AcQuisition Technology bv

Headquarters:
Rijnstraat 20
5347 KN 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

M338

Octal Relay Controller 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.

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

CONTENTS

Intentionally left blank.

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

1. INTRODUCTION

1.1. VALIDITY OF THE MANUAL

This document is of version 1.1 and provides information on the use of the M338 relay M-module.

1.2. PURPOSE

This manual serves as an instruction for the M338 relay M-module. The connection of peripherals is described. Furthermore it gives the user additional information for special applications and configuration of the product.

1.3. SCOPE

The scope of this manual is the M338 relay M-module.

1.4. DEFINITIONS, ACRONYMS AND ABBREVIATIONS

AcQ	AcQuisition Technology
APIS	AcQ's Platform Interface Software
ESD	Electronic Static Discharge
M-module	Mezzanine module

1.5. NOTES CONCERNING THE NOMENCLATURE

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

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

Code examples are 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.

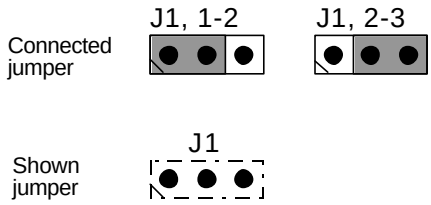


Figure 1 Example jumper nomenclature

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

In chapter 2 a short description of the M338 hardware can be found. Chapter 3 covers the installation and setup of the board as well as the connection of the peripherals. In chapter 4 the operation and the usage of the M338 is described in detail. AcQ provides APIS based example software for the M338, which is described in chapter 5. Finally this document contains an annex consisting of a bibliography, component image, technical data and a document history.

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

2. PRODUCT OVERVIEW

2.1. INTRODUCTION

The M338 Octal Relay Controller M-module is designed as a plug-on module for the M-module interface. The M338 is equipped with eight monostable relays, with 2 throw over contacts each. One throw over contact is available at the front connector, while the other contact is used for reading back the relay state. The maximum continuous current for the relays is 1A. The maximum switching voltage is 85 V_{ac}/100 V_{dc}. For more information, refer to section 6.3.

The M338 features an emergency break input that must be activated for normal operation. If not, all relays are switched off to the normally closed position and can not be driven.

2.2. TECHNICAL OVERVIEW

Below the features of the M338 are listed.

Features:
- High-current relays (up to 6A). Connector specifications limits the output current to 1 A.
- 100 V nominal relay voltage
- Optical isolated input for hardware reset of relays, sink type, -5 .. +36 V_{dc} input voltage
- 3 programmable counters/timers

Controller:
- Z8536 CIO Parallel I/O controller
- Change-of-state interrupts
- Pattern Match interrupts

Relays:
- Normally-open contacts
- State-of-the-art mechanical PCB relays with excellent switching times (10 ms on, 5 ms off)
- Switch up to 1 A
- 85 V_{ac}/100 V_{dc} maximum switching voltage
- Emergency shutdown input removes power from all relays instantaneously without software interaction.
- True read-back using secondary switch

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 table, 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, 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 static dissipative.)

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. JUMPER SETTINGS

The M338 can operate in two modes. Choose the Break mode when all relays must be switched off in emergency cases. If not, select the Compatibility mode. This sub paragraph will describe the jumper settings for the different modes.

3.2.1. BREAK MODE

In this mode the break inputs are operational. Only seven relays are available (0-6).

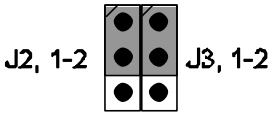


Figure 2 Break mode

This mode has two sub modes: isolated and non-isolated. If non-isolated mode is selected, the break output (pin BO) must be connected to the M-module ground potential (pin GND) via a normally closed safety switch. If isolated mode is selected, the break input is not related to the M-module ground potential. The break input and output must be tied to an external source. The break input voltage must be at least 12 Volt higher than the break output. If no external source is available, the non-isolated mode must be used. Any input voltage higher than 15V requires an extra series resistor. The value can be calculated with:

$$R_{SER} = ((U_{BREAK\ INPUT} - U_{BREAK\ OUTPUT}) - 12V) / 50mA.$$

Figure 3 shows the jumper position for both modes.

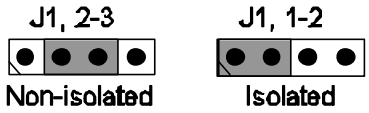


Figure 3 Non-isolated/isolated mode

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

3.2.2. COMPATIBILITY MODE

In this mode all eight relays are available and always enabled. The break relay is not used.

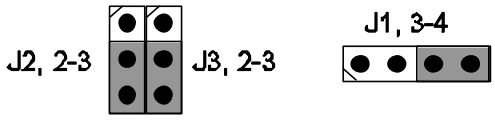


Figure 4 Compatibility mode

3.3. CONNECTING THE MODULE

This section gives an overview of the I/O pins of the M338 which are available on the female sub-D connector at the front of the M-module and the 24 pole female header on the component side of the M338.

The M338 can be configured for two different modes. In break mode all relays can be switched off by the absence of an external signal. In this case only seven relays are available (0 to 6). The external pins of the eighth relay are used for the break input and output. In compatibility mode all eight relays are available. The break inputs are replaced by the external pins of the eighth relay.

Figure 5 and 6 show the pin assignment for break mode.

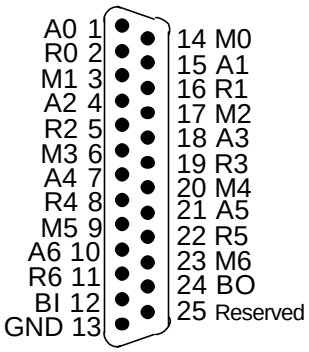


Figure 5 Front view 25 pole female sub-D

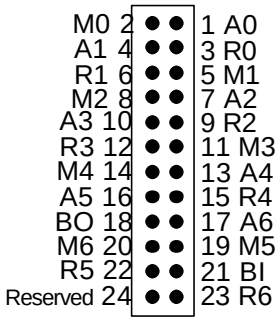


Figure 6 Top view 24 pole female header

Signal name	Description
Ax	Operating contact relay x
Mx	Middle (common) contact relay x
Rx	Resting contact relay x
BI	Break input
BO	Break output
GND	Ground

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

Figure 7 and 8 show the pin assignment for compatibility mode.

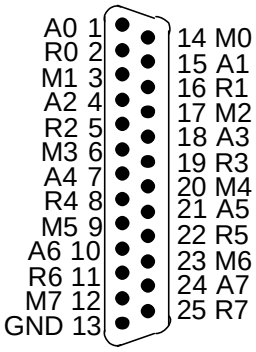


Figure 7 Front view 25 pole female sub-D

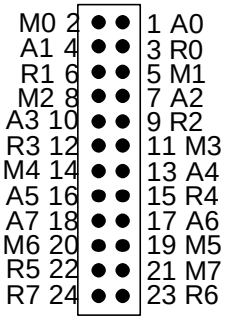


Figure 8 Top view 24 pole female header

Signal name	Description
Ax	Operating contact relay x
Mx	Middle (common) contact relay x
Rx	Resting contact relay x
GND	Ground

Intentionally left blank.

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

4. FUNCTIONAL DESCRIPTION

The M338 Relay output M-module has an M-module interface (A08/D08), a Z8536 CIO parallel I/O controller, eight relay outputs, one control relay, a read back facility with contact debounce on all eight output relays, interrupt generation on a change of state of the read back switches or break input, an identification EEPROM and an external reset input.

The read back switches can be used to compare the actual relay status to the programmed state. The break input must be activated for normal operation. If not, all output relays are powered off and return to the normally closed state.

4.1. BLOCK DIAGRAM

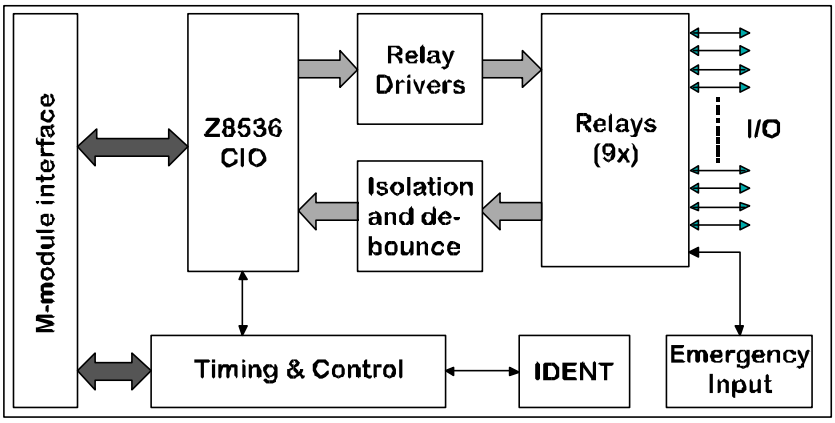


Figure 9 M338 block diagram

4.2. M-MODULE INTERFACE

The M338 has an A08/D16 INTA compliant M-module interface. The following sections give an overview of the M-module interface.

4.2.1. REGISTER MAP

The following table shows the register map of the M338. All addresses are relative to the base address of the M-module.

Offset	Width	Name	Description
0x0e	16-bit	CIOCTRL	Z8536CIO control register
0xfe	16-bit	EEPREG	EEPROM control register

The following sections give a detailed description of the registers.

4.2.2. Z8536CIO CONTROL REGISTER

This register is used to access the internal registers of the Z8536CIO chip. Each internal register can be accessed with the following two-step sequence: First, write the address of the target register to the Z8536CIO control register then read the same control register to get the status of the target register or write to change the target register. Each read operation on the Z8536CIO control register that follows the two-step sequence is executed on the same target register when there is a write operation after the two-step sequence the contents of the target register will be modified.

To put the Z8536CIO chip in reset, set bit 0 of the Master Interrupt Control register of the chip. The reset can be cleared by clearing bit 0 of the Z8536CIO control register.

CIOCTRL Offset : 0x0E

15 … 8	7	6	5	4	3	2	1	0
Not used	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0

Read / Write

Refer to the Z8536CIO data sheet for more information about the internal registers of the Z8536CIO chip.

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

4.2.3. 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:

EEPREG Offset: 0xFE

15 ... 4	3	2	1	0
not used	unused	CS	CLK	DI/O
Reset:	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. For information on the memory organization refer to the M-module Specification.

4.3. RELAYS

The relays used on the M338 have throw over contacts. The figure below shows such throw over contact in Normally Closed state (NC).

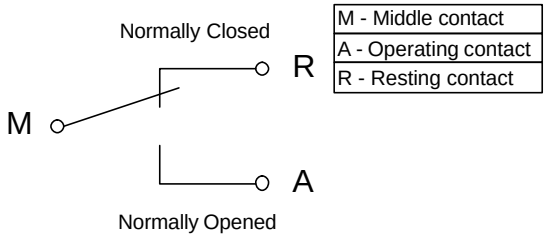


Figure 10 Throw over contact in NC state

After reset, the relays are always in the Normally Closed state. Activating the relay will put it in the Normally Opened state. The Normally Opened and Normally Closed state are referred to as the NO respectively NC state.

4.3.1. WRITING RELAY DATA

The relays are driven by a relay driver which is controlled by port B of the Z8536CIO controller. To write relay data, the Z8536CIO controller must be reset and initialized.

After initialisation, the relays can be driven by writing a byte to the port B data register. This byte represents the relays which must be activated. Setting a '1' to any of the port B data bits will activate the coil of the corresponding relay (0..7) and set the switch to the NO state. Writing a '0' to any of the port B data bits will release the switch to the NC state.

Port B data register

7	6	5	4	3	2	1	0
R7	R6	R5	R4	R3	R2	R1	R0
Reset:							
0	0	0	0	0	0	0	0
Read/write							

Reading this register will return the last data written to it. It does **not** reflect the state of the read back switches or the relay coils. This read back data may differ from the data in this register when either the break input is not activated or the relay driver or one or more of the relays are malfunctioning. It can be used for read-modify-write operations.

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

4.3.2. READING RELAY READ BACK SWITCH DATA

The read back switches of the output relays are available at port A of the Z8536CIO controller. To read the state of the read back switches, the Z8536CIO controller must be reset and initialized.

Port A data register

7	6	5	4	3	2	1	0
R7	R6	R5	R4	R3	R2	R1	R0
Reset:							
0	0	0	0	0	0	0	0
Read only							

When a bit is read as ‘1', the read back switch is activated which means that the relay is in NO state. When a bit is read as ‘0', the read back switch is not activated which means that the relay is in NC state.

4.3.3. BREAK INPUT

The M338 features a break input to enable or disable all relays. When activated, all relays are enabled. For normal operation it must be activated. The break relay read back switch is available at Port C of the Z8536CIO. To read the state of the break input, the Z8536CIO controller must be reset and initialized.

Port C data register

7	6	5	4	3	2	1	0
x	x	x	x	x	x	x	BI
Reset:							
0	0	0	0	0	0	0	0
Read only							

When bit 0 of the port C data register is set to ‘1', the read back switch of the break relay is in the NO which means that the break input is activated and all relays are enabled.
When bit 0 of the port C data register is set to ‘0', the read back switch of the break relay is in the NC which means that the break input is not activated and all relays are disabled.

Figure 11 shows how the break inputs enable or disable the output relays by interrupting the power supply of the coil.

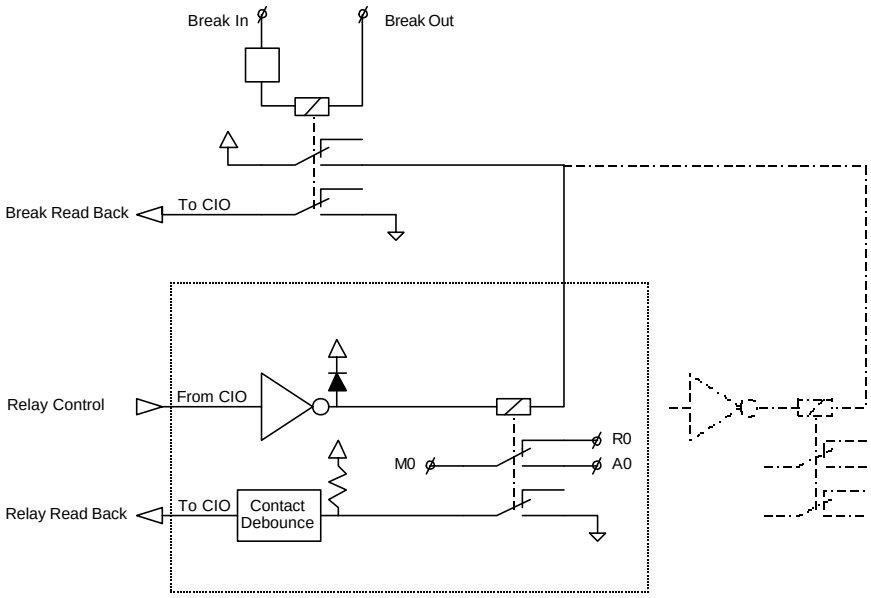


Figure 11 Basic schematics for one relay

4.4. INTERRUPT HANDLING

The M338 can be programmed to activate an interrupt request line to the base board. Interrupts can be generated on changing of the interrupt pattern or by recognition of a certain pattern.

Detailed information about configuring interrupts can be found in the Z8536CIO data sheet.

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

5. SOFTWARE

This chapter describes the example software which is available for the M338 Octal Relay M-module. The example software is available in ANSI-C source code and consists mainly of an M33X function library which provides functions for easy access to the M338 and a demo program which illustrates the usage of the software library.

The M33X 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.

5.1. 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).

5.1.1. 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 interrupt 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.

5.1.2. 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.

5.1.3. 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 M338 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.

5.2. TYPE DEFINITIONS AND STRUCTURES

The table below contains a list and description of all types and structures used in the M338 example software (standard ANSI-C types are not listed).

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

M33X_HANDLE	struct {
	  APIS_HANDLE apis_handle;
	  UINT8 irqstat;
	}
		M33X handle
		APIS handle
		IRQ status

IDCODE	union {
	  struct {
	  short synccode,
	  modnum,
	  revision,
	  modchar,
	  res[4];
	  char manstr[16];
	  } id;
	  short reg[16];
	}
		ID EEPROM contents
		
		Sync code (0x5346)
		Module number (binary coded)
		revision number
		Module characteristics
		reserved
		manufacturer string
		ID data
		raw data

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

5.3. FUNCTION REFERENCE

m33x_open() — Open path to M338

Syntax: `int m33x_open(APIS_PATH path_id, M33X_HANDLE *m33x_handle)`

Description: Opens a hardware path to M338.

Arguments: APIS_PATH path_id
Module APIS path ID
M33X_HANDLE *m33x_handle
Module handle, passed to the caller

Returns: APIS_NOERR or APIS error code

Example: `m33x_open(path_id, &m33x_handle);`

m33x_close() — Close path to M338

Syntax: `int m33x_close(M33X_HANDLE *m33x_handle)`

Description: Closes hardware path to M338.

Arguments: M33X_HANDLE *m33x_handle
Module handle returned by m33x_open()

Returns: APIS_NOERR or APIS error code

Example: `m33x_close(&m33x_handle);`

m33x_get_apis_version() — Get APIS version

Syntax: `int m33x_get_apis_version(char *verstr)`

Description: Get APIS version ID.

Arguments: char *verstr
Pointer to APIS version ID

Returns: 0 on success or 1 if APIS version is invalid

Example: `m33x_get_apis_version(verstr);`

m33x_int_install() — Install interrupt routine

Syntax: `int m33x_int_install(M33X_HANDLE *m33x_handle)`

Description: Installs the interrupt service routine m33x_irqh().

Arguments: M33X_HANDLE *m33x_handle
Module handle returned by m33x_open()

Returns: APIS_NOERR or APIS error code

Example:
```
m33x_int_install(&m33x_handle);
```

m33x_int_deinstall() — De-install interrupt routine

Syntax: `int m33x_int_deinstall(M33X_HANDLE *m33x_handle)`

Description: De-installs the interrupt service routine m33x_irqh().

Arguments: M33X_HANDLE *m33x_handle
Module handle returned by m33x_open()

Returns: APIS_NOERR or APIS error code

Example:
```
m33x_int_deinstall(&m33x_handle);
```

m33x_waitforirq() — Wait for interrupt

Syntax: `int m33x_waitforirq(void)`

Description: Wait for an interrupt.

Arguments: None

Returns: APIS_NOERR when an APIS signal is received.
APIS_ESIG when another signal is received.

Example:
```
m33x_waitforirq();
```


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

m33x_clear_irqstat() — Clear interrupt status

Syntax: `void m33x_clear_irqstat(M33X_HANDLE *m33x_handle)`

Description: Clear IRQ status field.

Arguments: M33X_HANDLE *m33x_handle
Module handle returned by m33x_open()

Returns: APIS_NOERR or APIS error code

Example:
```
m33x_clear_irqstat(&m33x_handle);
```

m33x_delay() — Insert delay

Syntax: `void m33x_delay(UINT32 delay)`

Description: Suspend process for a requested delay. The delay is provided in msecs but the minimum delay is one system tick (mostly 10 msecs).

Arguments: UINT32 delay
Delay in msecs.

Returns: Nothing

Example:
```
m33x_delay(1000);
```

m33x_getid() — Get module information

Syntax: `int m33x_getid(M33X_HANDLE *m33x_handle, void *buffer)`

Description: This function reads the contents of the ID EEPROM on an M-module. For the structure used to identify M-modules, IDCODE, please refer to page ?.

Arguments: M33X_HANDLE
Module handle returned by m33x_open()
void *buffer
Pointer to data buffer

Returns: APIS_NOERR or APIS error code

Example:
```
m33x_getid(&m33x_handle, &eepbuf);
```

m33x_cio_wr() — Write Z8536CIO register

Syntax: `int m33x_cio_wr(M33X_HANDLE *m33x_handle, UINT8 reg, UINT8 data)`

Description: This function is used to write data to the specified Z8536CIO register. This operation will be performed with interrupts masked.

Definitions for internal Z8536CIO registers and bit definitions can be found in *m3xxdefs.h.*

Arguments:
M33X_HANDLE
Module handle returned by m33x_open()
UINT8 reg
Z8536CIO register
UINT8 data
Data to write

Returns: APIS_NOERR or APIS error code

Example: `m33x_cio_wr(&m33x_handle, PA_DATA, ALL_INP);`

m33x_cio_rd() — Read Z8536CIO register

Syntax: `int m33x_cio_rd(M33X_HANDLE *m33x_handle, UINT8 reg, UINT8 *data)`

Description: This function is used to read data from the specified Z8536CIO register. This operation will be performed with interrupts masked.

Definitions for internal Z8536CIO registers and bit definitions can be found in *m3xxdefs.h.*

Arguments:
M33X_HANDLE
Module handle returned by m33x_open()
UINT8 reg
Z8536CIO register
UINT8 *data
Pointer to location to store data

Returns: APIS_NOERR or APIS error code

Example: `m33x_cio_wr(&m33x_handle, PB_DATA, &data);`

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

m33x_cio_reset() — Reset Z8536CIO

Syntax: `int m33x_cio_reset(M33X_HANDLE *m33x_handle)`

Description: This function is used to reset the Z8536CIO. This operation will be performed with interrupts masked.

Arguments: M33X_HANDLE
Module handle returned by m33x_open()

Returns: APIS_NOERR or APIS error code

Example:
```
m33x_cio_reset(&m33x_handle);
```

5.4. SOFTWARE DISTRIBUTION

This section gives an overview of the software distribution.

File	Description
M33X/SOFTWARE/m33xrel.txt	Release notes
M33X/SOFTWARE/LIB/m33xdefs.h	Definitions for M33X software library
M33X/SOFTWARE/LIB/m33xlib.c	APIS based ANSI-C M33X software library
M33X/SOFTWARE/EXAMPLE/m330_m331demo.c	M330/M331 demo program
M33X/SOFTWARE/EXAMPLE/m332_m333demo.c	M332/M333 demo program
M33X/SOFTWARE/EXAMPLE/m338demo.c	M338 demo program
M33X/SOFTWARE/EXAMPLE/makefile.bor	Example make file for Borland C
M33X/SOFTWARE/EXAMPLE/makefile.lin	Example make file for Linux
M33X/SOFTWARE/EXAMPLE/makefile.os9	Example make file for OS-9
MMODID/SOFTWARE/LIB/ideeprom.h	Definitions for the ID EEPROM library
MMODID/SOFTWARE/LIB/modideep.c	ANSI-C ID EEPROM APIS software library

M338 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 M338 software integrated in the APIS environment with as target platform the i4000/OS-9.

```
+---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
     |         +---....
     +---M33X                        M33X distribution
     |    +---DOC                    M33X documentation
     |    +---SOFTWARE
     |         | m33xrel.txt         Release notes / version info
     |         +---LIB               M33X support libraries
     |         |    m33xdefs.h
     |         |    m33xlib.c
     |         +---EXAMPLE           M33X example software
     |              m330_m331demo.c  M330/M331 demo software
     |              m332_m333demo.c  M332/M333 demo software
     |              m338demo.c       M338 demo software
     +---MMODID
          +---SOFTWARE
               +---LIB
                    modideep.c      M-module ID EEPROM library
                    ideeprom.h      M-module ID definitions
```


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

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.

6. ANNEX

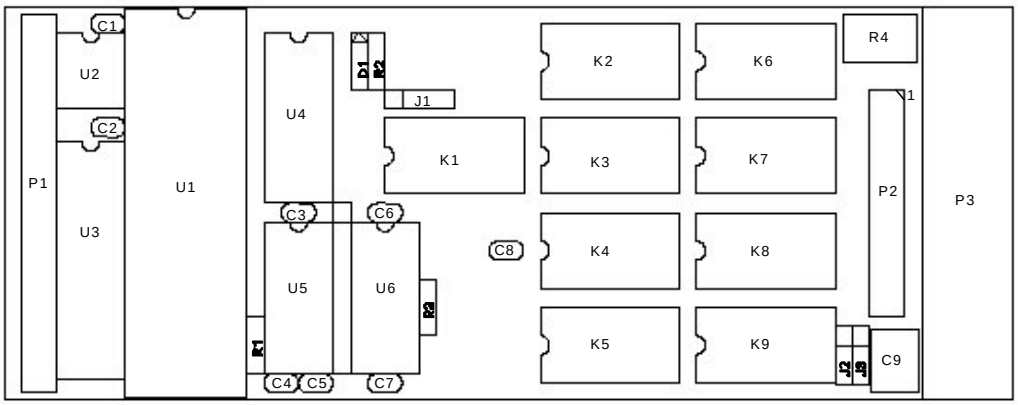
6.1. BIBLIOGRAPHY

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.
Downloadable from www.acq.nl

Z8536-CIO Technical Manual
Z8536 CIO data sheet by Zilog

6.2. COMPONENT IMAGE



6.3. 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.

Relays:
V23042-A2001-B101
Approvals: UL, CSA
Relays contact ratings:
Max. switching voltage: 100 V_{dc} / 85 V_{ac}
Max. continuous current: 1 A
Contact material:
Stationary contacts: silver palladium, gold-plated
Movable contacts: silver palladium

Power supply:
+5VDC ±10%, typical 600mA.

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

Humidity:
Class F, non-condensing.

6.4. DOCUMENT HISTORY

! Version 1.0
First release
Replacement for M338 Hardware Manual and M338 Software Manual
! Version 1.1
Technical data changed