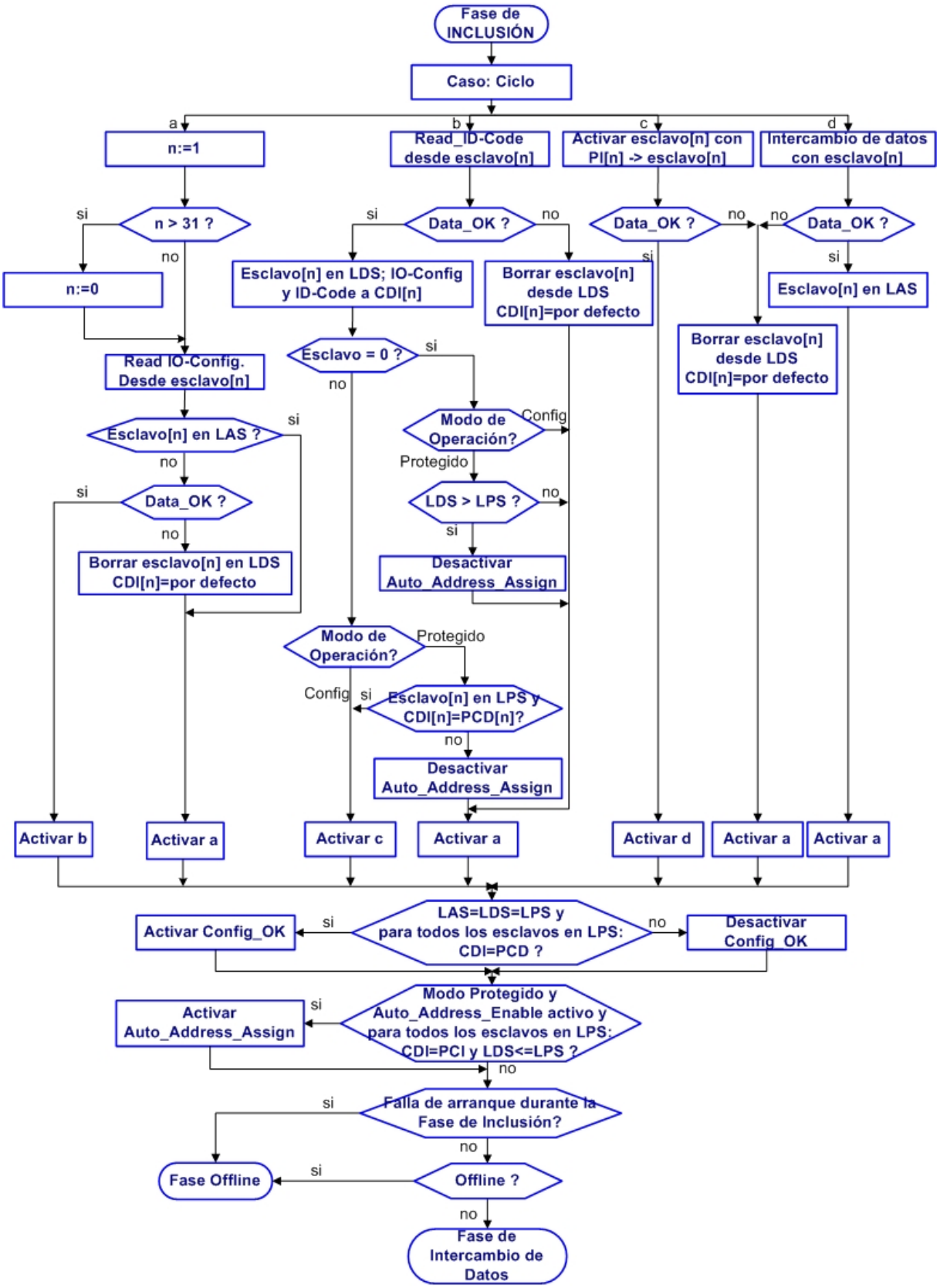


Figura 5.31: **Fase de Inclusión.**

Capítulo 6

PRUEBAS Y RESULTADOS

En este capítulo se hace la presentación de las pruebas y resultados obtenidos a lo largo del proceso de implementación del maestro *AS-i*, a nivel de *hardware* y *software*. La gran parte de estas pruebas fueron desarrolladas en una red *AS-Interface* comercial con el fin de hacer un análisis detallado, en tiempo real, del desempeño del maestro y la respuesta de los esclavos de la red, según los requerimientos de la norma IEC 62026-2 [2].

6.1. Pruebas de Transmisión de Señales a la Red

Para el desarrollo de un análisis completo de las señales transmitidas a la red *AS-i* comercial, se hace la presentación de algunos patrones de corriente y de volataje ideales, generados en *Matlab* como los mostrados en la figura 6.1 [1], con el fin poder efectuar una comparación con aquellas señales reales que son generadas por el maestro *AS-i*.

Como se ha mencionado en capítulos anteriores, la transmisión es inicialmente generada como una forma de onda de tensión que posteriormente es convertida a corriente y puesta en la red para efectuar intercambio de datos con los esclavos. En el maestro *AS-i* desarrollado, estos patrones de voltaje se han generado por medio del DAC del microcontrolador como lo muestra la figura 6.2(a) en el canal A y posteriormente ésta misma señal se lleva al driver de línea por medio de un circuito de acople (*buffer*) que ha sido mencionado previamente en la sección 4.3 y cuya señal es mostrada en el canal B del osciloscopio. En las gráficas se resalta tanto el T_{BIT} de $6\mu s$ como la tensión de $2V_{P-P}$ que cumplen con los requerimientos de la norma.

Con el desarrollo de las pruebas se analizaron diferentes algoritmos con el fin de llegar a un óptimo desempeño de los recursos del microcontrolador, los cuáles

[1] Estas señales son referenciadas en [1] pag.64

Fuente: *Ver referencia [1] p.64.*

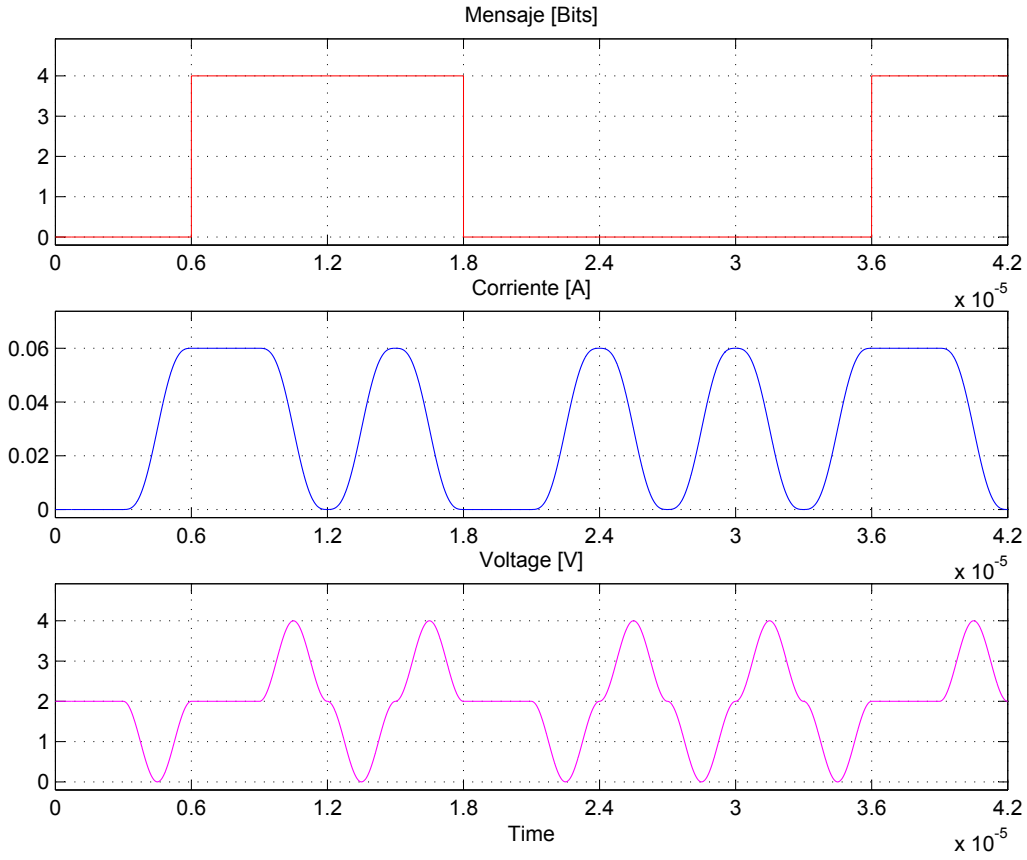
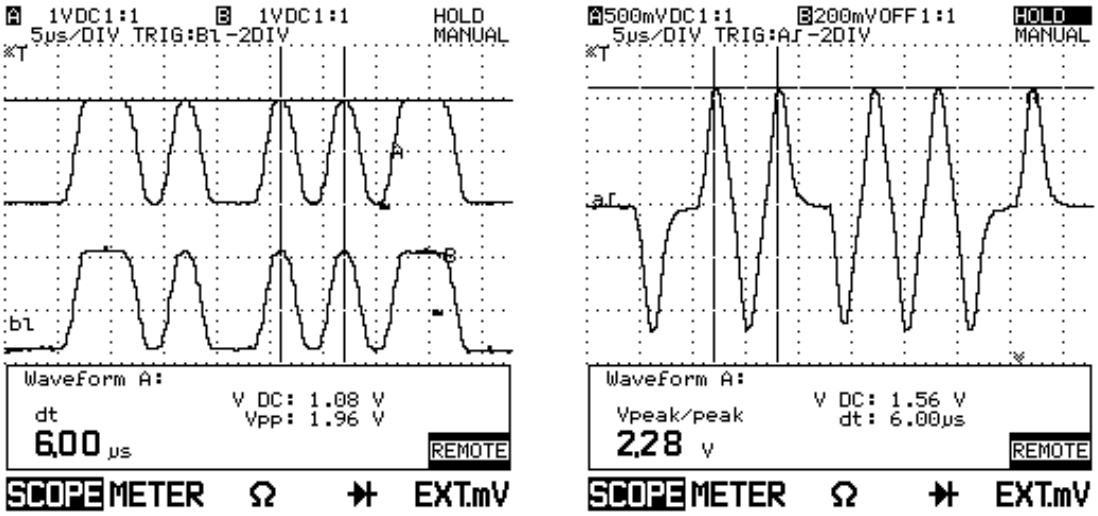


Figura 6.1: **Patrones ideales del arreglo binario** 0110001*b*.

al final del proceso permitieron trabajar a una frecuencia de bus de 22,1184Mhz para garantizar 26 muestras por T_{BIT}. Además, podemos apreciar cómo se generó el datagrama binario 0110001*b* modulado en corriente.

En la figura 6.2(b) se puede observar cómo la onda de tensión es convertida en una señal de corriente aproximadamente de 60mA a través del generador *Howland*, para luego ser inyectada en el bus *AS-i*. La variación en el valor de los picos de voltaje es ocasionada por el circuito de desacople de la fuente de alimentaciónn de la red y las características físicas de la línea de transmisión. Cabe resaltar que en esta señal se siguen garantizando los valores de amplitud y T_{BIT} exigidos por la norma [**?**].

Fuente: *Autores del proyecto.*



(a) (b)

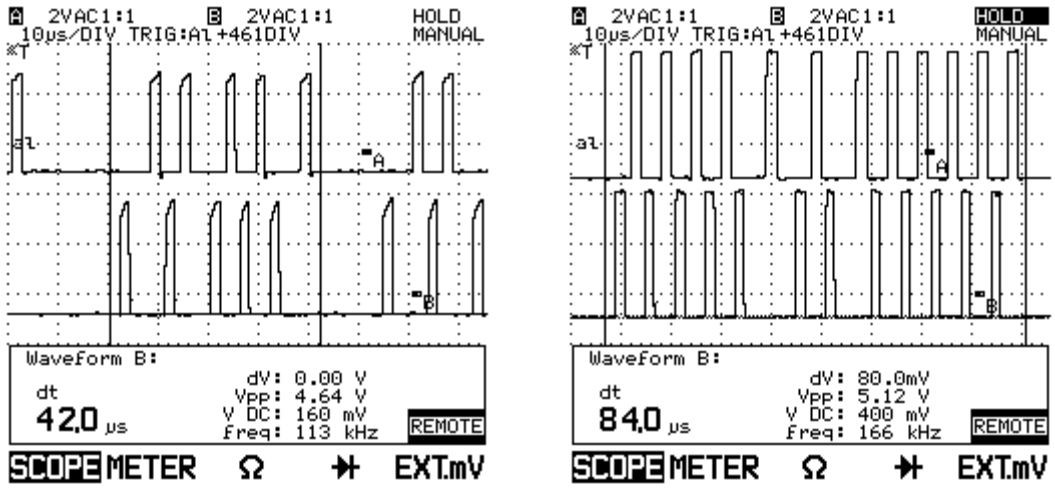
Figura 6.2: **(a) Forma de onda de corriente generada. (b) Forma de onda de voltaje sobre el bus.**

6.2. Pruebas de Recepción de Señales desde la Red

En esta sección se muestran las formas de onda provenientes de la red *AS-i* comercial[2], durante el proceso de recepción de los datos enviados desde los esclavos. En la figura 6.3 se pueden observar las formas de onda de salida de los dos canales de recepción implementados por *hardware* y explicados brevemente en el capítulo 4, para efectuar el análisis de detección tanto de los pulsos positivos (Canal A), como de los pulsos negativos (Canal B), que fueron generados por el proceso de comparación de la señal de voltaje en el bus. La figura 6.3(a) muestra los pulsos detectados para una respuesta de esclavo antes de la etapa de aislamiento digital, donde se resalta la adecuada alternancia de los pulsos, la ventana de tiempo de $42\mu s$ para los 7-*Bits* de la trama, y el inicio y fin del mensage con un pulso negativo y positivo, respectivamente. Asimismo se puede ver en la figura 6.3(b) los pulsos detectados para una petición de esclavo después del proceso de aislamiento digital de la señal, el cual permite una mejora sustancial en la forma de los pulsos, en comparación con los vistos en la figura 6.3(a).

[2]Red ubicada en la Escuela de Ingeniería Mecánica.

Fuente: *Autores del proyecto.*



(a) (b)

Figura 6.3: **(a) Pulsos detectados para una respuesta de esclavo antes de la etapa de aislamiento. (b) Pulsos detectados para una respuesta de esclavo después de la etapa de aislamiento.**

6.3. Pruebas de Comunicación con el PC

Para determinar el funcionamiento de la comunicación entre la red *AS-i* y el PC se realizaron las pruebas que se mencionan a continuación:

La primera prueba consistió en comprobar la transmisión de datos utilizando la tarjeta PCI-serial y un sencillo programa desarrollado en LabVIEW® con el cuál se envió una trama específica de maestro y cuya respuesta del esclavo fué visualizada y verificada mediante un osciloscopio. Para este fin fué necesario utilizar el circuito integrado SN75C3221 fabricado por *Texas Instruments*, el cual ajusta las señales del puerto serial del PC a los niveles lógicos utilizados por el microcontrolador.

En segundo lugar se logró la visualización de la configuración de entradas/salidas (I/O) y los códigos de identificación de 3 de los esclavos ubicados en la red *AS-i*, cuyas direcciones especificas son 01, 02 y 03. Estos datos fueron llevados directamente a la Imágen de Datos de Configuración (CDI) como se muestra en la figura 6.4.

Para finalizar se verificó el intercambio de datos entre el maestro implementado y dos de los esclavos de la red, específicamente la caja de mando cuya dirección actual

Fuente: *Autores del proyecto.*

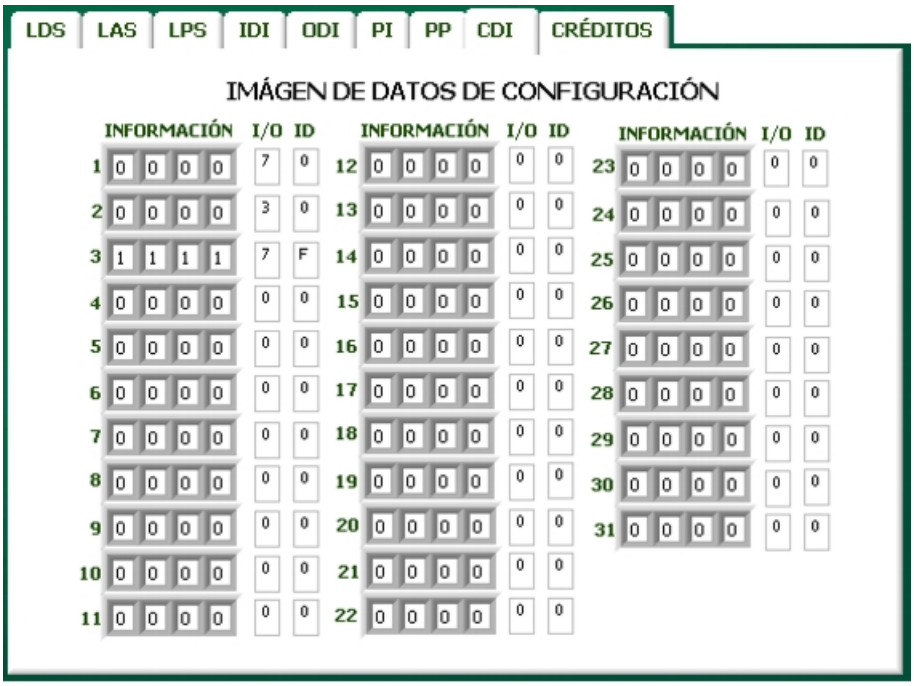


Figura 6.4: **Imágen de Datos de Configuración (CDI) de tres esclavos ubicados en la red *AS-i*.**

es 01 y el módulo de entradas/salidas digitales con dirección 02. En el caso del módulo digital, se pudo observar el correcto funcionamiento del sensor BERO inductivo, al ser ubicado en cualquiera de las entradas. La figura 6.5 muestra la Imágen de Datos de Entrada (IDI) del maestro y las entradas/salidas de los esclavos activos en la red. En su orden, el esclavo número 1 muestra la activación de la caja de mando y el esclavo 2 muestra que la entrada uno está activa, es decir, que el sensor recibió una señal que activa el puerto de entrada, mientras las salidas del esclavo están inactivas. Cabe aclarar que los valores por defecto de los bits que representan las entradas (IN1, IN2) están en 0 cuando están inactivas y los valores de los bits que representan las salidas (OUT1, OUT2) están en 1 por defecto.

Fuente: *Autores del proyecto.*

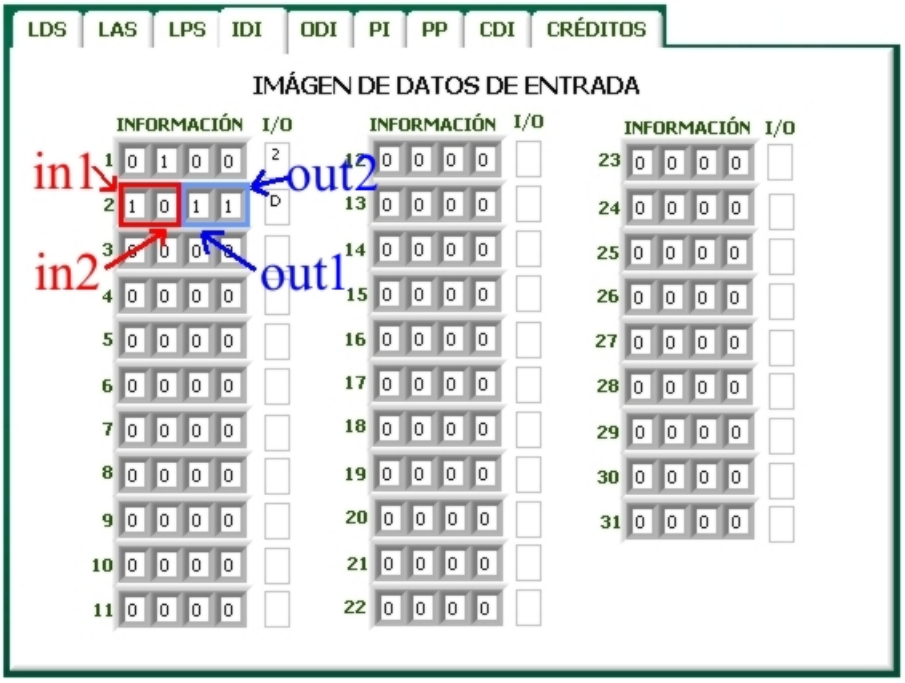


Figura 6.5: **Imágen de Datos de Entrada (IDI) de dos esclavos ubicados en la red *AS-i*.**

Capítulo 7

CONCLUSIONES Y RECOMENDACIONES

7.1. Conclusiones

Se desarrolló un programa implementado en LabVIEW® denominado "Maestro AS-i", que incluye la implementación de las 6 etapas de la máquina de estados del maestro, la construcción de las peticiones (datagramas) según la norma IEC 62026-2, así como el manejo de la información proveniente desde los esclavos. Todo lo anterior se efectuó desde un punto de vista académico, ya que permite el estudio e interpretación de la comunicación en el protocolo *AS-Interface*.

El perfil de maestro implementado corresponde a un *Standar Master*, el cual maneja entradas y salidas binarias. El perfil M.3 de entradas y salidas digitales y analógicas, no pudo ser implementado, debido a la no disponibilidad de esclavos que manejen variables analógicas.

Se determinó una frecuencia de muestreo (460,800 Hz) teniendo en cuenta el criterio de Nyquist, el almacenamiento en memoria del equipo utilizado, la velocidad de transmisión y procesamiento de datos del microcontrolador y la velocidad de transmisión de la tarjeta PCI-Serial, es decir, la velocidad de transmisión máxima alcanzada por el pueto serial. Cabe resaltar que debido a la velocidad alcanzada en la transmisión de datos por el puerto serie (460,800 *bps*) fué posible la implementación del maestro en modo contínuo, ya que el sistema tiene la velocidad suficiente para convertir datos y transmitirlos a la red inmediatamente después de que fueron tomados.

Realizadas las pruebas vistas en el capítulo 6 se pudo observar que el maestro se comunica con cada uno de los esclavos de la red, recibiendo adecuadamente sus contestaciones y mostrando la información de las variables que se manejan. Además se constato la realizacion de todas las actividades esperadas, siendo la de mayor importancia el intercambio de datos, en donde se aprecia el cambio en las variables de entrada y de salida.

Para permitir el normal funcionameinto de la comunicación entre el maestro implementado en PC y la red *AS-i* comercial, se desarrolló una interfaz física (*hardware*) que permite al microcontrolador MC9S12E128 efectuar un acoplamiento adecuado con el fin de lograr la transmisión y recepción de datos hacia y desde el bus *AS-i*.

Para lograr la sincronización entre la transmisión y recepción de datos y para poder utilizar la máxima velocidad de transmisión proveniente del PC, el microcontrolador trabajó con una frecuencia de bus de 22,1184 MHz. Además fué necesario el uso de 3 módulos temporizadores, el módulo DAC, y el manejo de registros de 16 bits para poder realizar un proceso de comunicación efectivo con los esclavos de la red.

Por último, se desarrolló la validación del sistema Maestro *AS-Interface* mediante la conexión a una red *AS-i* comercial ubicada en la Escuela de Ingeniería Mecánica, efectuando intercambio de datos con los esclavos propios de la red. Aquí se destacó el reconocimiento de las diferentes direcciones de los esclavos y además se verificó según la norma IEC 62026-2 la amplitud de la señal de corriente generada por el *hardware* implementado que permite la comunicación; la señal voltaje obtenida al inyectar la señal de corriente a la red; el ancho de pulso de salida de la parte de recepción del *hardware*; el tiempo de bit; el correcto funcionamiento de la fuente de alimentación y por supuesto la verificación de las respuestas obtenidas desde los esclavos.

7.2. Recomendaciones

Los autores de este proyecto se permiten proponer las siguientes recomendaciones para futuras mejoras en el sistema desarrollado que también pueden ser aplicadas a otros proyectos de grado:

- Realizar la implementación de los circuitos en montaje superficial, con el fin de reducir el tamaño del sistema y la integración del microcontrolador con las demás etapas del *hardware* para evitar posibles errores en la conexión y buscar un diseño más compacto.

- Estudiar a fondo la implementación del *software* en LabVIEW® para que se desarrolle una mejor interfaz de visualización del maestro, ya que para efecto de la implementación de este proyecto se trató de simplificar el manejo de las listas y los datos pertenecientes al protocolo de comunicación *AS-i*.

- Implementar un perfil maestro híbrido que permita el manejo de variables binarias y analógicas, bajo una misma arquitectura de red.

Bibliografía

[1] O. L. Núñez, "Módulo de Integración de Sensores y Actuadores lógicos basado en el protocolo AS-I de Comunicaciones Industriales." Master's thesis, Universidad Industrial de Santander, Bucaramanga, Colombia, 2006.

[2] *AS-Interface Complete Specification, V3.0, Rev. 0*, I.E.C Std. IEC 62026-2, September 2004. [Internet]. Visite: http://www.iec.org

[3] D. Caro, *Automation Network Selection.* The instrumentation systems and automation society, USA,2004.

[4] (2005) Fieldbus tutorial. a foundation fieldbus technology overview. [Internet]. Visite: http://www.fieldbus.org/

[5] A. Rosado, *Sistemas Industriales Distribuidos:una filosofía de automatización.*, Universidad de Valencia, España, 2005.

[6] *Low-voltage switchgear and controlgear. Controller-device interface systems. Actuator Sensor interface (AS-i)*, CENELEC Std. EN 50295, 1998. [Internet]. Visite: http://www.cenelec.org

[7] *Actuator Sensor Interface. AS-Interface. Complete Specification, V2.1*, I.E.C Std. IEC 62026-2, August 1998. [Internet]. Visite: http://www.iec.org

[8] *Profiles Annex A and B to the Complete AS-Interface Specification, V3.0, Rev. 0*, I.E.C Std. IEC 62026-2, September 2004. [Internet]. Visite: http://www.iec.org

[9] F. Semiconductor, "MC9S12E-Family Device User Guide," Motorola, Tech. Rep., 2003. [Internet]. Visite: www.freescale.com

[10] T. I. Designers, "SN75C3221: 3V to 5.5V Single-Channel RS-232 Compatible Line Driver/Receiver," TEXAS Instruments, Tech. Rep., 2001. [Internet]. Visite: www.ti.com

[11] Metrowerks, "CodeWarrior Development Studio IDE 5.5 User's Guide," Metrowerks Corporation, Tech. Rep., 2003. [Internet]. Visite: www.metrowerks.com

[12] C. Inc., "Wideband RF Transformers. TTWB Series," Coilcraft Inc, Tech. Rep., 2005. [Internet]. Visite: www.coilcraft.com

[13] A. D. Designers, "AD8032: Rail-to-Rail I/O Amplifiers," Analog Devices Inc., Tech. Rep., 1999. [Internet]. Visite: www.analog.com

[14] ——, "AD8612: Ultrafast 4 ns Single Supply Comparators," Analog Devices Inc., Tech. Rep., 2000. [Internet]. Visite: www.analog.com

[15] ——, "ADUM1200: Dual-Channel Digital Isolators," Analog Devices Inc., Tech. Rep., 2006. [Internet]. Visite: www.analog.com

[16] T. I. Designers, "LMV821: LOW-VOLTAGE RAIL-TO-RAIL OUTPUT OPERATIONAL AMPLIFIERS," TEXAS Instruments, Tech. Rep., 2006. [Internet]. Visite: www.ti.com

[17] ——, "OPA2691: Dual Wideband, Current-Feedback OPERATIONAL AMPLIFIER With Disable," TEXAS Instruments, Tech. Rep., 2006. [Internet]. Visite: www.ti.com

[18] L. T. Designers, "LT1054: Capacitor Switched Regulator," Linear Technologies Inc., Tech. Rep., 2005. [Internet]. Visite: www.linear.com

[19] M. Corporation, "Motorola HC12 Assembler," Metrowerks Corporation, Tech. Rep., 2003. [Internet]. Visite: www.metrowerks.com

[20] A. Lázaro, *LabVIEW: Programación Gráfica para el Control de Instrumentación.*, 1st ed. Editorial Paraninfo, 1997.