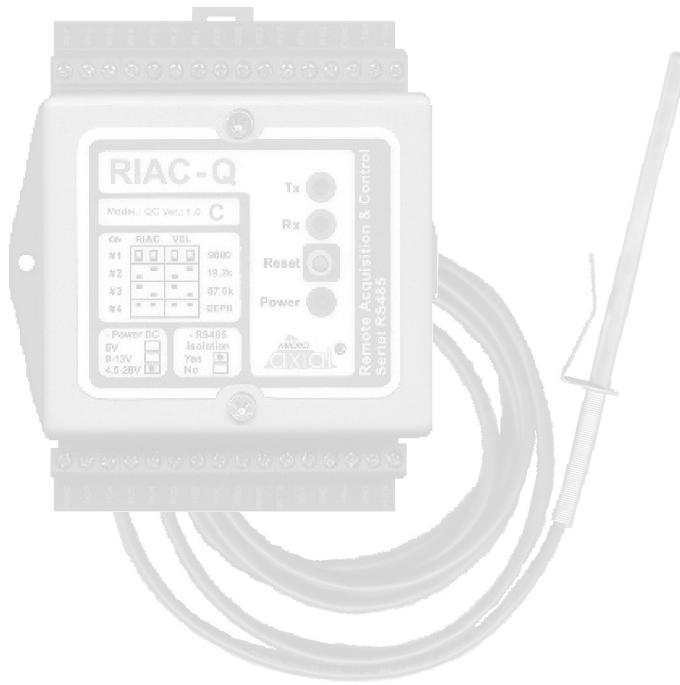




ADQUISICION DE DATOS & CONTROL



RIAC-QF

Módulo
Remoto e Inteligente
para
Adquisición & Control

DESCRIPCIÓN TÉCNICA DEL HARDWARE Y SOFTWARE
DESCRIPCIÓN DE COMANDOS



ADQUISICION DE DATOS & CONTROL

Módulo
Remoto e Inteligente
para
Adquisición & Control

DESCRIPCIÓN TÉCNICA
HARDWARE
SOFTWARE

— RIAC-QF —

Módulo Remoto e inteligente para Adquisición y Control

- DESCRIPCIÓN TÉCNICA

* Conversor Analógico-Digital
(modelos QFA1000 / QFD1000 / QFA1600 / QFD1600)

* Entradas digitales

* Salidas Digitales

* Terminales bidireccionales

* Contador / temporizador

* Comunicación serial RS232 / RS485

- Programación en VBASIC, QBASIC, C, etc. (Ver CD)
- Funciones de librería para el usuario, RiacQX. (Ver CD)



Carlos Calvo 3928, (1230) Capital Federal, Argentina, Tel: +54-11 4931-5254,
microaxial@micoaxial.com.ar www.microaxial.com

ÍNDICE

1. DESCRIPCIÓN TÉCNICA DEL HARDWARE Y SOFTWARE

<i>RIAC, Introducción</i>	1.2
<i>Descripción técnica</i>	1.2
<i>Versión y descripción de terminales</i>	1.4
<i>Especificaciones eléctricas</i>	1.4
<i>Dirección y velocidad</i>	1.7
<i>Instalación</i>	1.10
<i>Alimentación</i>	1.10
<i>Conexión a la línea de comunicaciones</i>	1.11
<i>RIAC-QF. Funciones de los terminales</i>	1.13
<i>Secuencia de Reset</i>	1.18
<i>Mecanismos de seguridad</i>	1.19
<i>Comandos públicos y privados</i>	1.20
<i>Prestaciones definidas por el usuario</i>	1.20
<i>Protección del software del usuario</i>	1.21
<i>Adquisición en Tiempo Real</i>	1.21
<i>Tiempos de adquisición</i>	1.22
<i>RIAC-Q 16 bits</i>	1.23
<i>Cómo programar los módulos RIAC</i>	1.29
<i>Modo Nativo</i>	1.29
<i>Modo QX</i>	1.33
<i>Apéndice A y B</i>	1.36

Contenido en CD

2. DESCRIPCIÓN DE COMANDOS

<i>Comandos RIAC-Q, Introducción</i>	2.1
<i>Ejemplo de comandos</i>	2.1
<i>AXICOM A, formalizando el protocolo</i>	2.2
<i>Comandos versus modelo</i>	2.3
<i>Modo QX, generalidades</i>	2.4
<i>Comandos RIAC</i>	2.5
<i>Comandos virtuales</i>	2.22

microAXIAL se reserva el derecho de modificar parcial ó totalmente las presentes características, sin aviso previo. La información provista en manuales, folletos, medios magnéticos, ópticos ó en cualquier otro tipo de soporte ha sido cuidadosamente revisada a efectos de evitar errores u omisiones; la aceptación de la misma, su interpretación así como el uso que se haga de ella queda bajo responsabilidad del usuario.

microAXIAL ® es marca registrada, se prohíbe el uso de la marca y la de sus productos asociados para todo tipo de publicación, salvo en aquellos casos en que se disponga aprobación escrita emitida por el titular de la marca.

RIAC-QF

SR. USUARIO

Agradecemos la confianza por la elección de nuestros módulos RIAC, nos ponemos a su disposición para obtener de ellos el mejor provecho, microAXIAL ha empeñado en su diseño y confección el máximo de los cuidados y nos resultará útil su parecer para mejorar el producto. Desde 1990 microAXIAL ofrece a nuestros clientes productos de calidad, stock permanente, asesoramiento y cordialidad en la atención. Consulte sobre nuestra amplia línea de productos.

Actualizaciones y correcciones respecto de la ultima revisión. (040326)

GARANTÍA

La placa RIAC cuenta con una garantía de 12 meses corridos a partir de la fecha de compra. La garantía cubre todo defecto en la placa RIAC imputable a nuestro proceso de fabricación. La cobertura quedará sin efecto en caso de instalación ó uso que difieran de las instrucciones contenidas en el presente manual. La garantía queda sin efecto si persona ajena a microAXIAL haya intentado reparar el producto. El cumplimiento de la garantía se gestionará en los comercios donde se halla adquirido el producto. Durante el periodo de vigencia de esta garantía será obligación reparar ó reemplazar, a opción de microAXIAL, la placa ó sus elementos defectuosos. La garantía no se extiende bajo ningún concepto a los equipos, instalaciones y/o aplicaciones a que se destine la placa para su uso. Los responsables de microAXIAL quedan expresamente excluidos de toda responsabilidad por daños ó perjuicios que pueda sufrir el comprador original ó terceros cualesquiera como consecuencia directa ó indirecta del uso ó tenencia de la placa.

RIAC, INTRODUCCIÓN

Los módulos remotos RIAC-QF permiten adquirir datos analógicos y digitales, así como realizar acciones de control. Se conectan mediante una línea a todo tipo de computador que cuente con un puerto serie; operan a velocidades comprendidas entre 1200 y 115200 Baudios. Sobre una misma línea es posible instalar múltiples unidades RIAC. Los módulos de la serie RIAC-QF rev.2.0, disponen de comunicación serie RS485 y RS232, un juego de puentes permite al usuario seleccionar la norma a utilizar. El presente manual describe todos los modelos del cuadro de la fig. 2.

Los módulos RIAC QF se proveen con y sin aislación galvánica (CAG/SAG). Los modelos CAG operan dentro de un amplio rango de alimentación: 4.5 a 28VCC. Los modelos SAG operan en el rango de 9 a 13VCC, se dispone también con alimentación de +5V.

RIAC trabaja con un protocolo de comunicación denominado AXICOM-A su principal característica es la facilidad de operación con gran cantidad de lenguajes (VBASIC, QBASIC, C, C++), ó bien con utilitarios tales como el HYPERTERMIAL ó el TERM95. El protocolo consiste en comandos enviados por el computador y respuestas de las unidades remotas, la información se transmite y recibe en código ASCII. Se disponen además funciones de librerías denominadas QX, éstas y los comandos se describen en otra sección del manual destinado al software.

Con cada módulo el usuario recibe además el programa AXICOM.EXE (Windows), éste facilita la prueba e instalación de las RIACs; cuenta con un EDITOR que permite la emisión de comandos y recepción de respuestas, y guía al usuario en el uso de los comandos. Dispone además aplicaciones como : voltímetro, minioscilloscopio, monitor digital, contador y un demostrativo de adquisición en tiempo real.

Finalmente cada módulo RIAC-QF ofrece al usuario la posibilidad de resguardar sus programas contra copias piratas u otros usos no autorizados.

DESCRIPCIÓN TÉCNICA

La figura 1 corresponde al diagrama en bloques de las unidades RIACs, puede apreciarse el microcontrolador y los ports de entrada y salida. Completan el esquema: la fuente de alimentación, la interface de comunicaciones, los circuitos de seguridad, las minillaves de selección y el Led de energía "LED VIVO".

Los módulos cuentan con los ports P0, P1 de 8 bits y P2 de 4bits. La función de cada uno de ellos depende del modelo, según puede apreciarse en la fig. 2. Salvo la diferencia en los ports, todos los modelos son similares en lo referido a comunicaciones, alimentación y conectores. Las opciones programables: velocidad, dirección, etc, se realizan mediante comandos y/o llaves selectoras. Se describen aquí todos los modelos RIAC-QF estándar. Las opciones especiales se describen en el Apéndice B.

En la figura 1 se destaca la línea divisoria de aislación galvánica, en las unidades con aislación galvánica (CAG) la línea de comunicación se acopla ópticamente, en tanto la energía para los circuitos de línea se logra mediante un convertidor aislado de CC/CC. En las unidades sin aislación (SAG) el acople de comunicaciones y energía se logra mediante simples conductores.

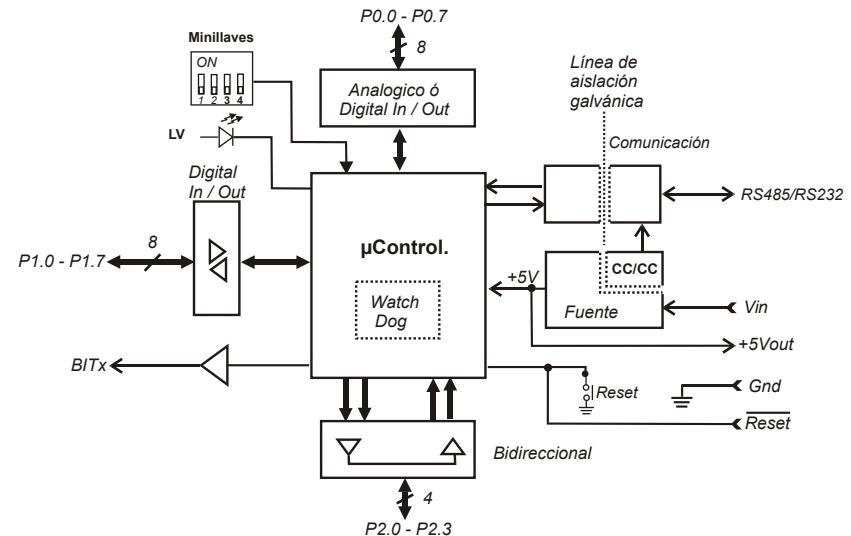


Figura 1. Diagrama en bloques del módulo RIAC

Modelos	Entradas analógicas	Entradas digitales	Salidas digitales	Entrada/salidas digitales (bidirec.)	Salidas analógicas	Contador 16 bits
QFA1000	10 bits Ocho P0.0 - P0.7	Ocho P1.0 - P1.7		Cuatro P2.0 - P2.3		Uno P2.0
QFD1000	10 bits Ocho P0.0 - P0.7		Ocho P1.0 - P1.7	Cuatro P2.0 - P2.3		Uno P2.0
QFB		Ocho P1.0 - P1.7	Ocho P0.0 - P0.7	Cuatro P2.0 - P2.3		Uno P2.0
QFC			Diez y seis P0.0 - P0.7 P1.0 - P1.7	Cuatro P2.0 - P2.3		Uno P2.0
QFE				Diez y seis P0.0 - P0.7 P1.0 - P1.7	Cuatro P2.0 - P2.3	Uno P2.0
QFA1600	16 bits Ocho P0.0 - P0.7	Ocho P1.0 - P1.7		Cuatro P2.0 - P2.3		Uno P2.0
QFD1600	16 bits Ocho P0.0 - P0.7		Ocho P1.0 - P1.7	Cuatro P2.0 - P2.3		Uno P2.0
QFA1600-DA	16 bits Seis P0.0 - P0.5	Ocho P1.0 - P1.7		Cuatro P2.0 - P2.3	Dos P0.6 - P0.7	Uno P2.0
QFD1600-DA	16 bits Seis P0.0 - P0.5		Ocho P1.0 - P1.7	Cuatro P2.0 - P2.3	Dos P0.6 - P0.7	Uno P2.0

Figura 2. Modelos de la serie RIAC-QF

VERSIÓN Y DESCRIPCIÓN DE TERMINALES

La versión se halla grabada internamente y puede solicitarse mediante el comando GV. Ejemplo de versión: RIAC-QFA1000 8I4B8A-S H20 S21 0302, que corresponde al modelo RIAC-QFA1000; 8I = 8 digital Input; 4B = 4 Bidirec; 8A = 8 analog Input; S = Standard; H20 = Hard 2.0; S21 = Soft 2.1; 0302 = número uso interno.

En la figura 3 se describen los terminales, los nombres son genéricos y aparecen en todos los modelos. La correspondencia con el modelo que el usuario dispone puede obtenerse en el cuadro de la figura 2.

∅ RIAC-QF Terminales de bornera ∅		
AGND	Masa analógica	P1.7 Port 1, bit 7
P0.7	Port 0, bit 7	P1.6 Port 1, bit 6
P0.6	Port 0, bit 6	P1.5 Port 1, bit 5
P0.5	Port 0, bit 5	P1.4 Port 1, bit 4
P0.4	Port 0, bit 4	P1.3 Port 1, bit 3
P0.3	Port 0, bit 3	P1.2 Port 1, bit 2
P0.2	Port 0, bit 2	P1.1 Port 1, bit 1
P0.1	Port 0, bit 1	P1.0 Port 1, bit 0
P0.0	Port 0, bit 0	GND Masa
AGND	Masa analógica	P2.3 Port 2, bit 3
BITX	Salida BitX	P2.2 Port 2, bit 2
RST	Entrada Reset	P2.1 Port 2, bit 1
GND	Masa	P2.0 Port 2, bit 0
+5V	VOut, salida regulada	CGND Masa comunicaciones
GND	Masa	-TRX Negativo RS485/Rx RS232
+VIN	Entrada alimentación	+TRX Positivo RS485/Tx RS232

Figura 3. Terminales de bornera y funciones para todos los modelos RIAC - QF.
Ver función específica en cuadro de la figura 2.

ESPECIFICACIONES ELÉCTRICAS

Alimentación y límites, modelo con aislación galvánica (CAG)

- VIN tensión continua entrada: 4,5VCC a 28VCC
- Consumo RIAC (línea RS485 abierta) a 12VCC: 44mA típico.
- VOUT, Volt. continua de salida (ripple $\pm 50mVpp$, 35Khz): 5VCC $\pm 5\%$.
- IOUT máxima corriente de salida: 50mA.
- Sumatoria de corrientes de inyección, valor máximo (1): 40mA
- Rango de temperatura ambiente: -10 °C a +40 °C
- Tensión aislación entre GND y CGND, fuga < 5µA: $\pm 220V$

Alimentación y límites. Sin aislación galvánica (SAG), modelos 9 a 13VCC

- VIN tensión continua entrada: 9VCC a 13VCC
- Consumo RIAC(línea RS485abierta) a 12VCC: 50mA típico
- VOUT, tensión continua de salida: 5VCC $\pm 5\%$.
- IOUT máxima corriente externa de salida: 25mA.
- Sumatoria de corrientes de inyección, valor máxmo (1): 40mA
- Rango de temperatura ambiente: -10 °C a +40 °C

Alimentación y límites sin aislación galvánica, modelos +5VCC

- | | |
|-------------------------------------|-----------------|
| - VIN tensión continua entrada: | 5VCC, +3%, -2% |
| - VOUT, tensión continua de salida: | = VIN |
| - Consumo RIAC(línea RS485abierta): | 50mA típico |
| - Rango de temperatura ambiente: | -10 °C a +40 °C |

Entradas digitales.

- | | |
|--|----------------|
| - Resistores de Pull-Up internos: | 10K |
| - Tensión entrada para nivel bajo (VIL): | 0,0V a 1,1VCC |
| - Tensión entrada para nivel alto (VIH): | 2,4V a 26,0VCC |
| - Corriente de entrada para VIH = 5V: | 400µA |
| - Máxima tensión inversa de entrada (2): | -0,8VCC |
| - Máxima corriente inversa de entrada (2): | -25mA |

Salidas digitales.

- | | |
|---|-------|
| - Salida colector abierto con transistor Darlington | |
| - Máxima corriente de colector para VOL < 1.3V: | 400mA |
| - Tensión de colector, directa máxima: | 48V |
| - Tensión inversa de colector, máxima (2): | -0,8V |
| - Máxima corriente inversa (2): | -25mA |

Bit X

- | | |
|---|-----------|
| - Tensión de salida, nivel alto: | Vin -1,5V |
| - Tensión de salida, nivel bajo: | 1,5V |
| - Corriente de salida, carga a tierra ó a positivo (Vin): | 20mA |

Terminales bidireccionales

- | | |
|------------------------------------|----------------|
| - Resistores de Pull-Up internos: | 10K |
| Entradas con resistores de pull up | |
| - Tensión entrada para nivel bajo: | 0,0V a 1,1VCC |
| - Tensión entrada para nivel alto: | 2,5V a 26,0VCC |
| - Tensión inversa máxima (2): | -0,8VCC |
| - Máxima corriente inversa (2): | -25mA |

Salida colector abierto con resistor de pull-up

- | | |
|--|---------------|
| - Tensión salida para nivel bajo: | 0,0V a 1,5VCC |
| - Tensión salida para nivel alto (mínimo): | 2,5V a 4,5VCC |
| - Máxima corriente colector : | 400mA. |
| - Tensión de colector, directa máxima: | 26,0VCC |

Entradas analógicas, conversor 10 bits.

- | | |
|---|-------------------------------|
| - Conversor: | 10 bits, $\pm 1LSB$ |
| - Canales multiplexados: | 8 |
| - Nivel de entrada: | 0 a 5V unip., $\pm 2,5V$ bip. |
| - Impedancia de entrada: | 20KOhms. |
| - Máxima tensión de entrada: | $\pm 10V$. |
| - Error porcentual sobre F. Escala: | < $\pm 0,5\%$ |
| - Modelos lazo corriente, corriente de entrada: | 4 - 20mA |
| - Impedancia de entrada: | 250Ω |

Entradas analógicas, conversor 16 bits.

- Conversor:	16 bits ± 1 LSB
- Canales desbalanceados/balanceados:	8/4
- Canales balanceados:	4
- Nivel de entrada bipolar:	$\pm 5,12V$ a $\pm 40mV$
- Ganancia:	1, 2, 4, 8, 16, 32, 64
- Impedancia de entrada:	20 K Ω
- Máxima tensión de entrada:	$\pm 10V$
- Error respecto F. Escala:	< $\pm 0,5\%$ (Ver calibración).

Salidas analógicas.

- Conversor:	8 bits ± 1 LSB
- Canales:	2
- Salida lazo de corriente:	0 - 20 mA
- Máxima impedancia de carga:	170 Ω
- Error respecto F. Escala:	< $\pm 1,5\%$.

Comunicaciones.

- Conexión a línea RS232 ó RS485 seleccionable.
- Velocidad [Baudios]. 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, seleccionable mediante minillaves ó comandos. Ajuste en planta 9600 Baudios.
- Comunic. asincrónicabits: 1 arranque, 7 información, 1 paridad par, 1 parada.
- Protección de nivel medio (0,25j) contra descarga sobre línea RS485 y RS232.

Dirección / Reset / Leds

- Dirección RIAC seleccionable mediante minillaves ó comandos. Ajuste en planta igual a 1.
- RESET, pulsador que fuerza a la RIAC a reiniciar su actividad. Durante el reset todas las salidas toman estado alto (reposo). Comunicaciones permanece en reposo.
- LED RX, titila cada vez que la RIAC recibe un comando.
- LED TX, titila cada vez que la RIAC devuelve una respuesta.
- LEDV, intermitente, indica la actividad del microcontrolador

Nota (1). Se denomina corriente de inyección a la que fluye desde cualquier terminal hacia la fuente interna de alimentación. Esta corriente se produce cuando la tensión sobre el terminal en cuestión supera el valor de 5VCC (alimentación interna) y existe un camino de circulación, como por ejemplo las resistencias de pull-up.

Nota (2). Se indican los límites de tensión y corriente inversa sobre los bornes, estos límites consideran los diodos "parásitos" que forman parte de los circuitos integrados. En la práctica la inversión de niveles no ocurre, la recomendación es a efectos de prevenir en caso de aplicaciones inusuales.

IMPORTANTE

Los módulos RIAC disponen internamente de una memoria EEPROM destinada a conservar diversos parámetros. Las operaciones de escritura en la memoria EEPROM tiene un límite máximo. Normalmente todas las operaciones sobre la EEPROM son de lectura, sin embargo hay comandos que escriben la EEPROM, se recomienda utilizarlos manteniendo en vista el límite de 100.000 escrituras. Al presente estos comandos son:

- GN (Gain), ZI (Zero Input), ZB (ZeroBalanced), WE (WriteEEProm)

Estos comandos se utilizan en operaciones de ajustes eventuales. Tenga presente el límite máximo si se los incorpora en ciclos de repeticiones sucesivas.

DIRECCIÓN Y VELOCIDAD

La dirección y la velocidad son dos de los parámetros más importantes y pueden ser seleccionados por el usuario.

La dirección identifica al módulo RIAC-QF como el nombre identifica a una persona. En las unidades RIAC de la serie QF un solo carácter es necesario para definir la dirección, este carácter puede ser numérico ó alfabetico, el rango a utilizar es:

Numérico: 1 hasta 9
Alfabético: A hasta Z

La dirección 0 queda reservada para el direccionamiento público. No se recomienda utilizar caracteres algebraicos, de expresión, puntuación y control.

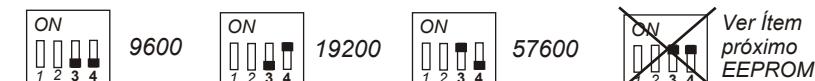
En cuanto a la velocidad de comunicaciones puede ser una de las que sigue: 1200, 2400, 4800, 9600, 19200, 38400, 57600 ó 115200 baudios.

Hay dos modos que permiten al usuario seleccionar la dirección y velocidad: modo EEPROM y modo CABLEADO.

Modo CABLEADO.

En el modo CABLEADO las minillaves permiten seleccionar la dirección y velocidad de operación. Las minillaves 1 y 2 permiten seleccionar una entre cuatro direcciones, en tanto las minillaves 3 y 4 permiten seleccionar tres velocidades.

Selección velocidad



Selección dirección



Modo EEPROM.

Todos los modelos RIAC-QF disponen internamente de una memoria no volátil denominada EEPROM, los datos contenidos en ella permiten definir la dirección y velocidad. En planta se graban los valores que siguen:

Dirección: 5 (rango 1 - 9 y A hasta Z)
Velocidad: 9600 baudios (rango 1200 hasta 115200 baudios)

Para que el módulo asuma los valores disponibles en la EEPROM, las minillaves 3 y 4 deben ubicarse como muestra la figura. Cuando así ocurre las restantes minillaves pierden significado. Si el usuario desea modificar los valores en la EEPROM podrá hacerlo mediante los comandos NR y VL.



Ventajas y desventajas de cada modo.

El modo EEPROM es el que permite un mayor número de alternativas de direcciones y velocidades. La desventaja de este modo se presenta cuando el contenido de la EEPROM se corrompe, esto podría ocurrir si hay severos y persistentes "rebotes" en la alimentación; vale decir que la unidad cuenta con protección contra estos eventos perturbadores (ver más adelante Mecanismos de Seguridad).

En tanto en el modo **CABLEADO** la dirección y la velocidad quedan firmemente definidas por las minillaves, pero contará con un número limitado de alternativas de velocidad y dirección.

A modo de ejemplo. En instalaciones con cuatro RIACs ó menos es conveniente el modo CABLEADO. Con cinco RIACs ó más resultará obligado disponer el modo EEPROM. En una misma instalación pueden coexistir módulos RIAC con modos distintos, sin embargo es conveniente que todas las unidades operen a la misma velocidad.

Ejemplo:

#1 CABLEADO, 9600	RIAC modo CABLEADO: #1, 9600
#2 CABLEADO, 9600	RIAC modo CABLEADO: #2, 9600
#3 CABLEADO, 9600	RIAC modo CABLEADO: #3, 9600
#4 CABLEADO, 9600	RIAC modo CABLEADO: #4, 9600
#5 EEPROM, 9600	RIAC modo EEPROM: #5, 9600
#6 EEPROM, 9600	RIAC modo EEPROM: #6, 9600

Dirección y/o velocidad perdidas. Qué hacer.

Si se está utilizando el modo EEPROM y encuentra limitaciones para modificar la dirección y/o velocidad desde el computador con que se está operando, siga los pasos que se indican a continuación a efectos de reinstalar la dirección y velocidad deseadas.

- Pasar a operar en el modo "CABLEADO". Ejemplo:

Dirección 1, velocidad 9600, minillaves: 1=OFF, 2=OFF, 3=OFF, 4=OFF.

- Emitir los comandos NR (número de RIAC) y VL (velocidad). Los comandos se destinarán a la unidad #1 a 9600 baudios, (puede utilizar AXICOM.EXE)..

#1 NR 7 7<CR>
#1 VL 19K2<CR>

* Nueva dirección en EEPROM
* Nueva velocidad en EEPROM

- Pasar las minillaves al modo EEPROM (minillaves: 1=xx, 2=xx, 3=ON, 4=ON).
- Pulsar Reset.

Terminada la secuencia la dirección será 7 (siete) y la velocidad 19200 baudios.

QUIEN SOY.

La unidad cuenta con el pulsador QUIEN SOY (S1). La acción de este pulsador fuerza a la RIAC-QF a identificarse. Cuando es activada la unidad envía por la línea de comunicaciones la dirección y velocidad con que se halla operando, ejemplo: 7,19K2<CR> indica tratarse de la unidad de dirección 7 que opera a 19200 baudios. Los datos son enviados siempre a 9600 baudios independientemente de la velocidad de operación posterior de la RIAC. El accionamiento de QUIEN SOY tiene la secuencia que sigue: se pulsa Reset y QUIEN SOY simultáneamente, luego se suelta Reset y tras algunos segundos se suelta QUIEN SOY.

INSTALACIÓN

El módulo se provee en una caja plástica ignífuga. Esta puede sujetarse sobre un riel DIN, ó bien haciendo uso de las orejas laterales. En ambos casos se recomienda un montaje vertical a efectos de favorecer la circulación natural del aire.

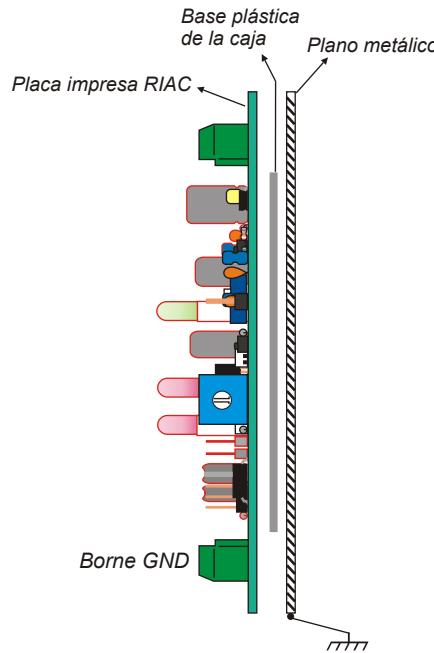


Figura 4. Disposición vertical y plano de tierra.

El consumo es del orden de los 95mA, sin carga sobre VOUT y la línea RS485 abierta. El circuito de alimentación se muestra en la figura 5, puede observarse que cuenta con un filtro y un regulador integrado tipo 7805. La tensión interna de alimentación es +5V, esta se halla disponible en el terminal +5V; se recomienda no extraer más de 50mA.

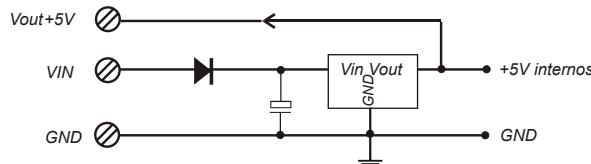


Fig. 5. RIAC, circuito de alimentación sin aislación

Es recomendable disponer de un plano metálico de blindaje de igual ó mayor superficie que la placa misma. El plano se conectará a la tierra física. En la práctica el módulo suele alojarse en una caja metálica que actúa como blindaje.

Los puntos que siguen describen la conexión a la energía y comunicaciones.

ALIMENTACIÓN

Existen dos modelos con y sin aislación galvánica, la alimentación difiere en cada caso y se documentan aquí por separado. Los módulos RIAC pueden utilizar las fuentes AXIPW de microAXIAL (optativo).

“SAG” Sin aislación galvánica.

Modelo 9 a 13VCC. Se proveerá alimentación de corriente continua a través de los terminales de bornera VIN y GND, en el rango de 9V a 13V.

Modelo +5VCC. A solicitud se proveen RIACs para ser alimentadas desde +5VCC. Resulta práctica en aquellas aplicaciones donde el usuario utiliza este valor de tensión. Se muestra el esquema eléctrico; inserte la alimentación por los terminales VIN y GND. **IMPORTANTE.** No invierta la polaridad, producirá daños irreversibles.

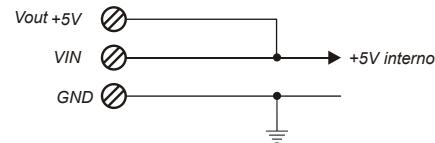


Fig. 6. Circuito de alimentación

“CAG” Con aislación galvánica.

En la figura 7 se muestra el circuito de alimentación. La tensión de entrada se aplica entre VIN y GND en un rango entre 4,5V a 28VCC, el consumo está en el orden de 40/50mA a 12VCC. Se puede tomar corriente del terminal Vout +5V, se recomienda no extraer más de 50mA; el ripple de la salida es del orden de 50mV pico a pico. En la figura puede observarse el uso de un convertidor CC/CC, utilizado para alimentar los circuitos de línea manteniendo la aislación galvánica.

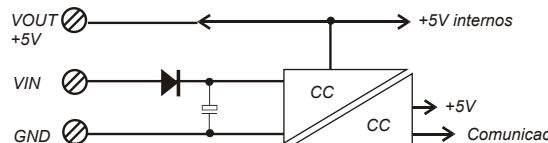


Fig. 7. RIAC, alimentación con aislación

CONEXIÓN A LA LÍNEA DE COMUNICACIONES

Los módulos RIAC-QF (revisión 2.0) pueden operar mediante las normas RS232 ó RS485; el usuario puede seleccionar la norma mediante el puente JP2.

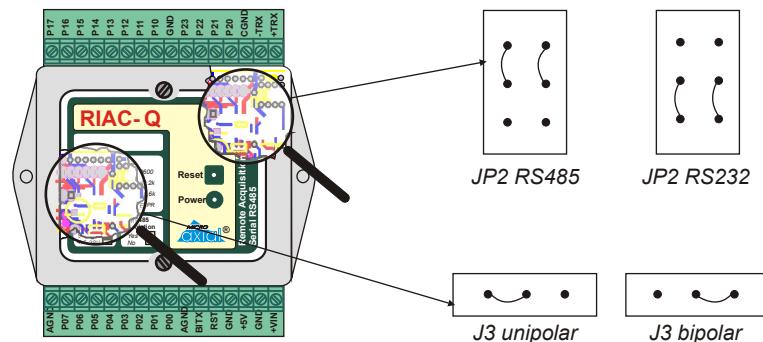


Fig. 8. Puentes de selección

Sugerencias para el uso de RS485 ó RS232. Se utilizará necesariamente RS485 cuando se disponen de dos ó más RIACs sobre el mismo puerto de comunicaciones. En el caso de utilizar una sola RIAC y para líneas no mayor a 40 metros es posible adoptar RS232, si el tendido es mayor a 40 metros convendrá considerar el uso de RS485.

Nota. Si ha de utilizarse RS485 y el computador cuenta únicamente con una salida RS232, microAXIAL ofrece una amplia serie de conversores RS232-RS485 que se instalan directamente sobre el port RS232.

Conexión RS232.

La figura 9 ilustra la conexión, ésta utiliza tres hilos. Bajo RS232 es posible la conexión de una sola RIAC. El cuadro muestra los terminales para los casos de conector DB25 y DB9.

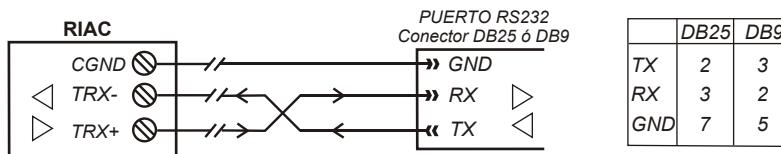


Fig. 9. Conexión RS232

Conexión RS485.

La figura 10 muestra este tipo de conexión, la línea se conecta sobre los terminales TRx+ y TRx-. Se sugiere la conexión a tierra del terminal CGND, la conexión a tierra es a efectos de proteger a la RIAC-QF contra descargas transitorias (0,25 joules máximo) y no es indispensable para la comunicación.

Un juego de Resistores de Terminación (RT) pueden resultar necesarios cuando se opera sobre líneas extensas, mayores a 500mts, y velocidades que superan los 38000 baudios, el valor de RT será del orden de 270 a 1200 ohmios en cada extremo.

Los módulos RIACs disponen internamente de un resistor de pull-up y otro de pull-down sobre los terminales de la línea de comunicaciones, estos imponen una tensión de marca cuando no hay transmisión en curso.

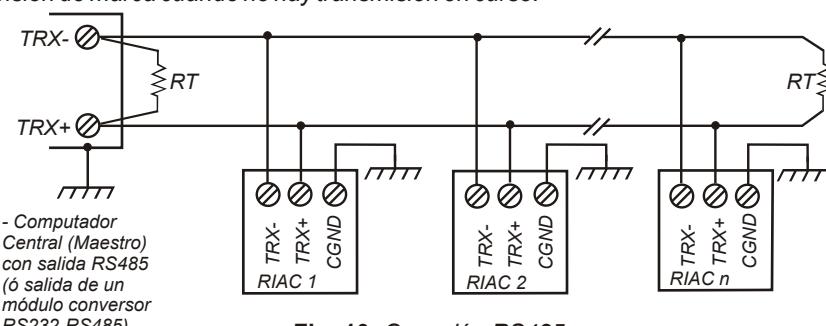


Fig. 10. Conexión RS485

RIAC-QF. FUNCIONES DE LOS TERMINALES

En los puntos próximos se describe la configuración y características eléctricas de los terminales de entrada y salida. Estos son monitoreados ó controlados vía los diversos puertos que dispone el microcontrolador, combinados con los inversores ULN2804 (buffer). En cada ítem se indica uno ó más comandos que permiten operar el terminal en cuestión, los comandos sirven como ejemplo y pueden contarse según los casos de otras opciones; se recomienda revisar el listado de comandos disponibles.

Entradas digitales.

La figura 11 muestra el circuito correspondiente a uno de los terminales de entrada, puede observarse la resistencia conectada al positivo (pull-up). El inversor es uno de los ocho pertenecientes al circuito integrado ULN2804, el diodo a tierra es parte interna del inversor. El estado de las entradas puede obtenerse mediante el comando RI (Read Input) ó BI (Bit Input).

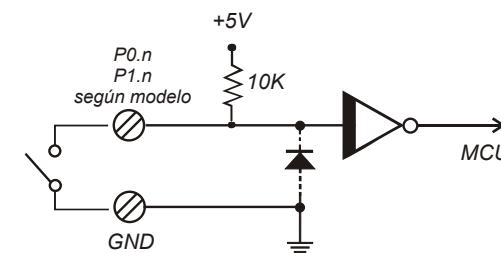


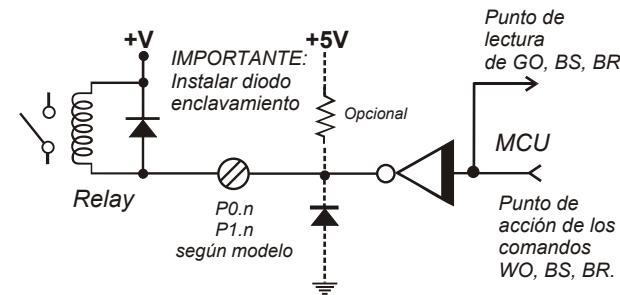
Figura 11. Entrada digital
Vin bajo: 0V a 1.1V
Vin alto: 2V a 26V
R.Pull-Up: 10K

Salidas digitales

La figura 12 muestra el circuito correspondiente a uno de los terminales de salida. El inversor pertenece al integrado ULN2804 y ofrece salida Darlington a colector abierto, el diodo a tierra es parte interna del inversor. Puede solicitarse con resistores de terminación.

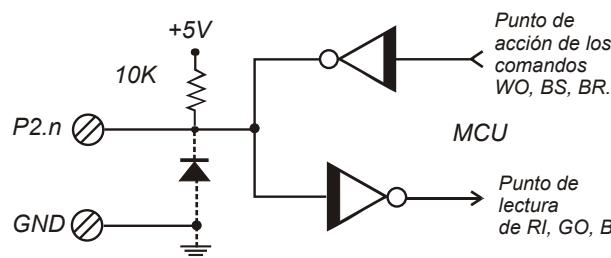
La carga usualmente se conecta entre el terminal de salida y el positivo de una fuente de alimentación. **MUY IMPORTANTE:** para protección del inversor y cuando la carga es inductiva (solenoides, relevadores) será necesario conectar un diodo de clivaje en paralelo a la carga; instale el diodo lo más próximo al inductor, sobre los terminales de ser posible.

Las salidas pueden llevarse a un nivel alto ó bajo mediante los comandos WO (Write Output), BS (bit set) y BR (bit reset). En tanto el comando GO (Get Output) permite conocer el estado de las salidas, este comando hace lectura del port del microcontrolador que gobierna los inversores.



Terminales bidireccionales

La figura 13 muestra el circuito correspondiente a uno de los terminales bidireccionales. Los inversores son ULN2803, un resistor de 10K al positivo completa el esquema, el diodo a tierra es parte interna del inversor. Cada terminal puede emplearse individualmente como entrada ó salida, el usuario seleccionará en cualquier momento la alternativa deseada. Para operar como entrante debe enviarse un nivel alto hacia el Darlington de salida. Los terminales de entrada serán leídos con los comandos RI y BI. En tanto WO, BS y BR gobernarán las salidas.



Entradas analógicas, resolución 10 bits.

El presente ítem es válido para aquellos modelos que cuentan con entradas analógicas. Estas unidades disponen de un conversor AD de 10 bits con un tiempo de conversión de 11μs, completa el esquema un circuito de muestreo, retención y una llave de selección de los canales de entrada. Todos los canales comparten una tierra común. El comando "AI" (Analogic Input) retorna el valor digitalizado de cualquiera de los canales, "VI" retorna el valor en Voltios. En tanto el comando "AA" devuelve el valor digitalizado de todos los canales. Se recomienda el uso del comando VI, sobre todo cuando se desea mantener la compatibilidad en el soft de usuario entre unidades de distinta resolución (10 y 16 bits).

Las entradas analógicas están preparadas para distintas alternativas; a saber: Unipolar, bipolar y lazo de corriente. Se describe cada alternativa por separado.

Nota. Si dispone un modelo de 16bit ver el apartado RIAC-Q16.

Entrada analógica unipolar (figura 8, J3 unipolar). La figura 14 muestra el circuito de entrada de uno de los terminales analógicos. Consiste en un divisor resistivo que ingresa al conversor AD. En la ecuación se obtiene el valor en voltios para un valor digitalizado igual a 873.

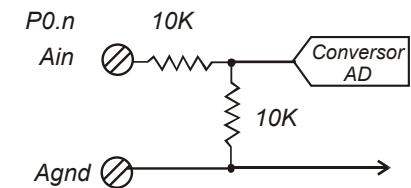
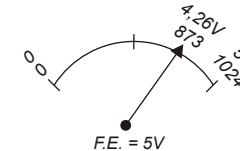


Figura 14. Entrada analógica unipolar
- Nivel de entrada: 0V a +5V
- Impedancia de entrada: 20K



$$Volt = \frac{F.Escala \times Vdigital}{1024} = \frac{5 \times 873}{1024} = 4,263V$$

- **Entrada analógica bipolar** (figura 8, J3 bipolar). La figura 15 muestra el circuito de entrada de uno de los canales. Contiene un divisor resistivo conectado a la tensión de referencia interna. En la ecuación se obtiene el valor en voltios para un valor digital de 713.

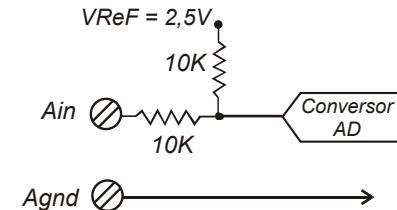
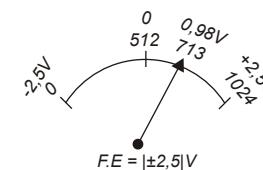


Figura 15. Entrada analógica bipolar
- Nivel de entrada: -2,5V a +2,5V
- Impedancia de entrada: 20K
- Desplazamiento: 2,5V



$$Volt = \frac{2xF.Escala \times (Vdigital-512)}{1024}$$

$$Volt = \frac{2 \times 2,5 \times (713-512)}{1024} = 0,98V$$

IMPORTANTE. El generador que provee la señal de entrada debe tener baja impedancia (menor a 200Ω). En particular el cero se logra con un "cable a tierra", en tanto el circuito abierto dará lugar a una tensión positiva igual al fondo de escala.

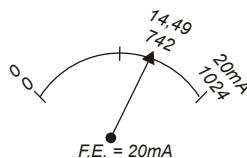
- **Entrada lazo de corriente** (figura 8, J3 unipolar). La figura 16 muestra el circuito de entrada preparado para un lazo de corriente analógico. La unidad RIAC se halla preparada para un rango de 0 a 20mA. El programa que gobierna la RIAC deberá

tener en cuenta el desplazamiento cuando se opera con señales de 4 a 20mA. En la ecuación se obtiene el valor en mA para un valor digital de 742.

También es posible utilizar VI para determinar el valor de corriente, a los efectos divida el valor en voltios por la resistencia de terminación (250Ω).

Las resistencias de terminación de 250Ω se hallarán instaladas en el impreso cuando fue solicitado por el usuario, caso contrario pueden montarse exteriormente.

Figura 16. Entrada lazo de corriente
 - Nivel de entrada: 0 a 20mA
 - Impedancia de entrada: 250Ω



$$MA = \frac{F.Escala \times Vdigital}{1024} = \frac{20}{1024} \times 742 = 14,49mA$$

AGND recomendación. Se recomienda utilizar AGND como terminal común para las señales analógicas, éste concurre a la masa general, pero no es circulado por corrientes que pueden incorporar señales espurias.

Entradas analógicas utilizadas como digitales (universales). Las entradas son universales, ya que además del uso analógico arriba descrito, es posible utilizarlas parcial ó totalmente como digitales. El programa de usuario hará reconocimiento de niveles. Por ejemplo todas las lecturas por arriba de 2,0V corresponden a niveles altos, en tanto las lecturas por debajo de 0,8V se toman como niveles bajos (valores utilizados por la familia TTL).

Bit X

Es una salida que puede tomar valores estáticos (alto ó bajo) ó bien una oscilación entre ambos valores, oscilación de periodo seleccionable por el usuario. El comando **BX** permite seleccionar las alternativas indicadas. El nivel de salida alto será de: $+Vin - 1,5V$, y el nivel bajo $1,0V$. Por ejemplo para $Vin = 12V$, el nivel alto valdrá $10,5V$. Corriente de salida hasta $20mA$ hacia tierra ó el positivo. Es posible utilizar en la salida un relevador de $12V / 20mA$ contra tierra ó positivo. Recordar conectar diodo de enclavamiento.

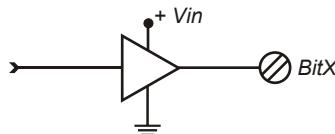


Fig. 17. Salida Bit X
 V Alto: $Vin - 1,5V$
 V Bajo: $1V$

Contador 4

El contador 4 se apoya en el port 2 y es un contador ascendente de 16 bit, la cuenta es continua y circular entre 0000 y 65535. Dispone de dos banderas de alerta que son: desborde y puesta a cero. La bandera de desborde se activa cuando la cuenta pasa de 65535 a 0 y se repone automáticamente tras la lectura del contador. La bandera de puesta a cero se activa cuando el contador es llevado al valor de cero ya sea por acción externa (pulsador) ó por acción de un comando, la bandera se repone tras la lectura del contador.

La máxima frecuencia de conteo es de 1500000 pulsos/seg. La unidad dispone de Arranque, Parada y Puesta a cero ya sea por acción externa (pulsadores) ó por comandos (WO, BS, BR). Las funciones y su activación se distribuyen en el Puerto 2 según se detalla abajo y se grafica en la figura 18:

P2.0	Entrada de pulsos a contar (Clock).	Flanco ascendente.
P2.1	Entrada de arranque (Run).	Flanco descendente.
P2.2	Entrada de puesta a cero (Zero).	Flanco descendente.
P2.3	Entrada de parada (Halt).	Flanco descendente.

Los comandos específicos son:

- OC 4 **OpenCounter 4.** Abre el contador 4, inicia el contador en cero y asigna las funciones contador a los bornes P2.n. Siguen siendo válidos los comandos WO, BS, BR y BI sobre P2.
- CC 4 **CloseCounter 4.** Cierra las funciones de contador.
- RC 4 **ReadCounter 4.** Lectura de la cuenta. Comando válido tras ejecutar OC 4.

El comando RC devuelve una cadena de caracteres con las siguientes alternativas:

- a) inicia con el primer dígito de la cuenta; el anteúltimo es el estado de operación que puede ser: "detenido" (Halt) ó cuenta en progreso (Run). El último es <CR>. Ejemplos:

324 H<CR>	Valor de la cuenta: 324. Contador detenido
18973 R<CR>	Valor de la cuenta: 18973. Contador en progreso
0 R<CR>	Valor de la cuenta: 0. Contador en progreso

- b) inicia con el estado de la cuenta, "+" indica desborde, " " (espaciador) indica puesta a cero. Ejemplo:

+ 832 R<CR>	Tras el desborde 832 cuentas,
" 1981 R"<CR>	Tras el cero 1981 cuentas

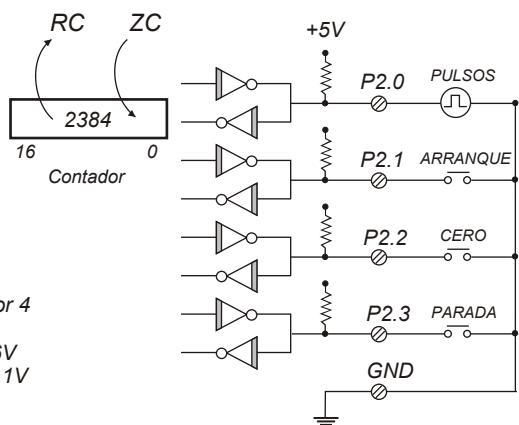


Fig. 18. Contador 4
R Pull-Up: 10K
Vin alto: 2V a 26V
Vin bajo: 0V a 1.1V

SECUENCIA DE RESET

Se indica a continuación la secuencia que realiza el módulo cuando entra en estado de RESET.

RESET activo.

El estado activo del RESET se produce por conexión de la energía, cuando es pulsado el botón de RESET, se dispara el "watch dog", ó bien se activa la entrada vía el terminal de RESET.

- Toda tarea en curso es abortada, el microcontrolador no aceptará comando ni realizará funciones.
- Las salidas a colector abierto permanecen sin conducir.
- En los terminales bidireccionales, las salidas quedan en estado alto.
- El transmisor queda en estado de reposo, el receptor bloqueado.
- Los LEDs TX y RX apagados. El LEDV apagado.

RESET pasivo.

Tras liberar el reset el microcontrolador inicializa el sistema con los resultados que se describen a continuación:

- En los terminales bidireccionales las salidas quedan en estado alto.
- En los terminales de salida los transistores a colector abierto permanecen sin conducir.
- Los LEDs TX y RX apagados.

- El LEDV titila más rápidamente indicando la entrada en actividad.
- El microcontrolador queda con capacidad de recibir y ejecutar comandos.

Cuando se ejecutan los comandos RESET (RS) ó Watch Test (WT), la secuencia arriba mencionada se cumple bajo control del software del usuario.

MECANISMOS DE SEGURIDAD

A efectos de garantizar una operación segura se disponen de diversos mecanismos, tanto en el nivel del hardware como en el de software, estos son:

Watch dog. Si se produjera una pérdida de control, un circuito interno en el microcontrolador produce una secuencia automática de reset. El circuito se denomina "watch dog". Con el propósito de prueba, el usuario puede simular la pérdida de control mediante el comando WATCH TEST (WT), este produce una parada del microcontrolador con el consecuente arrastre al reset.

Caída y rebotes. Si la tensión interna de alimentación (5Volt) cae por debajo de 4.0V la unidad produce un reset del microcontrolador. El reset será librado 50mS después de que la alimentación es repuesta. Ésto evita acciones descontroladas ante un suministro irregular de energía.

Protección por software. Cuando la RIAC-Q recibe un comando, realiza diversas verificaciones antes de aceptarlo, si el comando es correcto realiza la acción y envía respuesta; por el contrario, si encuentra elementos inválidos no realiza acción alguna y no genera respuesta. Un comando inválido puede producirse por error en la emisión de un comando ó por ruido en la línea de comunicaciones.

Cuando se detectan elementos inválidos la RIAC-Q retiene el motivo que lo produjo bajo un código de error. La lista que sigue muestra las verificaciones y los códigos generados en caso de error. El código de error se puede obtener mediante el comando STATUS.

- 0: no error
- 1: código recibido no válido
- 2: el comando emitido no puede ser público
- 3: formato no válido (no ubica dirección, ó código con exceso caracteres)
- 4: error de paridad
- 5: exceso de caracteres en el comando
- 6: número de campos no válido
- 7: campo o con exceso de dígitos. Falta la dirección de la RIAC
- 8: parámetro numérico incorrecto
- 9: velocidad seleccionada no disponible
- 10: pérdida de caracteres (overrun)
- 11: bit de parada no válido
- 12: no corresponde uso de 9º bit
- 13: exceso de dígitos y/o alguno de ellos no es válido.

COMANDOS PÚBLICOS Y PRIVADOS

Cuando se utiliza varias RIACs conectadas a una misma línea, algunos de los comandos pueden destinarse a todos los módulos remotos. Estos comandos se denominan públicos (broadcasting) e inician con la dirección cero. Ejemplo:

#0 WO 2 43 /* Escribe en los ports 2 de todas las RIACs el dato 43.
#0 RS /* Todas las unidades son reseteadas.

Los comandos privados contienen una dirección distinta de cero. Todos los comandos disponibles son privados, algunos pueden ser destinados a todas las unidades y por lo tanto pueden utilizarse como públicos. En el capítulo destinado a los COMANDOS RIAC se indica la posibilidad de cada uno.

PRESTACIONES DEFINIDAS POR EL USUARIO

Los módulos RIAC disponen de prestaciones que define el usuario mediante comandos ó con auxilio del software AXICOM. Los parámetros escogidos quedan en la memoria EEPROM del sistema, por lo tanto se retienen aún con falta de energía.

Bautismo. Cada unidad dispone de una dirección, esta le pertenece a la RIAC y el procedimiento de asignación se denomina bautismo, las direcciones pueden definirse en el rango de 1 a 9 y de A hasta Z. Se recomienda no hacer uso de las direcciones w, x, y, z, éstas serán utilizadas por microAXIAL para futuras prestaciones. La dirección establecida en EEPROM actuará cuando las minillaves 3 y 4 se hallen en ON ON. El bautismo se realiza mediante el comando NR. Ejemplo:

"#1 NR 9 9" /* Nueva dirección: 9, ver comando NR.

Velocidad. Es posible cambiar la velocidad de operación de todas las RIACs conectadas a una línea ó bien de una sola RIAC en particular. Los módulos pueden operar entre 1200 Baudios y 115.2KBaudios, de planta se proveen programado para 9600 Baudios. La velocidad se puede establecer mediante el comando VL. Ejemplo:

"#1 VL 9600" /* Ver comando VL

Nombre de fantasía. Cada módulo puede recibir un nombre que recuerde la función que realiza, por ejemplo CALDERA. El usuario puede crearlo y modificarlo haciendo uso del comando DEFINE FUNCTION, DF. Ejemplo:

"#1 DF CALDERA"

PROTECCIÓN DEL SOFTWARE DEL USUARIO

Los módulos RIAC-Q contienen internamente un sistema para proteger el software de usuario contra usos no autorizados ó copias piratas. La protección se obtiene mediante la combinación de los comandos **DEFINE KEY**, **COMPARE KEY** y **LOCK**, esta prestación permite sustituir los protectores que usualmente se instalan en el port de la impresora. Es importante destacar que la protección no invalida el uso de la RIAC, impedirá que progrese el programa que gobierna el módulo.

DEFINE KEY (DK) permite asignar a cada RIAC-Q una clave de protección, ésta deberá contener ocho (8) caracteres ASCII como máximo, la RIAC eliminará aquellos en exceso. Ninguno de ellos deberá ser; #, <CR>, <LF>, <ETX>, <STX>, coma, punto y coma, ó punto. Ejemplo,

- Comando: #2 DK MICROAXI<CR>
- Respuesta: 2, MICROAXI<CR>

Una vez definida la clave, y cuando el usuario desee establecer la protección, se emitirá el comando **LOCK (LK)**. El comando **LOCK** cierra toda la posibilidad de nuevas definiciones de la clave de usuario, evitando modificaciones involuntarias ó maliciosas. Cuando **LOCK** es emitido la RIAC no volverá a tomar una nueva clave, el comando **LOCK** es irreversible.

- Comando: #2 LK<CR> Pone el candado definitivo
- Respuesta: #2,1<CR> Dato 1, candado.puesto

El comando **COMPARE KEY (CK)** realiza la comparación entre la clave asignada y la clave que el comando **COMPARE** envía. Si hay coincidencia la RIAC retorna un '1', caso contrario retorna un '0'. La respuesta demora 3 segundos, a efectos de frustrar los intentos de violación de la clave por consulta sistemática.

- Comando: #2 CK MICROAXI<CR>
- Respuesta: 2,1<CR>

ADQUISICIÓN EN TIEMPO REAL

Los módulos RIAC-QF pueden enviar información del estado de sus terminales, digitales y analógicos, sin la necesidad continua de un comando que solicite la información. El envío se realiza en forma automática durante períodos regulares, previamente definido mediante el comando RT. Esta modalidad de operación se denomina **Tiempo Real**, una de las ventajas es que las funciones temporales de adquisición quedan a cargo de la RIAC-QF.

Para más detalles consultar el comando **REAL TIME**, en el capítulo **COMANDOS RIAC**. El programa AXICOM presenta una sección especial como guía para el usuario. Se sugiere también consultar la **Nota de Aplicación N° 3 de microAXIAL**.

VELOCIDAD DE OPERACIÓN

La velocidad de operación (V_{op}) de los comandos está determinada por los siguientes factores:

- La velocidad de comunicación adoptada, V_{com} .
- El número total de caracteres en los comandos, N_c .
- La velocidad de procesamiento de la PC y el algoritmo utilizado.
- El tiempo de conmutación de controladores y constantes iniciales de línea.

La expresión que sigue permite el cálculo de la velocidad, en donde F_c es un factor de corrección que depende de la velocidad de procesamiento de la PC y del algoritmo empleado. El valor está comprendido entre 0,70 y 0,95.

$$V_{op} \left[\frac{\text{baudios}}{\text{seg}} \right] = \frac{V_{com} \times F_c}{\frac{V_{com}}{1000} + N_c \times 10}$$

No es significativa la velocidad de procesamiento de la RIAC a velocidades de comunicación inferiores a 19K2 baudios. El cuadro muestra los valores prácticos obtenidos mediante un algoritmo simple operando sobre una Pentium de 1.0GHz con 64 MB RAM y un modulo RIAC-QFA1000.

Comandos y respuestas	N_c	9600	19200	57600	115200	
#1 AI 3<CR> / 1,1023<CR> version 10bits	15	46	85	196	294	Lect/Seg
#1 BI 2 0<CR> / 1,0<CR>						
#1 BS 2 0<CR> / 1,1<CR>	14	48	89	204	308	Lect/Seg
#1 BR 2 0<CR> / 1,1<CR>						
#1 RI 1<CR> / 1,255<CR>	14	49	91	214	325	Lect/Seg
#1 WO 2 15<CR> / 1,15<CR>	17	44	82	195	293	Lect/Seg
#1 ST<CR> / 1,0<CR>	10	61	111	246	357	Lect/Seg
#1 AA<CR> / 1,1023...<CR>	48	18	32	71	103	Lect/Seg
		144	256	568	824	Cnles/Seg

Velocidad de los comandos más significativos.

El programa AXICOM dispone de un procedimiento de prueba de los comandos más significativos, esta rutina permite evaluar la velocidad de los comandos de la RIAC conjuntamente con la PC y la velocidad de comunicaciones deseadas.

Velocidad de comunicaciones en RS232/485. Utilizando RS232 se cubre sin dificultades todo el rango desde 1200 a 115K2 baudios. En RS485 pueden presentarse colisión de datos si la lógica de conmutación de transmisión a recepción ocupa tiempos mayores a 1mS.

RIAC- Q 16 bits

Introducción

La descripción que sigue corresponde a modelos de 16 bits, estos son: RIAC-QFA1600/QFD1600, RIAC-QFA1600 DA/QFD1600 DA y las características correspondientes a la sección analógica son las que siguen.

Características, sección analógica de RIAC-QFA1600 / QFD1600

- Conversor AD de 16 bits de resolución
- 8 canales desbalanceados ó 4 balanceados
- Posibilidad de combinar canales balanceados y desbalanceados.
- Nivel de entrada bipolar.
- Ajuste de cero que incluye los desplazamientos externos.
- Número máximo de muestras: 50 muestras/seg. modulo estándar

Características, sección analógica de RIAC-QFA1600 DA / QFD1600 DA

- Conversor AD 16 bits de resolución
- 6 canales desbalanceados ó 3 balanceados ó combinación
- Nivel de entrada bipolar.
- Ajuste de cero que incluye los desplazamientos externos.
- Número máximo de muestras: 10 muestras/seg.
- Dos canales de salida DA de 8 bits de resolución
- Salida lazo de corriente 0-20mA
- Salida de tensión 0 a 3.6V, con resistor de terminación.
- Las características de la sección digital es la descripta en puntos anteriores de este manual.

Descripción

Los módulos RIAC-QFA1600 y QFD1600 disponen de conversor analógico digital de 16 bits de resolución, cuentan además con entradas balanceadas y entradas desbalanceadas también llamadas "single-end" ó "tierra común". El usuario puede combinar ambas alternativas, por ejemplo dos canales balanceados y el resto desbalanceados.

Todos los canales admiten señales bipolares, tanto en el modo balanceado como desbalanceado. Los módulos disponen de ganancia programable que brindan al usuario ocho fondos de escala distintos. Cuenta además con ajuste de cero realizado mediante comandos de usuario; esta alternativa tiene en cuenta los desplazamientos internos y externos; el valor de ajuste es guardado en memoria no volátil y es automáticamente descontado en todas las mediciones posteriores.

Los modelos RIAC QFA1600 DA y QFD1600 DA disponen además de dos conversores digital-analógico de 8 bits. La salida sirve para conformar un lazo de corriente de 0-20mA.

Canales desbalanceados

La figura 19 muestra el esquema de uno de los canales desbalanceados, la tensión de entrada se aplica entre el borne del canal de interés -P0.0 a P0.7- y el terminal común AGnd (analogic Ground). El amplificador de ganancia programable (PGA) es compartido por todos los canales, una llave electrónica conecta el canal seleccionado por el usuario con el amplificador; la llave se ha omitido en el dibujo. La lectura se realizará con los comandos VI y AI.

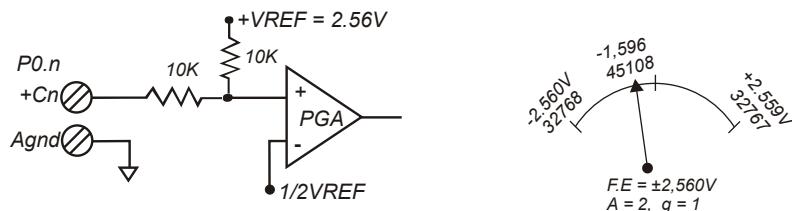


Fig. 19. Entrada canal desbalanceado.

Canales balanceados

RIACQ dispone de un total de 4 canales balanceados similares al de la figura 20. El amplificador de ganancia programable es compartido por todos los canales, una llave electrónica (se ha omitido en el dibujo) permite seleccionar el canal deseado. La lectura se realizará con el comando VB. Esta configuración tiene la virtud de presentar un alto rechazo al ruido en modo común, resulta además indispensable cuando se ha de efectuar mediciones sobre puentes.

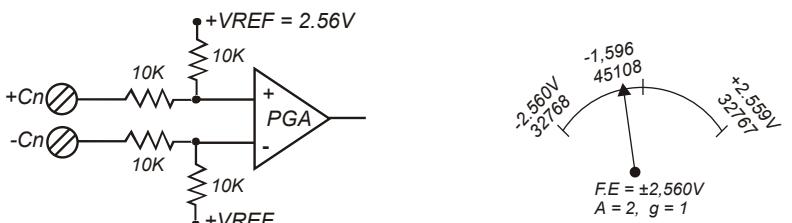


Fig. 20. Entrada canal balanceado.

El cuadro de la figura 21 indica los canales y la asignación que tiene sobre la bornera. Puede apreciarse que hay dos modos de tomar los canales, la duplicidad es para ofrecer alternativas de cableado. Ejemplo: #1 VB 2, corresponderá al canal definido entre P0.2 y P0.6. En tanto "#1 VB 5" hace referencia a un canal definido entre P0.2 y P0.3. Es posible combinar ambas alternativas.

	+Cn	-Cn	Comandos
Altern 1	P0.0	P0.4	VB 0
	P0.1	P0.5	VB 1
	P0.2	P0.6	VB 2
	P0.3	P0.7	VB 3
Altern 2	P0.0	P0.1	VB 4
	P0.2	P0.3	VB 5
	P0.4	P0.5	VB 6
	P0.6	P0.7	VB 7

Figura 21. Distribución de los canales balanceados.

PGA amplificador de ganancia programable.

La unidad cuenta con un amplificador cuya ganancia puede modificar el usuario mediante el comando GN. Se cuenta con un rango de 8 ganancias a saber: 1, 2, 4, 8, 16, 32, 64 y 128. La tabla de la figura 22 muestra el vínculo entre el valor de ganancia y el correspondiente fondo de escala. Se adjunta un ejemplo de comando:

#1 GN 4<CR>
#1,4<CR>

* Selección g = 4 correspondiente a una ganancia de 16.
** OK

Figura 22. La tabla vincula la ganancia con los valores de fondo de escala.

g	A	FE	VI, VB Desbalanceado Balanceado		AI Desbalanceado Valor digital
			39,999 0,000 -40,000	mV	
7	128	±40 mV	39,999 0,000 -40,000	mV	32767 0 32768
6	64	±80 mV	79,998 0,000 -80,000	mV	32767 0 32768
5	32	±160 mV	159,995 0,000 -160,000	mV	32767 0 32768
4	16	±320 mV	319,990 0,000 -320,000	mV	32767 0 32768
3	8	±640 mV	639,981 0,000 -640,000	mV	32767 0 32768
2	4	±1,280 mV	1,280 0,000 -1,280	Volt	32767 0 32768
1	2	±2,560 mV	2,560 0,000 -2,560	Volt	32767 0 32768
0	1	±5,120 mV	5,120 0,000 -2,760	Volt	32767 0 47872

Ajuste de cero del sistema.

Los modelos RIACQ de 16 bits cuentan con ajuste de cero mediante comandos. El ajuste puede corregir los desplazamientos internos y los externos de tensión, para tal propósito se dispone de ZI para el caso de entradas desbalanceada y ZB para las balanceadas.

La tensión de entrada presente al emitir el comando ZI ó ZB, se toma como "valor cero" ó tensión a partir de la cual se refieren todas las lecturas posteriores sobre el canal. El "valor de cero" se considera junto al de ganancia como un juego de valores, éste se conserva en memoria no volátil (EEPROM) como el juego de valores de calibración, y perduran aún tras un corte de energía ó un reset.

Si en medidas posteriores el usuario decide cambiar de ganancia, automáticamente la placa RIAC realizará las correcciones para garantizar la exactitud de la medida.

Ver los comandos ZI, ZB y la Nota de aplicación 4 "Adquisidores, ajuste de cero y fondo de escala".

Comandos analógicos.

El usuario utilizará el esquema desbalanceado ó balanceado, esto depende del tipo de sensor a utilizar. En tanto la RIAC realizará la lectura según el comando recibido. El comando VB retornará el valor en mV ó Voltios de los canales balanceados. En tanto VI retornará el valor de los canales desbalanceados. Ejemplo:

#5 VI 3<CR> /* Canal desbalanceado 3
5, -1.240<CR> /* tensión medida -1.240V

#5 VB 0<CR> /* Canal balanceado 0
5, 2.367<CR> /* tensión medida 2.367V

El comando AI retorna el valor digital (únicamente para desbalanceado).

Las expresiones que siguen obtienen el equivalente en Voltios a partir del valor retornado por el comando AI. El conversor opera en la convención complemento a dos.

$$Volt = | FE | \times \frac{Valor\ Digital}{32768} \quad \text{Excusión positiva } 0 \leq Valor\ dig. \leq 32767$$

$$Volt = -| FE | \times \frac{65536 - V. Digital}{32768} \quad \text{Excusión negativa } 32768 < Valor\ dig. \leq 65535$$

Recomendación. El empleo de los comandos VI y VB facilita la compatibilidad, el mismo programa de usuario servirá tanto para el modelo de 10 como para el de 16 bits.

Conversores DA

Los módulos QFA16-DA y QFD16-DA cuentan con dos conversores digitales-analógicos de 8 bits, denominados IDAC0 e IDAC1. La figura 23 muestra el esquema básico de uno de los canales, puede en él apreciarse que se trata de una salida de corriente proporcional al valor digital, la expresión que sigue vincula los términos.

$$I_{out} [mA] = \text{valor digital} \times 20 / 256$$

Para un correcto funcionamiento la impedancia de carga externa deberá hallarse comprendida entre 0 y 170Ω. El comando AO se emplea para gobernar las salidas analógicas. Si el usuario desea una salida de tensión, se agregará una resistencia de terminación como muestra la figura 24. Un valor de 125Ω arrojará una tensión de fondo de escala de 2.5V. El máximo valor de R que garantiza linealidad de salida es de 170Ω.

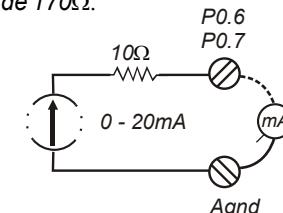


Figura 23. Lazo de corriente 0-20mA.
P0.6 corresponderá al canal 0 y
P0.7 al canal 1

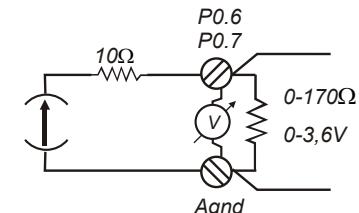


Figura 24. Salida de tensión.
P0.6 corresponderá al canal 0 y
P0.7 al canal 1

COMO PROGRAMAR LOS MÓDULOS RIAC

Los módulos operan merced a la recepción de comandos y el envío de respuestas. Estas acciones se realizan desde una PC mediante dos alternativas denominadas modo Nativo y modo QX.

El modo nativo es el original del los módulos RIAC, en este modo se envían los comandos y se reciben las respuestas totalmente en ASCII, el programa de aplicación deberá procesar la información que intercambia con la RIAC, suele también decirse que opera con los datos "crudos".

En el modo QX la aplicación ve al módulo RIAC a través del control RiacQX.ocx, esto facilita la confección del programa ya que los datos llegan "digeridos" a la aplicación. El control RiacQX ha sido preparado para VBasic, pero puede combinarse con otros lenguajes. El modo QX es 5 a 10% más lento que el modo nativo.

El capítulo "DESCRIPCIÓN DE COMANDOS" contiene en forma detallada los comandos en modo nativo y modo QX, así como propiedades, tablas y misceláneas.

Finalmente, se dispone del programa AXICOM, éste permite familiarizar rápidamente al usuario con la RIAC.

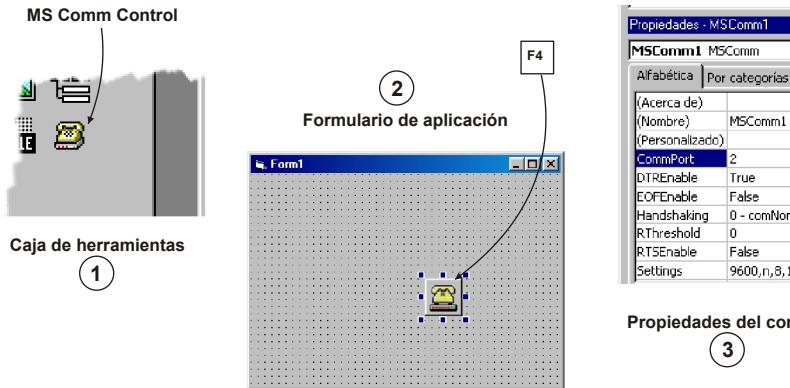
MODO NATIVO

El computador (maestro) envía comandos destinados a los módulos RIAC, estos devuelven respuestas, el intercambio de información es en código ASCII.

En la aplicación se deberán separar los campos y obtener los datos útiles. Además deberán tomarse los recaudos para el caso en que no se reciba respuesta de la RIAC, ó la respuesta no es válida. Los motivos pueden ser:

- La RIAC se halla desconectada de la línea ó sin alimentación.
- El comando fue enviado a un puerto inexistente.
- La dirección indicada es errónea.
- La velocidad ó la paridad son inadecuados.

Modo Nativo, VBASIC. Instale en la caja de herramientas el control de comunicaciones Microsoft Comm Control (el ícono suele ser un teléfono). El control está asociado a mscom32.ocx. Pegue en su formulario de trabajo el control de comunicaciones. Luego diríjase a las propiedades del control (F4) y defina los valores según muestra el ejemplo:



Rebautice el nombre (Name) del control, aquí se ha adoptado "rcom".

`(name) = rcom`

El nombre elegido es consecuente con los ejemplos.

`rcom.comport = n`

`n = 1, 2, 3, 4, puerto de comunicación en donde se halla instalada la RIAC.`

`rcom.setting = "9600, e, 7, 1"`

El valor 9600 es la velocidad de comunicaciones, si utiliza 4800 deberá escribir "4800, e, 7, 1". Los siguientes corresponden a la paridad (even = par), número de bits de información y número de bit de parada; estos valores deben ser lo aquí indicados.

`rcom.dtr = True`

Pone DTR en marca. La placa RIAC no necesita de este valor, es a efectos de utilizar con los conversores de la serie AP485. Si no es su caso ignore esta acción.

Las propiedades del control pueden también definirse desde el programa de usuario. A continuación se muestran ejemplos en VBASIC, los programas se hallan junto a otros ejemplos y aplicaciones en el CD de usuario; en estos la documentación se exhibe ampliada.

Ejemplo rdemo_1.

El programa envía el comando GV (get version) y espera la respuesta.

```
*****
RIAC. RDemo_1, ejemplo de empleo en VBASIC
*****
Private Sub Form_Load()
    ****INICIALIZACION DEL PUERTO DE COMUNICACIONES
    rcom.Settings = "9600,e,7,1"      * 9600Bd, paridad par, 7 datos, 1 bit de parada
    * Se indica aquí el puerto donde se halla instalada la RIAC.
    rcom.CommPort = 1
    *Apertura del puerto 'rcom'
    rcom.PortOpen = True

    ****ENVIODE COMANDO
    *"-#1 GV" es el comando Get Version, que pide a la RIAC que retorne la versión.
    rcom.Output = "#1GV" + Chr$(13)      *.... vacomando

    ****RECEPCIONDE RESPUESTA
    *La respuesta se captura con la sentencia: 'a = rcom.input'.
    hc = Timer                         *Hora cero. Referencia inicial de tiempo
    a$ = ""                            * Limpia la variable.

    * Lazo de espera de la respuesta. Se espera por 100mSeg (0.1Seg)
    Do
        a$ = a$ + rcom.Input           * Lectura del buffer de recepcion
        Loop Until hc + 0.1 < Timer   * Se espera por un total de 100mS (0.1)

        * Si a = "" entonces no se recibio respuesta.
        If a$ = "" Then
            rlabel_1.Caption = a
        End If

        * Cierre del puerto de comunicaciones
        rcom.PortOpen = False

    End Sub
```

IMPORTANTE. En relación a la "recepción de la respuesta", el usuario deberá tener en cuenta que ésta puede no arribar por motivos antes descriptos. El programa toma en cuenta estos aspectos, a los efectos se espera la respuesta dentro de un lazo temporal (DO-LOOP), una vez terminado asume la situación.

Ejemplo rdemo_2.

El programa permite enviar un comando y queda a la espera de una respuesta. Es una versión simplificada del panel principal del AXICOM.

Ejemplo rdemo_3.

El ejemplo se aplica al circuito de la figura 25. Consiste en un control sencillo de temperatura. El sensor conectado al canal 0 ($P0.0$) permite registrar la temperatura; cuando ésta sobrepasa los 68°C se debe desconectar el calefactor. El calefactor se halla gobernado por el contacto del relay ubicado en $P2.1$. El proceso se inicia haciendo click en un botón sobre el formulario RDEMO_3; a partir de ese instante se mide y presenta la temperatura, en tanto se hará titilar el LED.

El corte se realizará cuando la temperatura exceda los 68°C ó bien cuando se pulsa el contacto en $P2.3$. El archivo rdemo_3 contiene el programa con ampliada documentación. **IMPORTANTE**. En este ejemplo se hace uso de ON_COM para detectar la respuesta.

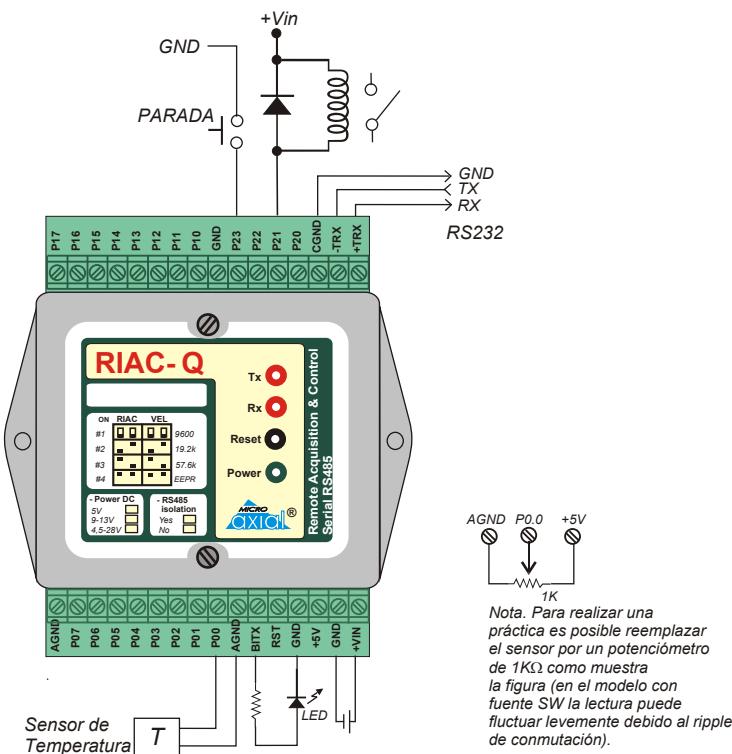


Fig. 25. RDemo_3. Esquema de un controlador simple de temperatura.

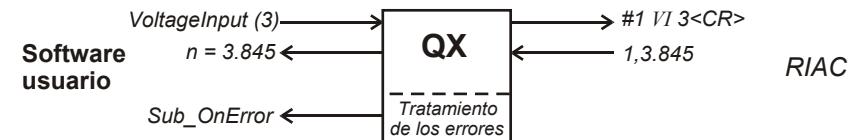
MODO QX

En el modo QX el programa de usuario accede a la RIAC a través del módulo de software RiacQX.ocx. QX permite escribir los comandos en forma más comprensible y además auxilia al programador durante la edición. La respuesta de la RIAC también es procesada y se ofrece directamente sobre una variable. Otra característica importante es la captura y tratamiento de los errores. Se muestra un ejemplo en modo QX.

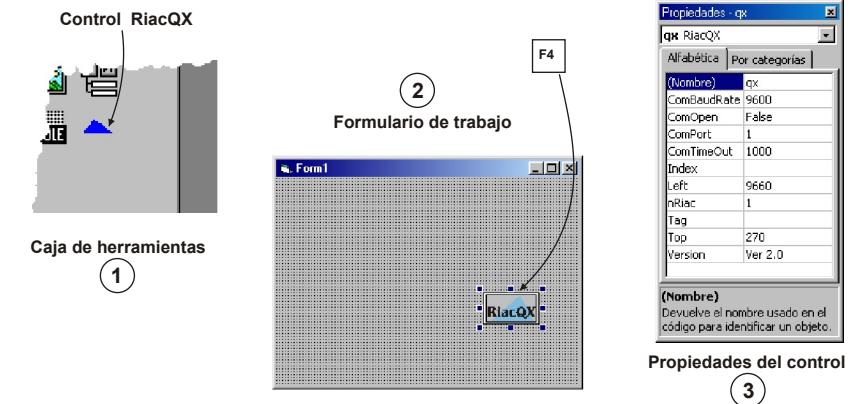
$n = \text{Voltage Input} (3)$

* Modo QX, la variable "n" toma el valor
* en voltios del canal 3.

El dibujo muestra el accionar de QX, puede observarse que la cara que mira hacia la RIAC opera con los comandos crudos. El modo QX es 5 a 10% más lento que el modo nativo.



Modo QX, VBasic. Instale en la caja de herramientas el control RiacQX.ocx. El control se provee en el diskette de usuario y puede ser trasladado a cualquier otra carpeta. Haga click en el "Panel de herramientas", seleccione "componentes" y pegue el control RiacQX. Pegue luego el control en el formulario de trabajo. Diríjase luego a las propiedades (F4) y defina los valores según se muestra el ejemplo.



<i>(name) = qx</i>	<i>El nombre elegido es consecuente con los ejemplos y demostrativos.</i>
<i>qx.comport=n</i>	<i>n = 1, 2, 3, 4, puerto de comunicación en donde se halla instalada la RIAC.</i>

Las propiedades del control se pueden definir también desde el programa de usuario. A continuación se muestran ejemplos en VBASIC, los programas se hallan junto a otros ejemplos y aplicaciones en el diskette/CD de usuario; en estos la documentación se exhibe ampliada.

Ejemplo QXdemo 1.

Lee el estado digital del puerto 2 mediante el comando ReadInput, el resultado es presentado en pantalla. El programa se muestra a continuación, obsérvese la presencia del RiacQ_OnError, subrutina que captura los errores. En la fase de desarrollo la subrutina se abre haciendo doble click en el ícono RiacQX.

```
*****  
* RIACQ.QXDdemo_1, ejemplode empleo en VBASIC  
* Programa valido para todos los modelos RIAC  
*****  
  
/* El presente ejemplo muestra el estado del puerto 2 del modulo RIAC, esto ocurre  
* cada vez que se pulsa el BOTON.  
* Dentro de la subrutina "qx_OnErrorHandler()" se realiza el tratamiento de los errores,  
* en el presente caso consiste simplemente en mostrarlos en la pantalla.  
* qx_OnErrorHandler() aparece en la fase de desarrollo tras pegar RiacQX.OCX y hacer click  
* sobre el icono.  
*****
```

```

Private Sub Boton_Click()
    * Limpia valores previos
    Resultados = ""
    Errores = ""
    * Pone el resultado. El valor es obtenido mediante el comando
    * El comando es enviado haciendo uso del control RiacQX
    Resultados = ax.ReadInput(2)

```

```
End Sub

Private Sub Form_Load()
    ' Programa valido para todos los modelos RIAC.
    ' Se ha adoptado el nombre qx para el control RiacQX
    ' qx.nRIAC sera la direccion de la RIAC a la cual se dirige.
    ' qx.ComPort, puerto donde se halla la/s RIACQ/S
    ' Abre el puerto.
    qx.nRiac = 1
    qx.ComPort = 1
    qx.ComOpen = True

End Sub
```

```
Private Sub qx_OnError()
    ' Punto de entrada en caso de error
    ' Cada vez que se ejecuta un comando en el modo QX, el valor de Status
    ' se incia a cero. En caso de producirse un error automaticamente se ingresa
    ' a la presente rutina donde se captura el valor del error.
    ' Ver en DESCIPCION DE COMANDOS la tabla de errores.
    Errores.Caption = qx.ErrNumber
```

End Sub

Ejemplo QXdemo 2

El programa es un ejemplo pleno en el uso de contador 4, así como un ejemplo completo del control QX.

Ejemplo QXdemo 3

Control sencillo de temperatura, resuelve la misma situación que rdemo_3, pero aplicando el modo QX

Modo QX, tratamiento de errores

Si el control QX no recibe respuesta tras un lapso de tiempo (`comTimeOut`) se producen dos hechos: a) Se retorna un dato virtual a modo de dar cierre formal al comando. b) Se da entrada a la subrutina de error (`ax_Error`).

En la rutina de error puede capturarse el motivo de la falta y proceder acorde.

Dato virtual retornado

- | | |
|---|--|
| <ul style="list-style-type: none">- Dato virtual retornado: -1.- Dato virtual retornado: False- Dato virtual retornado: "Error"- Dato virtual retornado: 0 | <ul style="list-style-type: none">- Variable entera (Integer): -1- Variable lógica (Boolean): False- Variable alfanumérica (String): "Error"- Variable real (Single): 0 |
|---|--|

I listado de errores

Overtime = 1	No se recibió respuesta alguna.
Frame Error = 2	Pulso de parada inválido.
Parity Error = 3	Error de paridad.
Invalid Port = 4	No se detectó el puerto.
Port In Use = 5	El puerto indicado está en uso por otro dispositivo.
Comm No Ready = 6	El puerto no está disponible.
Invalid Parameter = 7	Parámetros de comunicaciones no válidos.
Port Already Open = 8	El puerto ya está operativo.
Over Real Time = 9	No se recibió respuesta operando en RealTime.
No Error = 0	No se detectaron errores.

APÉNDICE A

NORMAS, RS232 y RS485, COMENTARIOS

La norma RS232 fue pensada para vincular un computador y un modem. En tanto la norma RS485 es balanceada y fue pensada especialmente para vincular múltiples dispositivos mediante dos hilos. El módulo RIAC-QF es binorma, si desea utilizarlo con RS485 y dispone de una PC con salida RS232 resultará necesario un conversor, se mencionan algunos modelos con sus características más destacables.

- AP9020/9923 Conversor RS232-RS485 DB9, autoalimentado.
- AP5063/5363 Conversor RS232-RS485/422, DB25 autoalimentado.
- AXI536x Conv. RS232-RS485/422, DB25, alimt. +5V, +12V, +9-28VCC.
- OPTO5061/5361 Conv. RS232-RS485/422, DB25, optoaislado, 4.5 a 28VCC.

Instalado el conversor el usuario puede desentenderse de las normas, todo consiste en enviar y recibir caracteres. **Nota.** Si se utiliza conversores autoalimentados como los modelos AP5000, resulta necesario activar la señal DTR.

NOTAS

APÉNDICE B

Se indican las características técnicas de las prestaciones no estándar, estas deben ser solicitadas por el usuario. Si le resulta útil consulte nuestros distribuidores ó a microAXIAL en forma directa. Nuevas alternativas son comunicadas en el archivo UPDATE.DOC.

- **Entradas ‘pull-down’.** Las entradas quedan referidas a tierra mediante resistores de pull-down de 10K, estos se instalan en planta.
- **Salidas, nivel bajo tras reset.** Las salidas tomarán el estado bajo al actuar el reset, ya sea por acción del watch dog, mediante el comando RESET (RS) ó al pulsar el botón homónimo. El nivel se mantendrá bajo hasta que es modificado por un comando de escritura.
- **Salidas con resitores de terminación.** Las salidas con $R = 10K$ (u otro valor) a +5V interno.

Módulo
Remoto e Inteligente
para
Adquisición & Control

DESCRIPCIÓN DE COMANDOS

MODO NATIVO

MODO QX

COMANDOS RIAC-Q

INTRODUCCIÓN

Todos los módulos RIAC de la serie Q operan con un protocolo de comunicaciones en código ASCII. El protocolo se denomina AXICOM-A y consiste en comandos enviados por el computador y respuestas de las unidades remotas. Comandos y respuestas pueden lograrse mediante programas generados con los lenguajes corrientes, VBASIC, VISUALC, QBASIC, DELPHI, etc. La presente sección contiene todos los comandos en modo nativo y/o modo QX, sus posibilidades y sintaxis. En tanto una tabla de Comandos versus Modelo muestra la disponibilidad de comandos en cada modelo. Se sugiere también consultar el ítem bajo el título “Cómo programar los módulos RIAC”.

Con la entrega de cada módulo el usuario recibe el programa AXICOM, éste facilita la prueba, instalación y permite la emisión de comandos y recepción de respuestas para familiarizar al usuario con las RIACs. Cuenta además con módulos de aplicación y prueba, a saber: voltímetro, contador, miniosciloscopio, monitor digital y adquisidor en tiempo real.

IMPORTANTE. Los comandos, prestaciones y correcciones posteriores a la impresión del presente manual se incorporan el archivo UPDATE.DOC. Sugerimos su lectura.

EJEMPLO DE COMANDOS

Se muestran ejemplos redactados en VBASIC, también en modo nativo como en modo QX.

'* Modo nativo . Comando que consulta a la RIAC 5, por el estado del Port 1.

Com.Output = "#5 RI 1" +CHR(13) '* Comando Read Input

a = Com.Input '* Respuesta

La respuesta puede ser: a = "5,134", la unidad #5 retorna el dato del port 1 cuyo valores en ese momento 134.

'*Modo nativo. Comando destinado a RIAC 7, escribir en el port 2 el valor 4.

Com.Output = "#7 WO 2 4"+CHR(13) '* Comando Write Output

a = Com.Input '* Respuesta

La respuesta será: a = "7,4". La unidad #7 retorna el valor escrito en el port 2.

'*Modo QX. Comando destinado a RIAC 7, escribir en el port 2 el valor 4.

n = WriteOutput (2,4) '* Comando Write Output

'* Respuesta n = 4

En el modo QX, la dirección a cual va destinado el comando se define previamente mediante qx.nRIAC (n), en el presente ejemplo n = 7. Esta dirección será asumida por todos los comandos posteriores. Para enviar un comando a otra RIAC de la red RS485 se deberá redefinir qx.nRIAC(n), donde n será la nueva dirección.

Hay comandos para uso específico cuya ejecución dependen del modelo, por ejemplo el modelo RIAC-QFB no acepta comandos analógicos, ya que no cuenta con entradas analógicas; si la placa recibiera un comando como AI (Analogic Input) simplemente lo ignora. La tabla comandos versus modelo vincula ambas características.

Finalmente, un comando público es aceptado por todo los módulos RIACs en una red RS485; la dirección emitida deberá ser cero (0), el comando público no genera respuesta. El comando privado va dirigido a una de las unidades RIAC a la vez y siempre genera respuestas.

AXICOM-A, FORMALIZANDO EL PROTOCOLO

Se describen detalladamente los comandos en modo nativo, es decir el grupo de caracteres que acepta como válidos los módulos RiacQ, (el modo QX es una interfase que facilita la conectividad con vista al usuario, pero de cara a la RIAC emite los comandos en modo nativo).

La comunicación es asincrónica, con un total de 10 bits por carácter de información, enviados en serie según las características que se detallan: un bit de arranque, siete bits de información, un bit de paridad par, un bit de stop. La comunicación se realizará en código ASCII. En tanto la velocidad de transmisión es seleccionable entre 1200 y 115200 Baudios.

Formato del comando transmitido. El encabezamiento es el carácter '#' seguido de la dirección de la unidad remota, un dígito ASCII entre 0-9 ó A hasta Z. Luego un espacio y sigue el código de operación formado por dos caracteres alfabéticos mayúsculas, ejemplo WO, RI, etc. Siguen 1 ó 2 campos numéricos dependiendo del tipo de comando.

Cada campo debe hallarse precedido de por lo menos un espacio. El uso de espaciadores adicionales no perturban la interpretación del comando, sin embargo hacen más lenta la comunicación. El cierre del comando es el carácter ASCII retorno de carro, <CR> código ASCII 13.

Los caracteres '#' y 'SP' (espaciador) tienen el uso arriba indicado y no deben ser empleados dentro de un parámetro alfanumérico. Se reserva para futuras aplicaciones los siguientes caracteres: +, -, *, \$, =, (,), <, >, punto y coma, coma, punto, null y ffh.

Formato de la respuesta. La respuesta comienza con la dirección de la unidad remota, sin encabezamiento ó espaciadores que le precedan. Tras la dirección sigue uno ó más campos separados por coma. Los campos pueden ser numéricos ó alfanuméricos, ello depende del tipo de respuesta. El último carácter es un retorno de carro (código ASCII, carácter 13).

Comandos versus Modelo

Comandos	Modelo	QA	QB	QC	QD	QE	QA16	QD16	Púb.	Priv.
AA	AllAnalogic	string	✓			✓				•
AI	AnalogIdnput	single	✓			✓		✓		•
AO	AnalogicOutput	integer					✓	✓	•	•
BI	BitInput	boolean	✓	✓	✓	✓	✓	✓		•
BL	BitLed	integer	✓	✓	✓	✓	✓	✓	•	•
BR	BitReset	boolean	✓	✓	✓	✓	✓	✓	•	•
BS	BitSet	boolean	✓	✓	✓	✓	✓	✓	•	•
BX	BitX	integer	✓	✓	✓	✓	✓	✓	•	•
CC	CloseCounter	integer	✓	✓	✓	✓	✓	✓	•	•
CK	Compare Key	integer	✓	✓	✓	✓	✓	✓		•
DF	DefineFunction	string	✓	✓	✓	✓	✓	✓	•	•
DK	DefineKey	string	✓	✓	✓	✓	✓	✓		•
GF	GetFunction	string	✓	✓	✓	✓	✓	✓		•
GN	Gain	integer					✓	✓	•	•
GO	GetOutput	integer	✓	✓	✓	✓	✓	✓		•
GV	GetVersion	string	✓	✓	✓	✓	✓	✓		•
LK	Lock	integer	✓	✓	✓	✓	✓	✓		•
NR	NumberRiac	integer	✓	✓	✓	✓	✓	✓	•	•
OC	OpenCounter	integer	✓	✓	✓	✓	✓	✓	•	•
RC	ReadCounter	long	✓	✓	✓	✓	✓	✓		•
RI	ReadInput	integer	✓	✓	✓	✓	✓	✓		•
RT	RealTime	integer	✓	✓	✓	✓	✓	✓		•
RS	Reset	integer	✓	✓	✓	✓	✓	✓	•	•
SI	SensorInput	single	✓			✓		✓		•
ST	Status	integer	✓	✓	✓	✓	✓	✓		•
VB	VoltBalanced	single					✓	✓		•
VI	VoltInput	single	✓			✓		✓		•
VL	Velocity	single	✓	✓	✓	✓	✓	✓	•	•
WO	WriteOutput	integer	✓	✓	✓	✓	✓	✓	•	•
WT	WatchDog Test	integer	✓	✓	✓	✓	✓	✓	•	•
ZB	ZeroBalanced	integer					✓	✓	•	•
ZC	ZeroCounter	integer	✓	✓	✓	✓	✓	✓	•	•
ZI	ZeroInput	integer					✓	✓	•	•

Modo QX, generalidades.

Propiedades de QX

Hay un grupo de propiedades que el control QX necesita para operar. Estas propiedades toman un valor de inicio, pero el usuario puede redefinirlos según la necesidad de su aplicación, esto se logra mediante los comandos virtuales.

Propiedad.	Valor inicial.	Comando virtual.
ComBaudRate	9600	s = qx.ComBaudRate (velocity)
ComOpen	False	bn = qx.ComOpen (True/False)
ComTimeOut	1000	i = qx.ComTimeOut (Time)
nRiac	1	i = qx.nRiac (#Riac)

El nombre del control "qx" es el nombre por defecto, y puede ser redefinido por el usuario. Velocity rango: 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200. Time rango: 100 a 100000mseg. RIAC rango: 1 - 9, A - Z.

Retorno de datos y listado de errores.

Si el control QX no recibe respuestas tras un lapso de tiempo (ComTimeOut), se retorna un dato virtual a modo de cierre formal al comando.

- Dato esperado Integer. Dato virtual returned: -1.
- Dato esperado Boolean. Dato virtual returned: False
- Dato esperado String. Dato virtual returned: "Error"
- Dato esperado Single. Dato virtual returned: 0

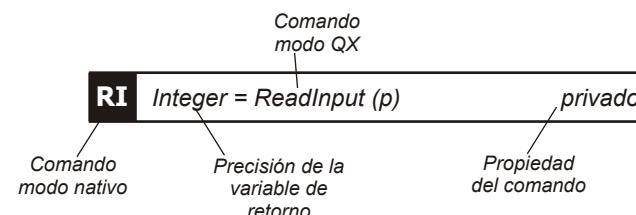
Se produce además el evento qx_OnError, dentro del evento se captura qx.ErrNumber que contiene el código de error.

Overtime = 1	No se recibió respuesta alguna.
Frame Error = 2	Pulso de parada inválido.
Parity Error = 3	Error de paridad.
Invalid Port = 4	No se detectó el puerto.
Port In Use = 5	El puerto indicado está en uso por otro dispositivo.
Comm No Ready = 6	El puerto no está disponible.
Invalid Parameter = 7	Parámetros de comunicaciones no válidos.
Port Already Open = 8	El puerto ya está operativo.
Over Real Time = 9	No se recibió respuesta operando en RealTime.
No Error = 0	No se detectaron errores.

COMANDOS RIAC

En cada encabezamiento los comandos aparecen descriptos en modo nativo y en modo QX. Se indica además si es público y/o privado.

Ejemplo:



Se describen los nombres y funciones de los campos utilizados a lo largo del capítulo.

- a: caractéres alfanuméricos.
- b: número de bits, un dígito entre 0-7
- c: canal analógico, un dígito entre 0 y 7. Contador, un dígito entre 0 y 4.
- d: dato numérico entre 0 y 65535.
- l: entero precisión doble (long)
- p: número de port, un dígito entre 0 y 2
- r: dirección de RIAC, dato alfanumérico entre 0 - 9 ó A - Z
- qx: nombre del control qx
- s: cadena de caracteres (string)
- bn: boolean, variable lógica
- Sgl: variable real presición simple
- <CR>: retorno de carro, código ASCII 13

Se muestra a continuación la expresión general y un ejemplo específico de los comandos según serán utilizados a lo largo del presente capítulo.

Modo nativo. Ejemplo de expresión general

#r RI p<CR> /* Read Input sobre puerto p de RIAC
r, d<CR> /* Devolución del dato d de RIAC.

Modo nativo. Ejemplo de expresión específica

#5 RI 1<CR> /* Lee unidad 5, port 1.
5,32<CR> /* Unidad 5, dato leído 32

Modo RiacQX

i = qx.ReadInput (1) /* Lectura del puerto 1. Sobre i el resultado de la lectura
/* Nota. La unidades previamente especificada
/* por qx.nRiac(n)

AA String = AllAnalogic privado, AD 10 bits

En aquellas versiones que dispone de conversor analógico-digital, permite la lectura de todos los canales analógicos.

```
#r AA<CR>          /* Expresión general de los comandos.
r,C0,C1,...C7<CR>    /* Expresión general de las respuestas.

#7 AA<CR>          /* Lee todos los canales analógicos,
7,23,0,45,125,201,48,48,2<CR>  /* desde C0 hasta C7.

s = qx.AllAnalogic  /* Retorna la cadena de caracteres "s", ejemplo.
                       /* s = "7,23,0,45,125,201,48,42,2"<CR>
```

Ver en comando AI las expresiones para obtener unidades eléctricas.

AI Long = AnalogicInput (c) privado, AD 10 bits

En aquellas versiones que disponen de conversor analógico-digital, permite la lectura digitalizada de cada uno de los canales "c" en forma individual. En situación de desborde el valor provisto es el de fondo de escala: 1023.

```
#r AI c<CR>
r,d<CR>

#7 AI 3<CR>          /* Lectura del canal 3
7,127<CR>            /* Dato digital canal 3 es 127

1 = qx.Analogic Input (3)  /* Lectura del canal 3, en "l", retorna I = 127
```

Las expresiones para obtener unidades eléctricas a partir del valor digitalizado pueden verse en la Descripción Técnica bajo el título: "**Entradas analógicas, resolución 10 bits**".

AI Long = AnalogicInput (c) privado, AD 16 bits

Esta función retorna el valor digitalizado de los canales analógicos desbalanceados con valores positivos y negativos (bipolar). La representación se realiza por la convención complemento a dos. El valor equivalente en voltios depende del fondo de escala seleccionado. En situación de desborde el valor provisto es el máximo del fondo de escala (ver comando GN).

```
#r AI c<CR>
r,d<CR>
#1 AI 3<CR>          /* lectura canal 3
```

```
1, 30124<CR>          /* retorna un número positivo
#1 AI 5<CR>            /* Lectura canal 5,
1, 63291<CR>          /* retorna un número negativo
1 = qx.AnalogicInput (3)  /* Lectura del canal 3, retorna I = 63 291.
```

	RIAC Valor digital	equivale
rango (+)	32767 1	+32767 +1
cero	0	0
rango (-)	65535 32768	-1 -32768

Las expresiones para obtener unidades eléctricas a partir del valor digitalizado pueden verse en la Descripción Técnica bajo el título "**Comandos analógicos**".

AO Integer = AnalogicOutput (c, d) público/privado

Tras la escritura de un valor digital **d** se obtiene un valor analógico proporcional. Sólo disponible en aquellas versiones que cuentan con un conversor digital-analógico. La salida analógica es esencialmente un lazo de corriente de 0 a 20mA. Mediante un resistor de terminación es posible obtener una tensión de salida.

```
#r AO c d<CR>
R,d

#5 AO 1 237<CR>
5,237<CR>          /* Escritura conversor 1, dato 237
                           /* Retorna 237

I = qx.AnalogicOutput (1,237)  /* Escritura conversor 1, dato 237.
                                   /* Retorna I = 237.
```

La expresión para obtener el valor analógico es:

$$I (\text{mA}) = (\text{Valor digital}) \times (\text{FE}/\text{Resolución})$$

$$\text{FE} = 20\text{mA}$$

$$\text{Resolución} = 2^8 = 256$$

$$I (\text{mA}) = 237 \times 20 / 256 = 18.51\text{mA}$$

Con un resistor **RT** de terminación.

$$V (\text{Voltios}) = \text{Valor digital} (\text{FE}/\text{Resolución})$$

$$\text{FE} = 20\text{mA} \times RT$$

$$\text{Resolución} = 2^8 = 256$$

El valor de **RT** deberá estar comprendido entre 0 y 170Ω.

BI Boolean = BitInput (p, b) privado

El comando permite leer en forma individual el estado del bit **b** del puerto **p**.

```
#r BI p b<CR>
r,b<CR>

#5 BI 2 3<CR>          /* Leer el bit 3 del puerto 2
5,0<CR>                  /* El bit 3 vale cero

bn = qx.BitInput (2,3)    /* Leer bit 3 del puerto 2, bn = "False".
```

BS Boolean = BitSet (p, b) público/privado

El comando permite poner en estado alto y en forma individual el bit **b** del puerto **p**. Realizada la acción, procede a devolver el estado del bit.

```
#r BS p b<CR>
r,b<CR>

#5 BS 2 3<CR>          /* Pone un 1 en el bit 3 del puerto 2
5,1<CR>                  /* Unidad 5 retorna el valor de 1

bn = qx.BitSet (2,3)     /* Pone un "1" en el bit 3 del puerto 2, bn = "True"
```

BL Integer = BitLed (d) público/privado

Comando que permite definir dos velocidades de encendido-apagado para el LED VIVO. Obsérvese que tras el reset el LED VIVO (LV) parpadea rápidamente y luego pasa a parpadear en forma pausada, éste es un ejemplo de las dos alternativas de velocidad que ofrece el comando **BL**. El valor **d=1** establece un periodo de 200msg, el valor de **d=0** establece un valor de 1sg.

```
#r BL d<CR>
r,d<CR>

#5 BL 0<CR>           /* El LED titila lentamente
5,0<CR>

I = qx.BitLed (0)       /* LED titila lentamente.
```

BR Boolean = BitReset (p, b) público/privado

El comando permite poner en estado bajo y en forma individual el bit **b** del puerto **p**. Realizada la acción, procede a devolver el estado del bit.

```
#r BR p b<CR>
r,b<CR>

#5 BR 2 3<CR>          /* Pone un 0 en el bit 3 del puerto 2
5,0<CR>                  /* Unidad 5 retorna el valor de 0

bn = qx.BitReset (2,3)   /* Pone un "0"en el bit 3 del puerto 2, bn = "False"
```

BX Integer = BitX (d) público/privado

Este comando actúa específicamente sobre el estado de salida del terminal **BX**. Permite definir a este bit con valores, alto ó bajo como es clásico. También permite oscilaciones permanentes. La acción depende del valor **d**.

```
#r BX d<CR>
r,d<CR>

#R BX 5<CR>            /* Oscilación 400ms'.
r,5

I = qx.BitX (5)         /* Oscilación a 400ms. Retorna I = 5.
```

d	<i>Salida del BIT X</i>
0	toma estado bajo
1	toma estado alto
2	oscilación, periodo 150 msg
3	oscilación, periodo 250 msg
4	oscilación, periodo 350 msg
5	oscilación, periodo 400 msg
6	oscilación, periodo 450 msg
7	oscilación, periodo 500 msg

CC Integer = CloseCounter (c) público/privado

El comando produce el cierre del contador. Tras la emisión del comando **CC** la placa RIAC desconocerá los comandos **RC** y **ZC**. El Port2 dejará de operar como contador y los bornes P2.n son llevados al estado alto.

```
#r CC c<CR>
r,c<CR>
```

#1 CC 4<CR>
1,4

/* Cerrar contador 4
** Retorna valor 4 indicando su cierre.

I = qx.CloseCounter(4) /* Cerrar contador 4. Retorna I = 4.

CK Integer = CompareKey (a) privado

El comando COMPARE KEY realiza la comparación entre la clave previamente asignada y la que el comando envía. Si hay coincidencia la RIAC retorna un '1', caso contrario retorna un '0'. La respuesta demora 3 segundos a efectos de frustrar los intentos de violación de la clave por consulta sistemática en el tiempo.

#r CKa<CR>
0/1<CR>

#2 CK HOL A <CR> /* La clave es HOL A
2,1<CR> /* Confirma.

I = qx.CompareKey (HOLA) /* La clave es HOL A, retorna I = 1.

NOTA. Ver los comandos DEFINE KEY y LOCK.

DF String = DefineFunction (a) público/privado

El comando permite asignar a cada RIAC un nombre de fantasía. El nombre puede utilizarse para memorizar la función que realiza la unidad, ó bien como clave adicional de seguridad. Una vez definido el nombre, puede posteriormente leerse mediante el comando GET FUNCTION. El nombre deberá contener ocho caracteres ASCII como máximo, la RIAC elimina aquellos en exceso. Ninguno de ellos deberá ser: #, <CR>, <LF>, <ETX>, <STX>, coma, punto y coma, ó punto.

#r DF a<CR>
r,a<CR>

#2 DF HORNO_7<CR> /* La función es HORNO
2,HORNO_7<CR> /* Confirma

s = qx.DefineFunction (HORNO) /* La función retorna s = "HORNO"

El nombre es guardado en la memoria no volátil (EEPROM), por tanto será retenido aún bajo corte de energía.

DK String = DefineKey (a) privado

Permite proteger el software de usuario contra usos no autorizados ó copias piratas. La protección se obtiene mediante la combinación de los comandos DEFINE KEY, COMPARE KEY y LOCK, esta prestación permite sustituir los protectores que usualmente se instalan en el port de la impresora. DEFINE KEY posibilita asignar a cada RIAC de una la clave de protección, ésta deberá contener 8 caracteres ASCII como máximo, la RIAC elimina aquellos en exceso. Ninguno de ellos deberá ser: #, <CR>, <LF>, <ETX>, <STX>, coma, punto y coma, ó punto.

#r DKa<CR> /* No produce retorno

#2 DK LLAVE<CR> /* La clave es LLAVE

s = qx.DefineKey (LLAVE) /* La clave es LLAVE.

Ver además los comandos COMPARE KEY y LOCK.

NOTA. Si se ha emitido con anterioridad el comando LOCK, la clave no será asumida, y la RIAC no devolverá respuesta.

GF String = GetFunction privado

Permite obtener el nombre asignado a la RIAC mediante el comando DEFINE FUNCTION.

#r GF<CR>
r,a<CR>

#2 GF<CR>
2,HORNO_7<CR>

s = qx.GetFunction /* La función retorna s = "HORNO".

GN Integer = Gain (g) público/privado, AD 16 bits

Permite seleccionar diversos fondos de escala. Disponible sólo en aquellos modelos que cuenten con amplificadores de ganancia programable, PGA. El fondo de escala guarda correspondencia con el parámetro que acompaña el comando, el cuadro ilustra la relación.

#r GN g<CR>
r,g<CR>

#5 GN 3 /* Parámetro 3 equivale a ganancia 8.
5,3 /* Confirma

I = qx.Gain (3)

/* El valor returnedo es I = 3 que corresponde
* a la ganancia 8.

El valor de ganancia seleccionado sirve para todos los canales, tanto balanceados como desbalanceados. El valor de ganancia se guarda en memoria no volátil, por tanto una pérdida de energía ó una situación de reset no provoca la pérdida de la ganancia previamente elegida; tras el reset, una lectura con los comandos VI, VB ó AI se realizan con el último valor de ganancia seleccionado por el usuario.

Modelos QFA16 / QFD16		
g	A: ganancia	F Escala
0	1	± 5.120V
1	2	± 2.560V
2	4	± 1.280V
3	8	± 640mV
4	16	± 320mV
5	32	± 160mV
6	64	± 80mV
7	128	± 40mV

GO Integer = GetOutput (p) privado

Obtiene el estado del port de salida p, de la RIAC r. La respuesta contendrá la información del estado de los bits del port p.

```
#r GO p<CR>
r,d<CR>
#2 GO 0<CR>          /* Lee unidad 2, port 0.
2,3<CR>                /* Unidad 2, dato leído 3
I = qx.GetOutput (0)    /* El port 0 retorna el valor I = 3.
```

GV String = GetVersion privado

Permite obtener el número de versión de la RIAC. La respuesta contendrá como máximo 20 caracteres ASCII indicadores de la versión.

```
#r GV<CR>
r,a<CR>
#2 GV<CR>          /* Obtenga la versión
2,RIAC-QFA 8I4B8A-5 H20 S20 0403<CR>
s = qx.GetVersion      /* Obtenga la versión,
's' = "RIAC-QFA8I4B8A-5 H20 S200403<CR>"
```

LK Boolean = Lock

privado

El comando LOCK cierra toda posibilidad de nuevas definiciones de la clave de usuario. Tras haber definido la clave se recomienda emitir el comando LOCK, esto evita modificaciones involuntarias ó maliciosas. La respuesta de la RIAC en todos los casos es '1'.

#r LK<CR>
r,1<CR> /* Siempre retorna "1"

#2 LK<CR>
2,1<CR>

Boolean = qx.Lock /* Retorna "True".

IMPORTANTE. Cuando LOCK es emitido la RIAC no asumirá la clave asignada en posteriores DEFINE KEYs, este comando es irreversible.

NOTA. Ver los comandos COMPARE KEY y DEFINE KEY.

NR Integer = NumberRiac público/privado

El comando permite bautizar al módulo RIAC con una nueva dirección. El comando dispone de las modalidades privado y público, se describen por separado c/una de las alternativas.

Privado. La forma que toma el comando es la siguiente: #r NR d d, en donde "r" es la dirección actual de la RIAC y "d" es la dirección futura. El valor "d" se repite idénticamente dos veces a solo efectos de aumentar la seguridad del comando. Cuando el comando es aceptado, la RIAC devuelve el valor de "d", y a su vez lo inscribe en la memoria EEPROM. Tras realizar esta acción fuerza a la placa a realizar una operación de reset.

#1 NR 9 9<CR>
1,9<CR> /* La nueva dirección es 9.

I = qx.NumberRiac (9,9) /* La nueva dirección es 9, retorno I = 9.

Público. La forma que toma el comando es: #0 NR d d, cuando la RIAC recibe este comando fuerza el valor de "d" sobre la memoria EEPROM y a continuación realiza un RESET. Dado que se trata de un comando público no hay una respuesta de confirmación.

#0 NR 9 9<CR>

I = qx.NumberRiac (9,9)

Importante: En ambas modalidades, tanto pública como privada, el bautismo consiste en asignar una dirección que siempre se inscribe en la memoria EEPROM. Cuando se realiza el reset, la nueva dirección será asumida, si las minillaves 3 y 4 se hallan en ON, caso contrario la dirección es escrita en la EEPROM y podrá ser asumida cuando S3 y S4 pasen a ON.

OC	Integer = OpenCounter (c)	público/privado
-----------	---------------------------	-----------------

El comando produce la apertura del contador. Tras la emisión de OC, la RIAC aceptará los comandos RC y ZC (ReadCounter y ZeroCounter). Para el contador 4 los terminales del puerto 2 cumplen la función que se reseña.

- P2.0 Entrada de pulsos
- P2.1 Entrada de arranque (Run)
- P2.2 Entrada para puesta a cero (Zero)
- P2.3 Entrada de parada (Halt)

El comando OC pone inicialmente en estado alto todos los bornes P2.n. Las acciones de arranque, parada y puesta a cero pueden realizar mediante pulsadores externos ó bien mediante los comandos de escritura: WO, BS y BR, las acciones se desencadenan en la transición alto a bajo. En tanto el estado de las entradas puede obtenerse mediante RI y BI. Ejemplo de OC.

#1 OC 4<CR>	** apertura del contador 4
1,4	** Okey
n = qx.OpenCounter (4)	** apertura del contador 4, n = 4.

RC	String = ReadCounter (c)	privado
-----------	--------------------------	---------

Lee el estado del contador c. La respuesta contendrá el valor de cuenta que ha alcanzado el contador en el momento de la recepción del comando. El comando Read Counter es reconocido solo si previamente se ha ejecutado el comando OpenCounter. Tras CloseCounter el comando Read Counter no será reconocido.

#r RC c<CR>	
R,d<CR>	
#5 RC4<CR>	**Lectura del contador 4.
5,2348R<CR>	**Unidad 5 devuelve dato del contador, 2348R.
	**La R final indica contador activo (RUN).
s = qx.ReadCounter (4)	** Lectura del contador 4, retorna s = "2348R"
	** El contador está activo R (RUN) y
	** el estado de cuenta es 2348.

Contador 4, modalidad de operación. El contador 4 es un contador ascendente de 16 bit, el valor máximo es 65535, cuando la cuenta supera el valor máximo, la RIAC agrega un signo "+" al valor de cuenta. Como señal de arrastre/desborde. El signo "+" no vuelve a aparecer en las sucesivas lecturas hasta tanto no ocurra un nuevo desborde. Esta modalidad operativa permite realizar cuentas en precisión múltiple utilizando el signo "+" como arrastre.

#5 RC 4<CR>
5,+192R<CR>

** Lectura del estado del contador 4.

** El signo "+" indica desborde.

El estado actual del contador es 192 y considerando el signo "+" ha de sumársele 65536 para lograr el número total de cuentas.

$$192 + 65536 = 65728.$$

Cuando el contador fue forzado a cero, la RIAC agrega un espacio (" ") al valor de la cuenta, el signo no vuelve a aparecer en las sucesivas lecturas hasta tanto no haya una nueva puesta a cero.

Tras el valor de cuenta aparece el estado actual del contador, R (Run) indica que se halla activo; en tanto H (Halt) indica que está detenida la cuenta.

Ver comandos OC y CC, mas detalles bajo el título "Contador 4" del manual RIAC-QF Descripción técnica.

RI	Integer = ReadInput (p)	privado
-----------	-------------------------	---------

Lee el port de entrada p, de la RIAC r. La respuesta contendrá la información del estado de los bits del port p.

#r RI p<CR>
r, d<CR>

#5 RI 1<CR>
5,32<CR>

** Lee unidad 5, port 1.
** Unidad 5, dato leído 32

I = qx.ReadInput (1)

** Lee port 1, dato leído I = 32.

RT	Integer = RealTime (n, m)	privado
-----------	---------------------------	---------

El comando indica a la unidad afectada que envíe datos en tiempo real. Una vez emitido el comando RT, la RIAC envía el estado de todos sus terminales, digitales y analógicos, a intervalos regulares, sin que medie para ello nuevos pedidos de la unidad maestra. El intervalo de tiempo es fijado por el producto de los parámetros n, m, según la expresión que sigue: (n.m)/100 [Seg]. Los valores de n y m comprendidos entre 0 y 255.

Si se opera con más de una RIAC sobre una línea RS485, solo una de ellas debe operar en el modo RT. Se sugiere en tanto no realizar operaciones con las restantes RIACs. Se ilustra la secuencia para una RIAC con entradas analógicas.

```
#r RT n m<CR>
r,1<CR>
```

** Retorna "1", indica RT activo.

```
#3 RT 15 10<CR>
3,1<CR>
```

I = qx.RealTime (15,10)

** Retorna I = 1.

La respuesta se produce cada $(15.10)/100 = 1.5$ Seg, con la secuencia que se indica:

<STX> <CR>	Start of text(código ASCII2)
3,23, 45,255, 10,12, 66,78, 82<CR>	Equivale al comando AA
3,45<CR>	Equivale al comando RI
3,14<CR>	Equivale al comando GO
<ETX><CR>	End of text(código ASCII3).

El envío automático de la RIAC cesa cuando esta recibe un nuevo comando RT, en donde el producto n por m es igual a cero. Ejemplo: #3 RT 0 0<CR>, este comando debe transmitirse inmediatamente tras haber recibido <ETX><CR> para evitar posibles colisiones de datos sobre la línea de comunicaciones.

El retorno será:

```
3,0<CR> el "0" indica RT inactivo.
```

Ver nota de aplicación 3, "Intervalo de muestreo" y ejemplos en el diskette/CD.

RS	Integer = Reset	público/privado
-----------	-----------------	-----------------

Fuerza a la RIAC a la condición de RESET, el comando es equivalente a pulsar la tecla homónima. La RIAC devuelve el status que disponía previo al instante del reset.

```
#r RS<CR>
r,d<CR>
```

```
#3 RS<CR>
3,0<CR>
```

I = qx.Reset

** Se efectúa RESET sobre la unidad 3.
** La unidad 3 devuelve el status, 0.

** Reset, retorna el previamente el status, I = 0.

SI	Single = SensorInput	privado
-----------	----------------------	---------

Sensor Input retorna el valor de un canal analógico en la magnitud de la variable sensada. Por ejemplo: 60,3 °C, 4817Kg. El usuario debe solicitar previamente la incorporación de este comando. Consulte por la línea de sensores de microAXIAL.

ST	Integer = Status	privado
-----------	------------------	---------

Lee el estado interno de la RIAC, este corresponde a un valor numérico que identifica la situación detectada por la RIAC con cada comando emitido. La tabla muestra el código de status y su significado.

- 0: comando sin error detectado
- 1: código recibido no válido
- 2: el comando no puede ser emitido como público
- 3: formato no válido (no ubica dirección, ó código con exceso caracteres)
- 4: error de paridad
- 5: exceso de caracteres en el comando
- 6: número de campos no válido
- 7: campo con exceso de dígitos / falta la dirección de la RIAC
- 8: parámetro numérico incorrecto
- 9: velocidad seleccionada no disponible
- 10: pérdida de caracteres (overrun)
- 11: bit de parada no válido
- 12: 9° bit no corresponde
- 13: exceso de dígitos y/o alguno de ellos no válido.

```
#r ST<CR>
r,d<CR>
```

```
#3 ST<CR>
3,0<CR>
```

I = qx.Status

** Se efectúa RESET sobre la unidad 3.
** La unidad 3 devuelve el status, 0.

** Valor de estado 0.

VB	Single = VoltBalanced (c)	privado, AD 16 bits
-----------	---------------------------	---------------------

Esta función retorna el valor en voltios de los canales analógicos balanceados. Al valor numérico se le adjunta el signo. En situación de desborde el valor provisto es el máximo del fondo de escala seleccionado (ver comando GN).

```
#r VB c<CR>
r,d<CR>
```

#1 VB 3<CR>
r, -1.284<CR>

/* Lectura canal 3
/* Tensión leída -1.284V.

sgl = qx.VoltBalanced (3)

/* Tensión leída sgl = -1.284V.

VI Single = VoltInput (c) privado, AD 10 bits

Permite la lectura en voltios de cada uno de los canales. Para versiones con entrada unipolar se debe tomar el valor sin cambios. En caso de bipolar se deberá restar 2,5V.

#r VI c<CR>
r, d<CR>

/* Lectura en voltios del canal c.

sgl = qx.VoltInput (c)

/* Lectura en voltios del canal c.

Ejemplo, caso unipolar

#7 VI 3<CR>
7,2.018<CR>

/* Lectura del canal 3
/* Lectura del canal 3, 2.018 voltios.

Ejemplo, caso bipolar

#7 VI 3<CR>
7,2.018

/* Lectura del canal 3
/* Lectura 2.018

Voltios = 2.018 - 2.500

Voltios = -0.482

VI Single = VoltInput (c) privado, AD 16 bits

Esta función retorna el valor en voltios de los canales analógicos desbalanceados. Al valor numérico se le adjunta el signo. En situación de desborde el valor provisto es el máximo del fondo de escala seleccionado (ver comando GN).

#r VI c<CR>
r, d<CR>

#1 VI 3<CR>
1, +2.973<CR>

/* Lectura en voltios de canal 3
/* Valor leído +2.973V.

#1 VI 5<CR>
1, -1.284<CR>

sgl = qx.VoltInput (5)

/* Lectura del canal 5. Valor leido sgl = -1.284.

VL String = Velocity (a) público/privado

El comando permite asignar al módulo RIAC una nueva velocidad. La velocidad se anota como una cadena de cuatro caracteres.

#r VL a<CR>
r, s<CR>

#3, 9600<CR>
3, 9600<CR>

/* Unidad 3, velocidad 9600
/* Aceptado

string = qx.Velocity (9600) /* Velocidad 9600.

Los valores de velocidad son los que se indican a continuación: 1200, 2400, 4800, 9600, 19K2, 38K4, 57K6, 115K.

19K2	equivale a	19200 baudios
38K4	equivale a	38400 baudios
57K6	equivale a	57600 baudios
115K	equivale a	115200 baudios

WO Integer = WriteOutput (p, d) público/privado

Escribe en la unidad r, port p, el dato d. Luego de la escritura sobre el port p, la RIAC procede a la lectura del mismo y devuelve por respuesta el valor leído, d.

#r WO p d<CR>
r, d<CR>

#2 WO 0 3<CR>
2,3<CR>

/* RIAC 2, port 0, dato 3
/* RIAC 2, confirma dato: 3

I = qx.WriteOutput (0,3) /* Envía dato 3 a puerto 0. Retorna valor leído.

WT Integer = WatchDogTest privado

El comando WT sirve para verificar si opera el Watch dog. A tal efecto y una vez recibido el comando, retoma el valor '1' como reconocimiento del comando, luego el microcontrolador queda en un lazo infinito, sin disparar el monoestable que constituye el watch dog, este tras un breve lapso fuerza un reset. El microcontrolador reinicia así desde el origen.

#r WT<CR>
r, 1<CR>

#3 WT<CR>
3,1<CR>

/* Se realiza una prueba en la unidad 3,
** Unidad 3 devuelve '1', a continuación
unreset es forzado por elwatch dog.

I = qx.WatchDog

ZB Integer = ZeroBalanced (c) público/privado, AD 16 bits

Permite definir el cero del sistema. Este comando puede aplicarse a cada una de las entradas balanceadas. La tensión de entrada presente al aplicar el comando ZB se toma como "valor cero" ó tensión a partir del cual se refieren todas la lecturas posteriores sobre el canal.

El "valor cero" se guarda en memoria no volátil, por lo tanto una pérdida de energía ó una situación de reset no altera el valor logrado. El "valor cero" se guarda junto a la ganancia presente en ese momento.

Si en medidas posteriores se modifica el valor de ganancia, automáticamente se realizan todas las correcciones para garantizar la exactitud de la medida. Es recomendable realizar el ajuste de cero con la ganancia que ha de emplearse en la práctica.

```
#r ZB c<CR>          /* C: canal  
r,c                      /* Ok, realizado.  
  
#5 ZB 2<CR>           /* "ajuste de cero" del canal 2  
5,2                      /* Ok, realizado.  
  
I = qx.ZeroBalanced (2)    /* Retorna I = 2.
```

Nota 1. El ajuste ZI y ZB pueden coexistir sin interferencias.

Nota 2. Ver nota de aplicación Nº 4, Adquisidores, ajuste de cero y fondo de escala.

Ajuste en planta. En planta se realiza el procedimiento que sigue:

- Se conectan entre si las entradas +Cn, -Cn y Agnd.
- Se selecciona la ganancia g = 3 (A = 8)
#r GN 3<CR>
- Se emiten los comandos de Zero para cada canal
#r ZB 0 <CR>
...
#r ZB 7<CR>

El ajuste en planta consiste en tomar un valor de cero absoluto con ganancia g = 3. El usuario puede optar por definir el cero para otro valor de ganancia. También puede repetir el proceso de ajuste descripto con la placa ya instalada dentro de su aplicación.

ZC Integer = ZeroCounter (c) público/privado

Fuerza el contador c al valor cero. El comando es aceptado sólo si previamente se ha ejecutado el comando OpenCounter. Tras Close Counter el comando ZeroCounter no es reconocido.

#r ZC c<CR>
r,c<CR>

#1 ZC (4)
1,4<CR>

I = qx.ZeroCounter (4)

/* Fuerza a cero el contador 4, retorna I = 4.

ZI Integer = ZeroInput (c) público/privado, AD 16 bits

Permite definir el cero del sistema. Este comando puede aplicarse a cada una de las entradas desbalanceadas. La tensión de entrada presente al aplicar el comando ZI se toma como "valor cero" ó tensión a partir de la cual se refieren todas la lecturas posteriores sobre el canal.

El "valor cero" se guarda en memoria no volátil, por lo tanto una pérdida de energía ó una situación de reset no altera el valor logrado. El "valor cero" se guarda junto a la ganancia presente en ese momento.

Si en medidas posteriores se modifica el valor de ganancia, automáticamente se realizan todas las correcciones para garantizar la exactitud de la medida. Es recomendable realizar el ajuste de cero con la ganancia con que se opera en la práctica.

```
#r ZI c<CR>          /* C = canal  
r,c                      /* Ok, realizado.  
  
#5 ZI 2<CR>           /* "ajuste a cero" del canal 2  
5,2                      /* Ok, realizado.  
  
I = qx.ZeroInput (2)      /* Retorna I = 2.
```

Nota 1. El ajuste ZI y ZB pueden coexistir sin interferencias.

Nota 2. Ver nota de aplicación Nº 4, Adquisidores, ajuste de cero y fondo de escala.

Ajuste en planta. En planta se realiza el procedimiento que sigue:

- Se conectan todas la entradas al borne Agnd.
- Se selecciona la ganancia, g = 3 (A = 8)
#r GN 3<CR>

- Se emiten los comandos de Zero para cada canal

```
#r ZI 0 <CR>
...
#R ZI 7<CR>
```

NOTAS

El usuario puede repetir el proceso de ajuste descripto con la placa ya instalada dentro de su aplicación, ó bien con otro valor de ganancia.

Long = AnalogicChanel (c)	comando virtual QX
---------------------------	--------------------

Retorna el valor digitalizado del canal c. El comando virtual AnalogicChanel opera combinado con los comandos AllAnalogic y RealTime. Estos dos últimos, mantienen el valor de cada canal.

```
s = AllAnalogic          /* Lectura de todos los canales
For c = 0 to 7
  l(c) = AnalogicChannel (c)    /* en l (c) se obtiene el valor de cada canal
Neet c
```

Integer = InputPort (p)	comando virtual QX
-------------------------	--------------------

Retorna el valor del puerto p. El comando virtual InputPort opera combinado con el comando RealTime, éste mantiene el valor de cada port. Ejemplo,

```
L = InputPort (2)          /* en "l" se obtiene el valor del puerto 2.
```