AcQuisition Technology bv

Headquarters:
Raadhuislaan 27a
5341 GL OSS
THE NETHERLANDS

Postal address:
P.O. Box 627
5340 AP OSS
THE NETHERLANDS

Phone: +31-412-651055
Fax: +31-412-651050
Email: info@acq.nl
WEB: http://www.acq.nl

M323

Quadrature Incremental Encoder Interface
M-module

User Manual

Version 1.1

Disclaimer:

The information in this document has been carefully checked and is believed to be entirely reliable. However, no responsibility is assumed for inaccuracies. AcQuisition Technology does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. AcQuisition Technology products are not designed, intended, or authorized for use as components in systems intended to support or sustain life, or for any other application in which the failure of an AcQuisition Technology product could create a situation where personal injury or death may occur, including, but not limited to AcQuisition Technology products used in defence, transportation, medical or nuclear applications. Should the buyer purchase or use AcQuisition Technology products for any such unintended or unauthorized application, the buyer shall indemnify and hold AcQuisition Technology and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that AcQuisition Technology was negligent regarding the design or manufacture of the part.

Printed in The Netherlands.

CONTENTS

1. INTRODUCTION . . . 3
1.1. VALIDITY OF THE MANUAL . . . 3
1.2. PURPOSE . . . 3
1.3. SCOPE . . . 3
1.4. DEFINITIONS, ACRONYMS AND ABBREVIATIONS . . . 3
1.5. NOTES CONCERNING THE NOMENCLATURE . . . 4
1.6. OVERVIEW . . . 4

2. PRODUCT OVERVIEW . . . 5
2.1. INTRODUCTION . . . 5
2.2. TECHNICAL OVERVIEW . . . 5

3. INSTALLATION AND SETUP . . . 7
3.1. UNPACKING THE HARDWARE . . . 7
3.2. BOOT ROM . . . 7
3.3. JUMPER SETTINGS . . . 8
3.3.1. BOOT SELECTION JUMPER J2 . . . 8
3.4. CONNECTING THE MODULE . . . 9
3.4.1. PIN ASSIGNMENTS . . . 10
3.4.2. CONNECTION QUADRATURE ENCODER DEVICES . . . 13

4. FUNCTIONAL DESCRIPTION . . . 15
4.1. BLOCK DIAGRAM . . . 15
4.2. QUADRATURE DECODER FUNCTION . . . 16
4.3. I/O INTERFACE . . . 17
4.4. NOISE IMMUNITY . . . 17
4.5. PERFORMANCE . . . 18
4.6. M-MODULE INTERFACE . . . 18
4.6.1. MEMORY MAP . . . 18
4.6.2. DUAL PORTED MEMORY . . . 18
4.6.3. PAGE REGISTER . . . 19
4.6.4. CONTROL REGISTER . . . 19
4.6.5. IDENTIFICATION EEPROM . . . 20
4.6.6. INTERRUPT HANDLING . . . 20
4.7. BOOTING THE M323 . . . 21
4.7.1. BOOTING FROM ROM . . . 21
4.7.2. BOOTING FROM RAM . . . 21

5. LOCAL FIRMWARE . . . 23
5.1. HOST INTERFACE . . . 23
5.2. COMMAND SET . . . 25
5.3. GENERAL COMMANDS . . . 26
5.4. HOST INTERRUPTS . . . 27
5.5. ENCODER CONTROL COMMANDS . . . 29

6. SOFTWARE . . . 33
6.1. APIS SUPPORT . . . 33
6.1.1. CONCEPT . . . 33
6.1.2. API . . . 33
6.1.3. CODE GENERATION . . . 34
6.2. TYPE DEFINITIONS AND STRUCTURES . . . 34

6.3. LIBRARY FUNCTION RETURN VALUES 35
6.4. FUNCTION REFERENCE 36
6.5. SOFTWARE DISTRIBUTION 39

7. ANNEX 41
7.1. BIBLIOGRAPHY 41
7.2. COMPONENT IMAGE 41
7.3. TECHNICAL DATA 42
7.4. DOCUMENT HISTORY 42

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

1. INTRODUCTION

1.1. VALIDITY OF THE MANUAL

This user manual is of revision 1.1. The manual is valid for the M323/R0.x from AcQuisition Technology bv, running firmware version R1.x.

1.2. PURPOSE

This manual serves as instruction for the operation of the M323 Quadrature Incremental Encoder input module, the connection of encoders and the integration on an M-module carrier. Furthermore it gives the user additional information for special applications and configurations of the product. The APIS-based example software and library are also discussed. Detailed information concerning the individual assemblies (data sheets etc.) are not part of this manual. In the annex you will find a bibliography.

1.3. SCOPE

The scope of this manual is the usage of the M323 Quadrature Incremental Encoder M-module.

1.4. DEFINITIONS, ACRONYMS AND ABBREVIATIONS

AcQ	AcQuisition Technology bv
APIS	AcQ Platform Interface Software
M-module	Mezzanine I/O concept according to the M-module specification
Platform	Combination of hardware and operating system

1.5. NOTES CONCERNING THE NOMENCLATURE

Hex numbers are indicated with a leading “0x”-sign: for example: 0x20 or 0xff.

File names are represented in italic: *filename.txt*.

A code example is printed in `courier`.

The jumpers are designated by a 'J', and a serial number. When specifying whether a jumper should be connected or removed it is referred to solely by this designation if it has only one position (e.g., 'J5 connected'). However, if the jumper has more than one position, it is also indicated which pins are connected to each other (e.g. 'J8,1-2'). Pin 1 of a jumper is always marked in the configuration diagram.

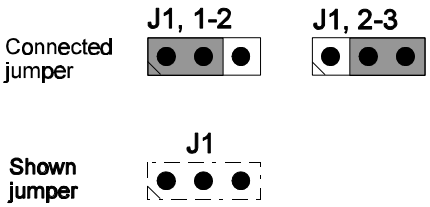


Figure 1 Example jumper nomenclature

In some illustrations jumpers are shown merely for purposes of orientation. In this case they are indicated with a dotted line. Their correct setting is described in another chapter.

Active-low signals are represented by a trailing asterisks (i.e. IACK*).

1.6. OVERVIEW

Chapter 1 contains an introduction, chapter 2 contains an overview of the product described in this user manual. Refer to chapter 3 for information on the installation and setup of the product. Chapter 4 contains a detailed description of the M323, chapter 5 describes the local firmware and chapter 6 the host software. Chapter 7 is the Annex containing technical data, the document history etc.

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

2. PRODUCT OVERVIEW

2.1. INTRODUCTION

The M323 M-module is an Intelligent quadrature encoder input M-module, based on an MC68332 micro controller. The M323 can be configured for interfacing either quadrature incremental encoders with or without an index channel. In index-mode up to four encoders with two quadrature channels and an index channel can be connected. In non-index-mode up to six encoders with two quadrature channels can be connected.
The M323 operates on two encoder input channels and decodes a pair of out-of-phase signals in order to increment or decrement a counter. The M323 is particularly useful for decoding position and direction information from a slotted encoder in motion control systems.
The MC68332 executes local firmware, downloaded by the host through the M-module interface or booted from ROM. Interaction with the host is done via a command structure in dual ported memory and a mailbox which provides a polling and interrupt mechanism.

2.2. TECHNICAL OVERVIEW

- MC68332 local 32-bit micro controller
- 64kByte RAM shared with the M-module bus
- Up to 1MBit local EPROM or FLASH memory in socket
- A08/D16 M-module interface
- Mailbox with polling, host interrupt and local interrupt mechanism
- Identification EEPROM
- Executes local firmware
- Firmware bootable from local ROM or downloadable via the M-module bus
- Controls up to four encoders with index or six encoders without index
- Inputs are optical isolated industrial inputs
- Industrial inputs can be adapted for a wide variety of signal ranges
- Module can accept 2 channel quadrature squarewave encoder outputs
- Index inputs can be configured for rising edge, falling edge, both edges or level sensitive
- Each encoder input is assigned to a 32-bit counter
- Index signal can be used to load the corresponding counter with a preset value
- Counter value during movement has one LSB uncertainty
- Counter value at ease is accurate
- Good noise immunity

Intentionally left blank.

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

3. INSTALLATION AND SETUP

3.1. UNPACKING THE HARDWARE

The hardware is shipped in an ESD protective container. Before unpacking the hardware, make sure that this takes place in an environment with controlled static electricity. The following recommendations should be followed:

- Make sure your body is discharged to the static voltage level on the floor, table and system chassis, by wearing a conductive wrist-chain connected to a common reference point.
- If a conductive wrist-chain is not available, touch the surface where the board is to be put (like a table, a chassis etc.) before unpacking the board.
- Leave the board only on surfaces with controlled static characteristics, i.e. specially designed anti static table covers.
- If handling the board over to another person, first touch this persons hand, wrist etc. to discharge any static potential.

IMPORTANT: Never put the hardware on top of the conductive plastic bag in which the hardware is shipped. The external surface of this bag is highly conductive and may cause rapid static discharge causing damage. (The internal surface of the bag is isolating.)

Inspect the hardware to verify that no mechanical damage appears to have occurred. Please report any discrepancies or damage to your distributor or to AcQuisition Technology immediately and do not install the hardware.

3.2. BOOT ROM

The M323 features a 32-pin JEDEC socket that can fit a (FLASH) EPROM containing the firmware. The use of Flash EPROM's, type AM29F010 from AMD, is recommended.
The MC68332 CPU boots from either dual ported RAM or local ROM. When booting from ROM, the local CPU will start running after a system reset. When booting from RAM, firmware must be downloaded and started by the host.
The boot method is configurable with jumper J2 for details please refer to section 3.3.1 and 4.7.

3.3. JUMPER SETTINGS

In the following paragraphs the jumper setting of the M323 is described. The figure below shows the location and orientation of the jumper.

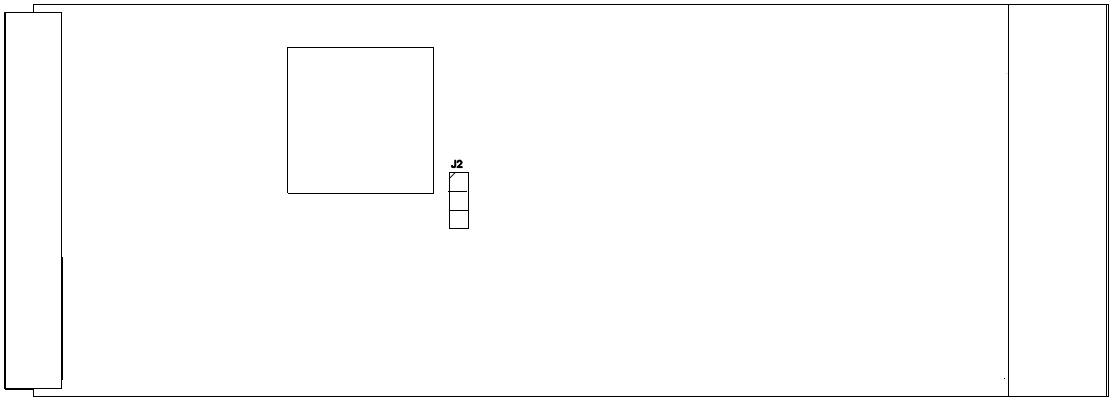


Figure 2 M323 Jumpers

3.3.1. BOOT SELECTION JUMPER J2

With jumper J2, the M323 can be configured for booting from ROM or booting from RAM. The figure below shows jumper J2 positions for both configurations.

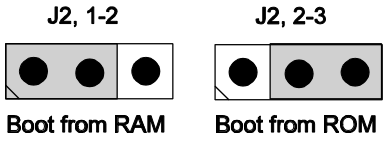


Figure 3 Boot device selection

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

3.4. CONNECTING THE MODULE

This section gives an overview of the I/O pins of the M323 which are available on the DSUB-25 P3 connector at the front of the module and the 24-pole male P2 header at the component side of the module. The figure below illustrates the position of P2 and P3. P1 is the M-module bus connector, a description of the pin-assignments of P1 is beyond the scope of the document.

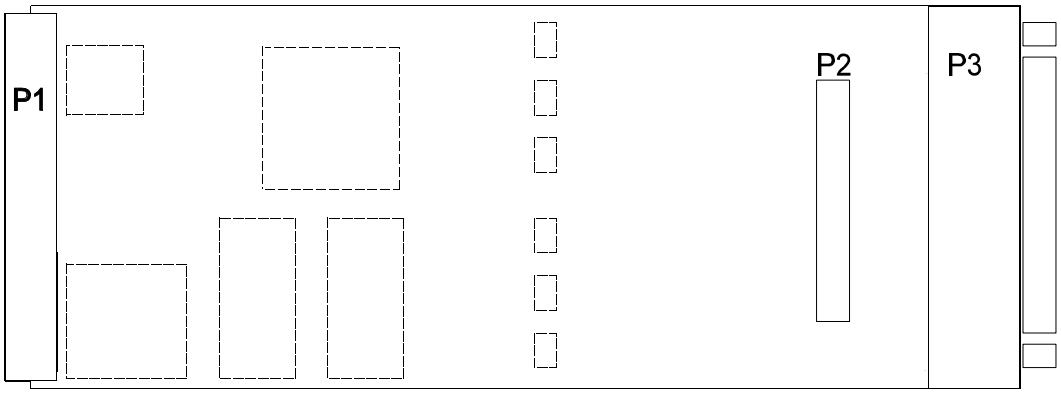


Figure 4 M323 Connector Position

The M323 can be configured to accept up to four incremental encoders with index (index-mode) or six incremental encoders without index (non-index-mode). The connector pin assignments depend on the operating mode of the M323. The connector layouts in both operating modes are explained. For details on configuration of the operating mode refer to chapter 5 . Finally some examples on connecting encoder devices are given.

3.4.1. PIN ASSIGNMENTS

Pin assignments M323 in index-mode			
Signal Name	P3-Pin #	P2-Pin #	Description
INA0+	1	1	Positive input signal A channel 0
INA0-	14	2	Return for input signal A channel 0
INB0+	2	3	Positive input signal B channel 0
INB0-	15	4	Return for input signal B channel 0
INI0+	3	5	Positive input Index channel 0
INI0-	16	6	Return for Index channel 0
INI1+	4	7	Positive input Index channel 1
INI1-	17	8	Return for Index channel 1
INA1+	5	9	Positive input signal A channel 1
INA1-	18	10	Return for input signal A channel 1
INB1+	6	11	Positive input signal B channel 1
INB1-	19	12	Return for input signal B channel 1
INA2+	7	13	Positive input signal A channel 2
INA2-	20	14	Return for input signal A channel 2
INB2+	8	15	Positive input signal B channel 2
INB2-	21	16	Return for input signal B channel 2
INI2+	9	17	Positive input Index channel 2
INI2-	22	18	Return for Index channel 2
INI3+	10	19	Positive input Index channel 3
INI3-	23	20	Return for Index channel 3
INA3+	11	21	Positive input signal A channel 3
INA3-	24	22	Return for input signal A channel 3
INB3+	12	23	Positive input signal B channel 3
INB3-	25	24	Return for input signal B channel 3

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

Pin assignments M323 in non-index-mode			
Signal Name	P3-Pin #	P2-Pin #	Description
INA0+	1	1	Positive input signal A channel 0
INA0-	14	2	Return for input signal A channel 0
INB0+	2	3	Positive input signal B channel 0
INB0-	15	4	Return for input signal B channel 0
INA4+	3	5	Positive input signal A channel 4
INA4-	16	6	Return for input signal A channel 4
INB4+	4	7	Positive input signal B channel 4
INB4-	17	8	Return for input signal B channel 4
INA1+	5	9	Positive input signal A channel 1
INA1-	18	10	Return for input signal A channel 1
INB1+	6	11	Positive input signal B channel 1
INB1-	19	12	Return for input signal B channel 1
INA2+	7	13	Positive input signal A channel 2
INA2-	20	14	Return for input signal A channel 2
INB2+	8	15	Positive input signal B channel 2
INB2-	21	16	Return for input signal B channel 2
INA5+	9	17	Positive input signal A channel 5
INA5-	22	18	Return for input signal A channel 5
INB5+	10	19	Positive input signal B channel 5
INB5-	23	20	Return for input signal B channel 5
INA3+	11	21	Positive input signal A channel 3
INA3-	24	22	Return for input signal A channel 3
INB3+	12	23	Positive input signal B channel 3
INB3-	25	24	Return for input signal B channel 3

The figure below illustrates the connector layout in index-mode:

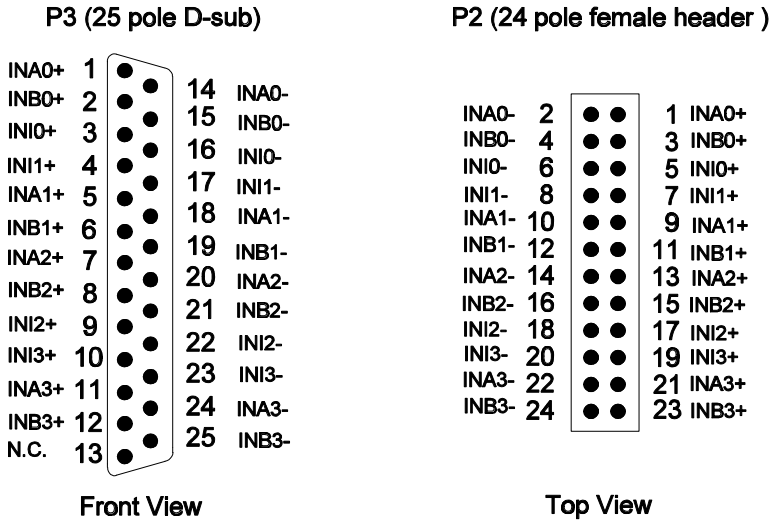


Figure 5 Connector layout, M323 with index

The figure below illustrates the connector layout in non-index-mode:

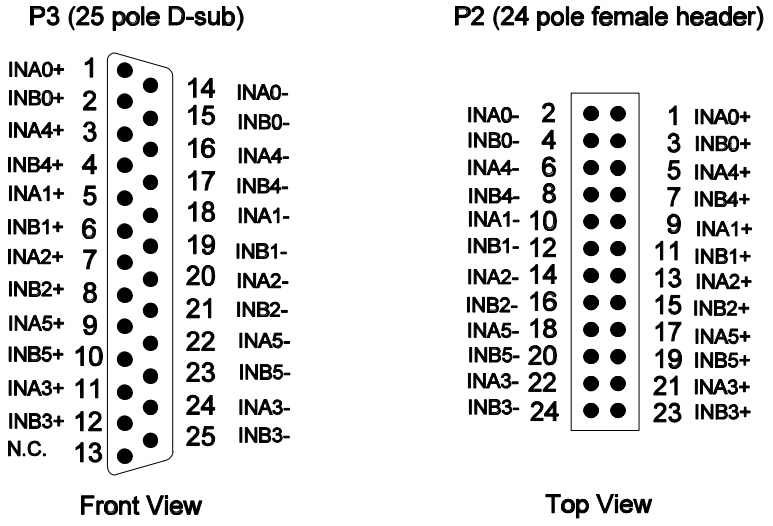


Figure 6 Connector layout, M323 without index

For cabling, shielded multi-cable must be used with metal DSUB-25 connector shells. The shield must be coaxial terminated to the metal shell.

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

3.4.2. CONNECTION QUADRATURE ENCODER DEVICES

- Example for connecting an encoder device with 2 quadrature TTL outputs and index, to channel 0 of the M323:

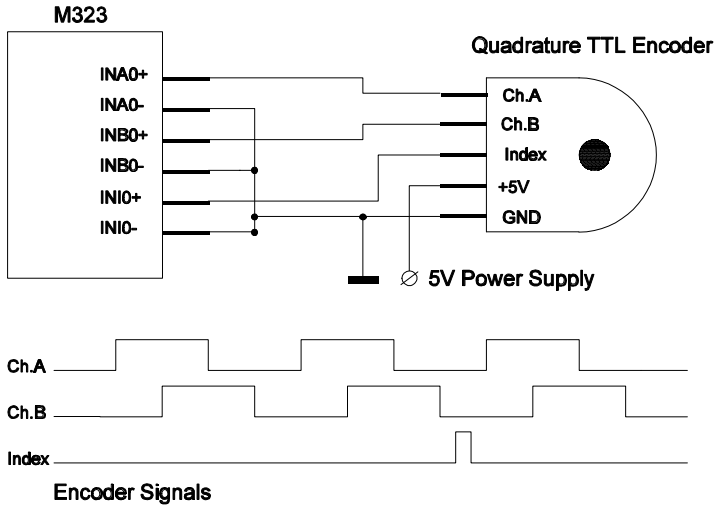


Figure 7 Connecting TTL Encoder to M323

NOTE: Quadrature TTL encoder devices in general are capable of sinking enough current for direct connection to the M323 however some encoder devices are not capable of sourcing enough current, in this case pull-up resistors between the channel inputs and the power-supply must be connected. For details refer to the application notes of the encoder device. For details and on the input circuitry and signal ratings please refer to section 4.3.

- Example for connecting an encoder device with 2 quadrature RS422 outputs, to channel 1 of the M323:

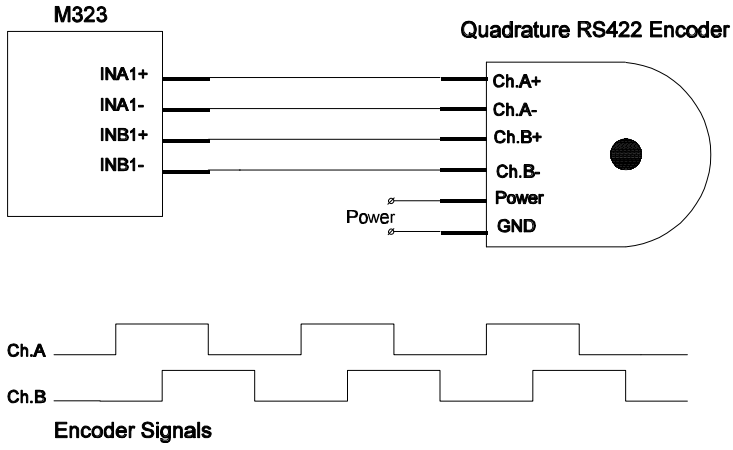


Figure 8 Connection RS422 Encoder to M323

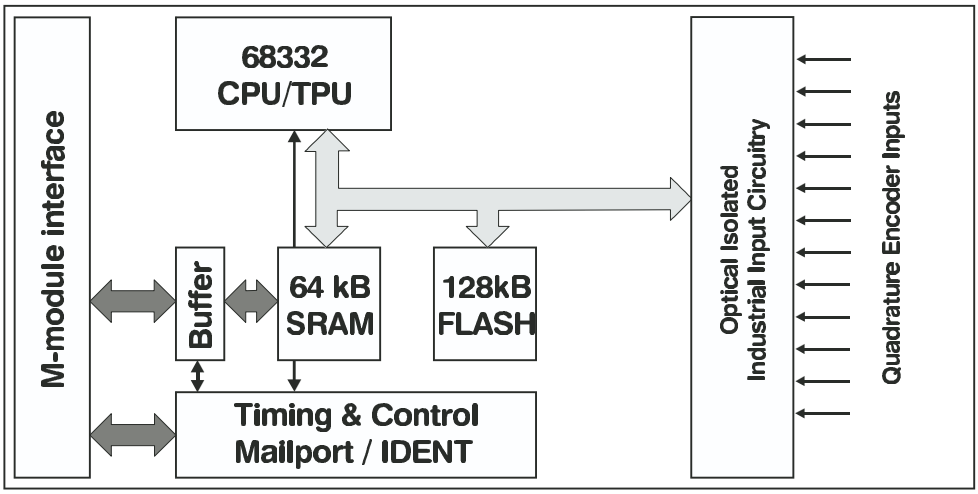
Intentionally left blank.

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

4. FUNCTIONAL DESCRIPTION

This chapter gives a detailed description of the M323. The M-module interface is described as well as the encoder inputs.

4.1. BLOCK DIAGRAM



As shown by the block diagram the M323 is based on a MC68332 micro controller with SRAM for execution of local firmware and FLASH for booting the local firmware. Communication with the host system takes place through the M-module interface. The front-end of the module consists of optical isolated industrial inputs for connecting quadrature encoder devices and optional index devices. The on-chip Time Processing Unit (TPU) of the micro controller is used for implementing the quadrature decoding function.
The following paragraphs contain detailed information regarding the sub components of the M323.

4.2. QUADRATURE DECODER FUNCTION

The M323 features a quadrature decode algorithm that operates on two encoder input channels and decodes a pair of out-of-phase signals in order to increment or decrement a counter. The M323 is particularly useful for decoding position and direction information from a slotted encoder in motion control systems. The figure below shows a typical application.

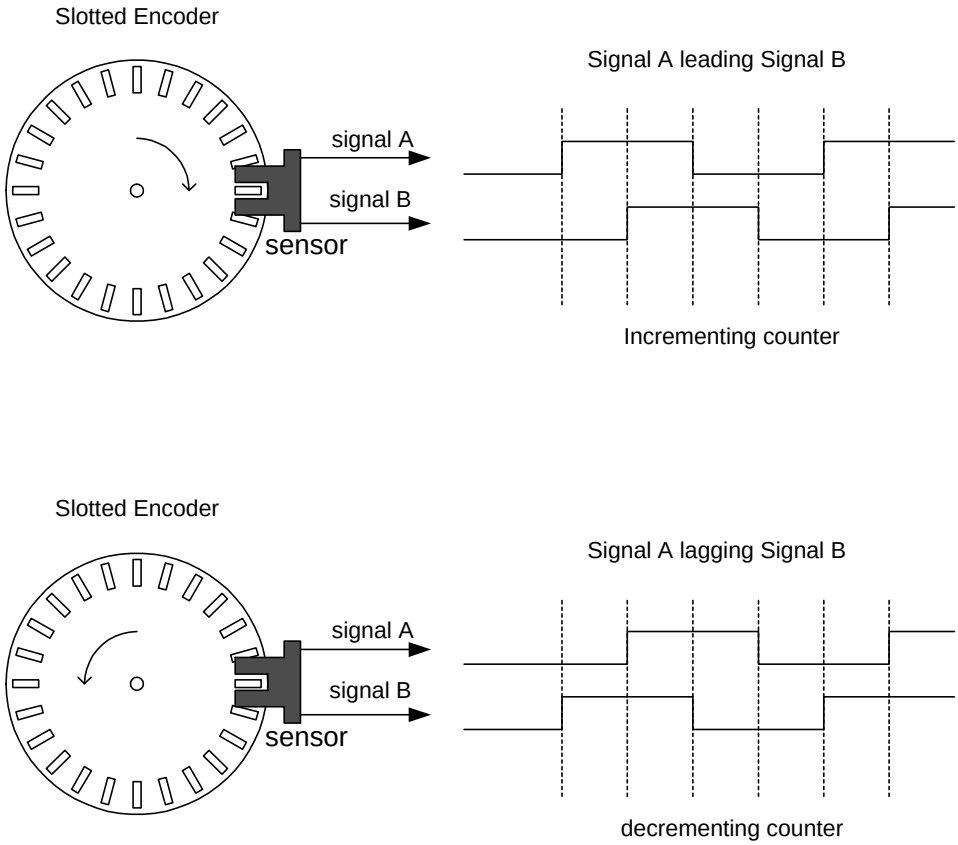


Figure 10 Typical M323 application

The M323 uses two input channels to decode quadrature signals into a 32-bit counter. The counter is updated when a valid transition is detected on either one of the two inputs, full 4x resolution is derived from the encoder signals. The counter is incremented or decremented depending on the lead/lag relationship of the two signals at the time of servicing the transition. The user can read the counter at any time. The counter is free running, overflowing to 0x00000000 or underflowing to 0xFFFFFFFF depending on the direction.

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

4.3. I/O INTERFACE

Each encoder input consists of an opto-coupler input, which is protected for reverse connection and has a leakage current protection. Default the encoder inputs are rated at 5V@1mA.
The figure below shows the input circuitry of encoder input A of channel 0, the remaining inputs have comparable input circuitry.

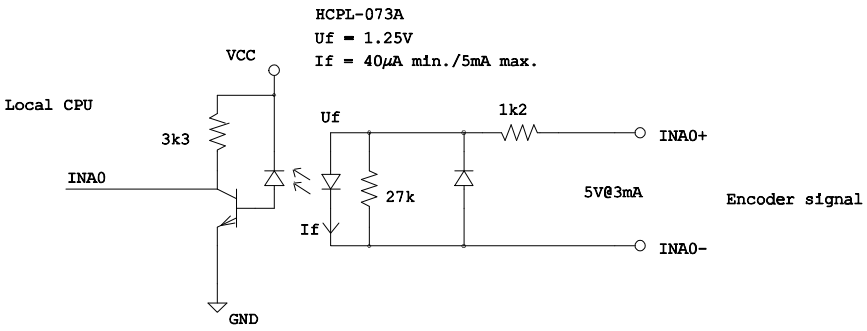


Figure 11 M323 encoder input

The M323 input circuitry is developed to accept 5V TTL compatible input signals. Transitions are detected around 1.3V. The minimum input current (If=40μA) for detecting a transition is approximately 86 μA, the maximum input current is 5 mA. To connect the M323 to signals larger than 5V, series resistors have to be applied. The table below contains values of series registers for common signal levels.

U Level	Switching Level	Rs
5 V	1,3V	-
12 V	1,3V	2k2
24 V	1,3V	4k7

4.4. NOISE IMMUNITY

To a large extend, the M323 hardware protect counters from erroneous updates due to noise. All input channel incorporate a digital filter which rejects pulses of less than 100nsec and guarantees to pass pulses of greater than 250nsec. In addition, when servicing a transition the M323 logic always checks the new signal state against the signal state of the last service, and if they are equal then no action is taken; This protects against noise pulses that are long enough to get through the digital filter, but not long enough to last from the actual transition time to the time that the M323 services the signal.

4.5. PERFORMANCE

The performance limit of the M323 is dependent upon the number of active encoder input channels. When one encoder input channel is used and no other TPU channels are active, the maximum transition rate is 100 kHz. When four channels are used simultaneously the maximum frequency for each channel is 50 kHz if six channels are used the maximum frequency is 33 kHz for each channel. The maximum rate of index pulses, regardless on which index channels the transitions are detected, is 5 kHz.

NOTE: Exceeding performance limits described above might result in erroneous behaviour.

4.6. M-MODULE INTERFACE

4.6.1. MEMORY MAP

The following table shows the address map of the M323 module. All addresses are relative to the base address of the module.

Offset	Width	Description
0x00	16 bit	Dual ported memory window
0x80	16 bit	Page register
0x82	16 bit	Control register
0xfe	16 bit	EEPROM register

4.6.2. DUAL PORTED MEMORY

The M323 features 64kByte dual ported memory. The dual ported memory on the M323 is divided in 512 pages of 128 bytes. The memory can be accessed by the host via a 128-byte memory window and a page register. The dual ported memory window resides at offset 0x00 to 0x7e of the M-module memory map.

CAUTION: The dual ported memory is 16 data bit wide and should only be written using 16 bit write accesses. Byte-writes to the dual ported memory are **NOT** supported, and will result in undefined behaviour.

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

4.6.3. PAGE REGISTER

The dual ported memory of the M323 is accessed using a page register for the upper address bits. The first 128 bytes of the address space of the M323 M-module is used as a "window" into the dual ported memory as described in the previous section.

PAGEREG Offset: 0x80

15 10	9	8	7	6	5	4	3	2	1	0
not used	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7
reset	x	x	x	x	x	x	x	x	x	x

Write only

A16-A7 Dual ported memory address bits
These bits determine the current dual ported memory page selected. A page has a length of 128 bytes (A6-A0).

4.6.4. CONTROL REGISTER

With the control register the local reset of the M323 M-module can be controlled. Furthermore the control register provides a mailbox. The mailbox controls interrupts to and from the module and provides a polling mechanism.

CTRL Offset: 0x82

15 4	3	2	1	0
not used	PBIT	RESET	LIRQ	HIRQ
reset:	0	1	0	0

Read/Write

PBIT Poll Bit
When read, this bit reflects the state of the Poll Bit either ‘0' or ‘1'. This bit can be cleared by the host by writing a one ‘1' and set by the local CPU. Writing a zero ‘0' has no effect.

RESET Local reset
When asserted, the M323 will be kept in reset state and pending interrupt requests will be cleared. After power-up the reset bit is set, however when the module is configured for booting from ROM (J2 in 2-3 position) the reset bit is cleared automatically.

LIRQ Local Interrupt request
When asserted an interrupt request to the local CPU (MC68332) will be made.

HIRQ Host Interrupt Request
This bit reflects the state of the host interrupt line, one '1' means asserted. Writing a one ‘1' to this location clears the pending interrupt.

4.6.5. IDENTIFICATION EEPROM

The identification of an M-module is implemented using a serial EEPROM with a 64*16 word organisation. The industry-standard component 93C46 is used in order to make the identification compatible throughout the complete range of available modules. Access to the identification EEPROM takes place through the following register:

EEPROM Offset: 0xFE

15 ... 4	3	2	1	0
not used	unused	CS	CLK	DI/O
	0	0	0	x

Read/Write

CS Chip-Select
: This bit corresponds to the chip select input of the EEPROM.

CLK Clock
: This bit corresponds to the clock input of the EEPROM.

DI/O Data input/output
: This bit corresponds to the data input of the EEPROM when writing, and data output of the EEPROM when reading.

For information on controlling the EEPROM refer to the NM93C46 data sheets. A software example can be found in the file *ideeprom.c* that is part of the software distribution. For information on the memory organization refer to the M-module Specification.

4.6.6. INTERRUPT HANDLING

The M323 is capable of generating interrupts of type A, software-end-of-interrupt. In the interrupt service routine the host must acknowledge the interrupt by writing a one to the host interrupt request bit in the control register. Because the M323 is not capable of delivering an interrupt vector, this must be handled by the carrier board.

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

4.7. BOOTING THE M323

The MC68332 micro controller of the M323 executes firmware from local memory. The module can be configured for booting from ROM or booting from RAM.

4.7.1. BOOTING FROM ROM

For this option an EPROM or FLASH containing valid firmware must be inserted in the JEDEC socket U15 and Jumper J2 must be set in the 2-3 position.
After power-up the module will boot from ROM without host interaction. When the PBIT in the control register is set the module is ready for accepting control commands.

Communication with the M323 takes place via a command structure that resides in the dual ported memory at page 4. For details on the command structure refer to section 5.1.

CAUTION: Write accesses to the dual ported memory at pages other than page 4, may result in erroneous behaviour.

4.7.2. BOOTING FROM RAM

When the module is configured for booting from RAM (jumper J2 in 1-2 position), at power-up the M323 is kept in reset state. Before any control operations are available, firmware has to be downloaded to the dual ported memory by the host.
Firmware must copied to the dual ported memory starting at offset 0 in page 0 using 16 bit wide write accesses.
The M323 local CPU must be started by clearing the RESET bit in the control register. When the PBIT in the control register is set the module is ready for accepting control commands.

Communication with the M323 takes place via a command structure in the dual ported memory at page 4. For details on the command structure refer to section 5.1.

CAUTION: Once the firmware is started any write accesses to the dual ported memory at pages other than page 4, may result in erroneous behaviour.

For details on downloading firmware please refer to the software distribution.

Intentionally left blank.

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

5. LOCAL FIRMWARE

This chapter is valid for the local firmware of revision R1.x.

5.1. HOST INTERFACE

The M323 features a command interface and mailbox register provided for communication between the host and the local CPU. The command structure resides at page 4 of the dual ported memory and the mailbox can be accessed via the control register on the M323. The command interface consists of a command field, a result field and a parameter field:

Offset *	Name	Description
0x00	command	command field
0x02	result	result field
0x04	parm0	parameter 0
0x06	parm1	parameter 1
0x08	parm2	parameter 2
...	...	...
0x22	parm15	parameter 15
0x44	irqstat	interrupt status

* The page register must be set to 4.

The M323 can accept commands at any time provided that a previous requested operation is completed. Before a command is given make sure the command field is empty (no outstanding command present). Then write the parameters if any to the parameter field. After that the command must be written to the command field. When a command is handled by the CPU, return parameters if any, are stored in the parameter field, the result is updated in the result field and finally the command field is cleared by the CPU. The CPU is ready to accept new commands. For a complete overview of the available commands and the usage refer to section 5.2.

The table below contains a list of the available result codes which can be found in the result field of the command structure.

Result Code	Hex	Description
NOERR	0x0000	no error
UNKCMD	0x8001	unknown command
PRANGE	0x8002	parameter out of range
EXCPERR	0x8003	general exception error
LOCBERR	0x8004	local bus error
LOCAERR	0x8005	local address error
LOCILL	0x8006	illegal instruction
SPURINT	0x8007	spurious interrupt
INVREQ	0x8008	invalid request
FLASHERR	0x800B	flash program error

The table below lists the interrupt status bits of the irqstat field:

Interrupt Status	Hex	Description
IRQ_BLANK	0x0000	Nothing to report
IRQ_INDEX_0	0x0001	signal index channel 0
IRQ_INDEX_1	0x0002	signal index channel 1
IRQ_INDEX_2	0x0004	signal index channel 2
IRQ_INDEX_3	0x0008	signal index channel 3
IRQ_EXCEPTION	0x0010	Internal exception error

An interrupt flag in the interrupt status field set, means that the corresponding event has occurred, the flag must be cleared by the host CPU. Interrupt flags are updated regardless whether or not the corresponding interrupt is enabled.

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

5.2. COMMAND SET

The table below gives an overview of the commands available to the host CPU.

Command	Hex	Description
DONE	0x0000	command completed
SYNC	0x5a5a	synchronize firmware, sign of live
VERSION	0x0001	returns version information
MSKI	0x0020	mask/un-mask interrupts
IACK	0x0030	acknowledge interrupts
INIT	0x0040	initialize module: index-mode/no-index-mode
CONFIG	0x0100	configure encoder channel: index input edge, rising/falling/both index function, enable/disable counter preset
PRESET	0x0200	write counter preset value, if 'counter preset on index' is disabled then the position counter will be loaded immediately
GETCNTR	0x0500	get counter value
GETINDEX	0x0600	get and clear channel index status get counter value captured on index get current level of index input
GETALLCNTR	0x0700	get value of all counters
DEBUG	*0x8001*	*service command for debug and test purposes*

Table entries represented in italic are no user commands, these commands are provided for service purposes. The following sections give a detailed description of the commands arranged by functionality.

5.3. GENERAL COMMANDS

This section describes general commands that are provided for obtaining firmware information and for setting the operating mode of the M323.

SYNC — Sign Of Live

Command: 0x5a5a
Inputs: None
Outputs: None

Function: This function is intended to check whether or not the MC68332 is running. The local software takes no special action and terminates the execution normally: set the result field to NOERR, set the POLL bit in the control register, clear the cmd field and generate a host interrupt if not masked.

VERSION — Get Module Information

Command: 0x0001
Inputs: None
Outputs: Version number

Function: This function is provided to provide version information of the module. As a result the output parameter 0 contains the revision of the local firmware. The firmware revision is a represented as a decimal number, for example 10 which means the firmware is of revision 1.0.

INIT — Select Operating Mode

Command: 0x0040
Inputs: Mode of operation
Outputs: None

Function: This function is provided to select the operating mode of the M323. The M323 can operate in two modes: four quadrature encoders with index or six quadrature encoders without index signal. Parameter 0 must contain the required configuration:

Parameter 0	Mode of operation
0	four channels with index
1	six channels without index

NOTE: The command INIT must be called to start decoding of encoder signals.

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

5.4. HOST INTERRUPTS

The M323 can generate host interrupts. There are various interrupt sources that use the host interrupt request. Commands are provided to (un)mask and acknowledge interrupts individually. This section gives a description of the interrupt related commands.

MSKI — (Un-)Mask Interrupts

Command: 0x0020
Inputs: Interrupt mask
Outputs: None

Function: A bit set in the interrupt mask will enable the corresponding interrupt, a bit cleared will disable the interrupt. Every interrupt source has a corresponding bit defined in the interrupt mask. Definition of the interrupt mask can be found in the table below.

Flag	Hex	Description
IRQ_BLANK	0x0000	nothing to report
IRQ_INDEX_0	0x0001	signal index channel 0
IRQ_INDEX_1	0x0002	signal index channel 1
IRQ_INDEX_2	0x0004	signal index channel 2
IRQ_INDEX_3	0x0008	signal index channel 3
IRQ_EXCEPTION	0x0010	internal exception error
IRQ_ALL	0xFFFF	all interrupt sources

IACK — Acknowledge Interrupts

Command: 0x0030
Inputs: Interrupt status
Outputs: None

Function: The M323 uses one interrupt line for all interrupt sources. Whenever an interrupt request is generated the pending interrupt must be cleared via the control register (see section 4.6.4). The nature of the event(s) responsible for generating the interrupt must be obtained from the interrupt status field in the command register. A bit set means the interrupt request is active. The IACK command must be executed to acknowledge pending interrupts. Parameter 0 must contain the interrupt status.
A bit set in the interrupt status will acknowledge the corresponding interrupt. Every interrupt source has a corresponding bit defined in the interrupt status. The status is defined as follows:

Flag	Hex	Description
IRQ_BLANK	0x0000	nothing to report
IRQ_INDEX_0	0x0001	signal index channel 0
IRQ_INDEX_1	0x0002	signal index channel 1
IRQ_INDEX_2	0x0004	signal index channel 2
IRQ_INDEX_3	0x0008	signal index channel 3
IRQ_EXCEPTION	0x0010	internal exception error
IRQ_ALL	0xFFFF	all interrupt sources

It is allowed to clear more than one pending interrupt simultaneously with the IACK command. As a result of the IACK command not acknowledged (pending) interrupts will cause a new host interrupt request.

The IACK command is a ‘special’ command because on completion the result field is not affected and the polling bit is not set.

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

5.5. ENCODER CONTROL COMMANDS

The commands in this section are provided to request a motion profile.

CONFIG — Configure Encoder Channel

Command: 0x0100
Inputs: Channel number, configuration parameters
Outputs: Undefined

Function: This function is provided to configure encoder channels. The table below contains an overview of the command parameters:

Parm #	Name	Description	Default
0	Channel	Channel number 0 to 5.	0
1	Index input signal	Configure index input: rising edge (0), falling edge (1) or both edges (2).	0
2	Index function	Enable (1) or disable (0) 'preset on index', when enabled the corresponding counter will be loaded with a preset value set with the PRESET command.	0

Note: Channel 4 and 5 are not available in 'index-mode'. Parameters 2 and 3 are not used in 'no-index-mode'.

PRESET — Load Preset Value

Command: 0x0200
Inputs: Channel number, preset value
Outputs: Undefined

Function: Write counter preset value. Parameter 0 contains the channel number and parameter 1 and 2 contain the preset value:

Parm #	Description
0	Channel number
1	Preset value upper 16-bits
2	Preset value lower 16-bits

When ‘preset on index’ is disabled the counter will be loaded immediately, if ‘preset on index’ is enabled with the CONFIG command then the counter will be loaded with the preset value when an index pulse is detected.

GETCNTR — Get Counter Value

Command: 0x0500
Inputs: Channel number
Outputs: Channel number, counter value

Function: Get counter value of a channel. Input parameter 0 must contain the channel number. After completion of the command output parameter 0 contains the requested channel number and output parameter 1 and 2 contain the 32-bit counter value. The table below contains a description of the output parameters:

Parm #	Description
0	Channel number
1	Counter value upper 16-bits
2	Counter value lower 16-bits

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

GETINDEX — Get and Clear Index Status

Command: 0x0600
Inputs: Channel number
Outputs: Channel number index status, input-pin state

Function: Get and clear the index status of a channel. Input parameter 0 must contain the channel number. After completion of the command output parameter 0 contains the requested channel number. Output parameter 1 and 2 contain the 32-bit counter value captured as a result of a transition of the index signal. Output parameter 3 contains the index flag which is zero if no index pulse has been detected or non-zero if an index has occurred. Output parameter 4 contains the current state of the index input, 0 if low or 1 if high. The table below contains a description of the output parameters:

Parm #	Description
0	Channel number
1	Captured counter value upper 16-bits
2	Captured counter value lower 16-bits
3	Index flag, 0 if no index detected or 1 if index detected
4	Current state of index input, 0 if low or 1 if high

As a result of this command the internal index flag will be cleared.

DEBUG — Undocumented

Command: 0x8001
Inputs: Undocumented
Outputs: Undocumented
Function: This function is provided for service and developing purposes therefore a description of the usage is beyond the scope of this document. For normal operation, do not use this function.

GETALLCNTR — Get the all counters

Command: 0x0700
Inputs: None
Outputs: list of counter values

Function: Get counter value of all counters. If the module is configured for index mode four counters are read, if the module is configured for non-index mode six counters are read. After completion of the command output parameters 0 and 1 contain respectively the upper and lower 16-bit of the signed 32-bit counter value for channel 0, parameters 3 and 4 contain the counter value for channel 1 and so on.

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

6. SOFTWARE

This chapter describes the example software which is available for the M323. The example software is available in ANSI-C source code and consists mainly of an M323 function library which provides functions for easy access to the M323 and a demo program which illustrate the usage of the software library.
The M323 library functions are APIS based, physical accesses and interrupt support are handled by APIS, AcQ Platform Independent Interface Software. The next section contains general information on APIS, for detailed information please refer to the APIS' Programmer's Manual.

6.1. APIS SUPPORT

AcQ produces and supports a large number of standard M-modules varying from networking and process I/O to motion control applications. Physically, the M-modules are supported by a large number of hardware platforms: VMEbus, PCI, CompactPCI as well as a wide variety of operating systems: OS-9, Windows NT, Linux etc.
APIS offers a way to program platform independent applications, example- and test software for controlling hardware. Application software written for APIS only needs re-compiling for a particular platform and is operational with little effort (provided that the application is operating system independent).

6.1.1. CONCEPT

Hardware accesses to registers and memory are handled by APIS. Some minor operating system dependent functions frequently used in hardware related software, such as interrupts handling and a delay function, are also provided by APIS.
APIS platform support consists of an Application Programming Interface in the form of definition files coded in ANSI-C and platform dependent modules, e.g. source files, libraries and/or drivers.
In the most simple outline, a platform dependent APIS module consist of nothing more then macro definitions in which APIS calls are substituted by direct hardware accesses. But in most cases an APIS module will consist of a library with interface routines and in some implementations a device driver is needed for interaction with the operating system.

6.1.2. API

The Application Programming Interface for APIS is implemented in two ANSI-C coded definition files: *apis.h* which contains general definitions and *platform_apis.h* which contains platform specific definitions and references to the APIS function calls.
The application source file must include the APIS header file *apis.h*. Porting of the application to a platform, consists of re-compiling the source code with a defined pre-processor macro for selection of the used platform. The APIS header file contains generic APIS definitions and includes a platform specific header file according to the platform selection macro.
APIS calls are translated to the platform specific calls in the APIS header file and the platform specific definition file *platform_apis.h* (*platform* is a name that identifies a hardware and operating system combination, e.g. i4000os9).
The macro PLATFORM must be defined, either via a pre-processor definition provided at compile time or via a macro-definition in the application source.

6.1.3. CODE GENERATION

APIS based example software is available in ANSI-C source code. Source code files must be compiled with the pre-processor definition PLATFORM set to a valid value, conforming the target platform. Building of the example software for the M323 is platform dependent, for details refer to the release notes of the APIS support package of the target platform and the APIS Programmer's Manual.
Examples of APIS supported platforms are i4000os9, i2000dos, i3000win etc.

6.2. TYPE DEFINITIONS AND STRUCTURES

The table below contains a list and description of all types and structures used in the M323 example software (standard ANSI-C types are not listed).

Name	Type	Description
INT8 UINT8 INT16 UINT16 INT32 UINT32	char unsigned char short unsigned short long unsigned long	8-bit signed data 8-bit unsigned data 16-bit signed data 16-bit unsigned data 32-bit signed data 32-bit unsigned data
PHA8 PHA16 PHA32	volatile unsigned char * volatile unsigned short * volatile unsigned long *	8-bit physical access 16-bit physical access 32-bit physical access
APIS_PATH APIS_HANDLE APIS_WIDTH	unsigned long void * int	APIS physical path ID APIS physical path handle APIS access size in bytes

IDCODE	union {
	  struct {
	  short synccode,
	  modnum,
	  revision,
	  modchar,
	  res[4];
	  char manstr[16];
	  } id;
	
	  short reg[16];
	}
		ID EEPROM contents
		Consisting of either:
		  Sync code (0x5346)
		  Module number (binary coded)
		  Revision number
		  Module characteristics
		  Reserved
		  Manufacturer string
		  ID data
		Or:
		  Raw data

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

Name	Type	Description

M323_DEV	struct {
	APIS_HANDLE apis_handle;
	volatile UINT16 irqstat;
	
	}
		M323 Device, which consists of:
		APIS handle
		Global interrupt status

M323_OPEN_T	enum mode {
	index,
	no_index
	}
		Function argument for m323_open()
		Open M323 in index-mode
		Open M323 in no-index-mode

M323_WRITE_T	struct {
	UINT16 channel;
	UINT32 counter;
	}
		Argument for m323 write()
		Channel number, 0 to 5
		Counter preset value

M323_READ_T	struct {
	UINT16 channel;
	UINT32 counter;
	}
		Argument for m323_read()
		Channel number, 0 to 5 (input)
		Counter value (output)

CMDSTRUCT *	typedef struct {
	UINT16 cmd;
	UINT16 result;
	UINT16 prm[32];
	}
		Command template for m323_ioctl()
		command field
		result field
		parameter field

* For each available command a command structure is defined according to the command template CMDSTRUCT. For an overview and details on the usage refer to the source file *m323lib.h*		

6.3. LIBRARY FUNCTION RETURN VALUES

Each library function returns a result which is either an APIS error code or an M323 library error code. APIS results have bit #15 cleared, library results have bit #15 set. For an overview of APIS error codes refer to the APIS programmer's manual, for an overview of the library result codes refer to the table below:

Result	Hex	Description
NOERR	0x0000	No error
UNKCMD	0x8001	Unknown command
PRANGE	0x8002	Parameter out of range
EXCPERR	0x8003	local exception error
INVREQ	0x8008	Invalid request
ERR_MEM	0x8100	System memory error
ERR_TIMEOUT	0x8200	Command timeout
ERR_IDEEP	0x8300	ID eeprom error
ERR_MODID	0x8400	Wrong module ID

6.4. FUNCTION REFERENCE

This section contains the reference of the functions provided by the M323 APIS-based software library.

m323_open() — Open M323 device

Syntax: `int m323_open(APIS_PATH path_id, M323_DEV *device, M323_OPEN_T *mode)`

Description: Open a hardware path to an M323, check APIS version compatibility, verify module ID by reading the identification EEPROM and initialize the M323. The function initializes the provided device structure of type M323_DEV. The programmer must make sure that enough space is available for the device structure. The module's transmitter and receiver channels will be initialized according to the mode argument. If the module is configured for booting from RAM firmware will be downloaded to the dual ported memory of the module, for detail refer to section 4.7.2.

Arguments:

APIS_PATH path_id
Module APIS path id

M323_DEV *device
Pointer to M323 device

M323_OPEN_T *mode
Operating mode: index or no_index

Returns: APIS error code or M323 library error code

m323_close() — Close M323 device

Syntax: `int m323_close(M323_DEV *device)`

Description: Close an M323 device, consequently a reset of the M323 will be performed and allocated memory is freed.

Arguments:

M323_DEV *device
Pointer to M323 device

Returns: APIS error code or M323 library error code

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

m323_read() — Get counter value

Syntax: `int m323_read (M323_DEV *device, M323_READ_T *encoder, UINT32 *count)`

Description: This function is provided to get the counter value of a specified channel through the GETCNTR command.

Arguments: M323_DEV *device
Pointer to M323 device.

M323_READ_T *encoder
Encoder data, consisting of the input parameter: channel (0 to 5) and the output parameter: counter value.

UINT32 *count
Not used, must be provided for compatibility reasons

Returns: APIS error code or M323 library error code

m323_write() — Write counter preset value

Syntax: `int m323_write (M323_DEV *device, M323_WRITE_T *msg, UINT32 *count)`

Description: This function can be used to write the preset value of a channel. When ‘preset on index’ is disabled the counter will be loaded immediately, if ‘preset on index’ is enabled with the CONFIG command then the counter will be loaded with the preset value when an index pulse is detected.

Arguments: M323_DEV *device
Pointer to M323 device.

M323_WRITE_T *encoder
Encoder data, consisting of the parameter: channel (0 to 5) and the parameter: preset value

UINT32 *count
Not used, must be provided for compatibility reasons

Returns: APIS error code or M323 library error code

m323_ioctl() — Control M323 or Library functions

Syntax: `int m323_ioctl (M323_DEV *device, void *command, int size)`

Description: This function is provided to pass raw commands to the M323 firmware and provides access to miscellaneous control functions provided by the M323 library. Each M323 firmware command is available through the m323_ioctl() call and the library function CLRIRQSTAT is provided for clearing the interrupt status. For details on command usage refer to chapter 5.

Arguments: M323_DEV *device

Pointer to M323 device.

void *command

The file *m323lib.h* contains type definitions of structures for each available command according to the command template show below:

```
typedef struct {          /* Command template */
   UINT16 cmd;            /* command field */
   UINT16 result;         /* result field */
   UINT16 prm[32];        /* parameter field */
} CMDSTRUCT;
```

The command structure is compatible with the host interface and firmware commands described in chapter 5. All M323 firmware commands are accessible through the m323_ioctl() function. Furthermore the M323 library command CLRIRQSTAT is implemented, this command is provided to clear interrupt status bits of the M323 device variable: device->irqstat according to the mask passed via parameter 0. Any bits set in the interrupt mask (parameter 0) are cleared in the module's global interrupt status (device->irqstat). The CLRIRQSTAT function is running as critical code with interrupts disabled. Additionally the definition file *m323lib.h* contains type definitions for each available command, compatible with the command structure template, however elements are referenced by name and the size is limited to the number of parameters actually used.

int size

Size of the provided command structure in bytes. Note that the size must be at least equal to the size of the command field, result field and the number of parameters, input or output whichever are the most. For an optimal size use the sizeof() ANSI-C macro.

Returns: APIS error code or M323 library error code

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

m323_irqh() — M323 Interrupt Service Routine

Syntax: `int m323_irqh (APIS_HANDLE apisHandle, M323_DEV *device)`

Description: The interrupt handler will clear the interrupt request of the M323. The module's interrupt status will be saved in device->irqstat.

Arguments: APIS_HANDLE apisHandle
APIS handle, the APIS handle can be obtained from the M323 device handle: device->apis_handle

M323_DEV *device
Pointer to M323 device

Returns: If the interrupt was caused by the module, 0 is returned otherwise -1 is returned.

6.5. SOFTWARE DISTRIBUTION

This section gives an overview of the software distribution.

File	Description
M323\SOFTWARE\m323rel.txt	Release notes
M323\SOFTWARE\LIB\m323lib.c	ANSI-C M323 APIS software library
M323\SOFTWARE\LIB\m323lib.h	Definitions for the M323 library
M323\SOFTWARE\LIB\m323defs.h	Definitions for the M323
M323\SOFTWARE\FIRM\m323firm.c	Downloadable firmware image for the M323
M323\SOFTWARE\EXAMPLE\m323demo.c	Demo program for the M323
M323\SOFTWARE\EXAMPLE\m323irqd.c	Interrupt driven demo program for the M323
M323\SOFTWARE\EXAMPLE\makefile.bor	Borland C makefile for M323 on i2000
M323\SOFTWARE\EXAMPLE\makefile.os9	OS9 makefile for M323 on i4000
MMODID\SOFTWARE\LIB\modideep.c	ANSI-C ID EEPROM APIS software library
MMODID\SOFTWARE\LIB\ideeprom.h	Definitions for the ID EEPROM library

M323 example software is APIS based, therefore APIS support for the target platform is required for code generation.

The following figure is an example of the M323 software integrated in the APIS environment on an i4000/OS-9 target platform.

```
+---PROJECT                          AcQ's distribution
     +---APIS                        APIS basis distribution
     |    +---DOC                    APIS documentation
     |    +---SOFTWARE
     |         | readme.txt          Distribution overview
     |         +---COMMON
     |         |    +---DEFS
     |         |    |    apis.h      General definitions
     |         |    +---OS9TRAP
     |         |    |   +--CMDS
     |         |    |       apistrap  OS-9 trap handler
     |         |    +---....
     |         +---I4000OS9          i4000/OS-9 support
     |         |    relnotes.txt     Release notes / version info
     |         |    apis_i4000os9.h  Platform specific definitions
     |         |    apis_i4000os9.c  Platform support library
     |         +---....
     |
     +---M323                        M323 distribution
     |    +---SOFTWARE
     |         | m323rel.txt         Release notes / version info
     |         +---LIB               M323 support libraries
     |         +---EXAMPLE           M323 example software
     +---MMODID
          +---SOFTWARE
               +---LIB
                    modideep.c      M-module ID EEPROM library
                    ideeprom.h      M-module ID definitions
```

Code generation is platform dependent, for information on building the software please refer to the release notes of the target platform and the APIS Programmer's Manual.

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands

7. ANNEX

7.1. BIBLIOGRAPHY

Specification for M-module interface and physical dimensions:
M-module specification manual, April 1996, MUMM.
Simon-Schöffel-Strasse 21, D-90427 Nürnberg, Germany.

APIS Programmer's Manual
AcQuisition Technology
P.O. Box 627, 5340 AP Oss, The Netherlands.

Data Sheet of the NM93C46
Memory Databook 1992 edition (400069)
National Semiconductor Corporation
1111 West Bardin Road; Arlington, TX76017, United States

7.2. COMPONENT IMAGE

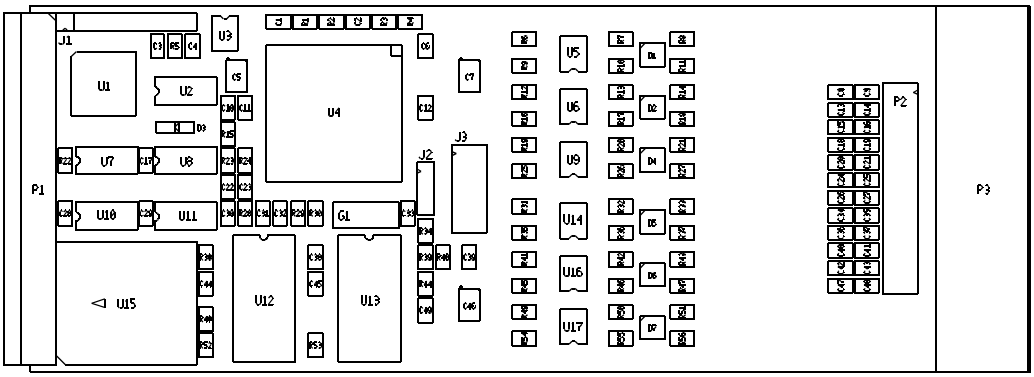


Figure 12 M323 Component Image

7.3. TECHNICAL DATA

Slots on the base-board:
Requires one 16-bit M-module slot.

Connection:
To base-board via 40 pole M-module interface.
To peripheral via 25 pole D-sub connector.

Power supply:
+5VDC ±10%, typical 125mA.

Temperature range:
Operating: 0..+60°C.
Storage : -20..+70°C.

Humidity:
Class F, non-condensing.

7.4. DOCUMENT HISTORY

- Version R1.0
  First release
- Version R1.1
  Function GETALLCNTR added to firmware command set.

AcQuisition Technology bv
P.O. Box 627, 5340 AP
Oss, The Netherlands