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

M395

12/16-bit DAC M-module with 16x Voltage Outputs

User Manual

Version 1.3

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

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

1. INTRODUCTION

1.1. VALIDITY OF THE MANUAL

This document version 1.2 provides information on the use of the M395/R0.x running firmware of revision 1.4 and up.

The M-module revision number is composed as Rx.y where x is the revision of the PCB and y is the revision of the PLD firmware.

The firmware version number consists of a major and a minor number. Firmware versions with the same major number are compatible. The minor number will be incremented for firmware updates that do not affect the usage of the module.

1.2. PURPOSE

This manual serves as instruction for the operation of the M395 Analog Output M-module, the connection of peripherals and the integration on a M-module carrier. Furthermore it gives the user additional information for special applications and configuration of the product.

Detailed information concerning the individual assemblies (data sheets etc.) Are not part of this manual. This manual also contains a description of the provided example software.

1.3. SCOPE

The scope of this manual is the usage of the M395 DAC M-module with 16x voltage outputs.

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
PWM	Pulse Width Modulation

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 example are printed in `courier`.

1.6. OVERVIEW

In chapter two a description of the M395 hardware can be found. The next chapter covers the installation and setup of the module as well as the connection of the peripherals. In chapter 4 the operation and the usage of the M395 is described in detail. AcQ provides APIS based example software for the M395, the software is described in chapter 5. Finally this document contains an Annex containing a bibliography, component image, technical data and the document history.

2. PRODUCT OVERVIEW

2.1. INTRODUCTION

The M395 is a 16-channel analog output M-module with voltage outputs and 12-bit or 16-bit resolution, suitable for process control applications. The M395 has 16 D-to-A converters with simultaneous output update capability. The resolution of the D-to-A converters is 12-bit, the output resolution can be configured to 16-bit. In 16-bit mode the M395 uses pulse width modulation to achieve the additional 4-bit resolution. A local TMS320C203 DSP provides amongst other functions the range selection and output correction. The operation of the local DSP is transparent for the user. Each DAC is accessed by a single 16-bit wide write to dual ported memory, providing the simplest possible interface for the software. The M395 has two output ranges which are software selectable: +/-10V bipolar and 0...10V unipolar and it has no potentiometers since output data correction is handled by the DSP according to factory calibrated system parameters stored in an onboard EEPROM. The front-end of the M395 is optically isolated, powered from either an on-board isolated DC/DC converter or from an externally applied source.

The M395 is available in the following versions:

- **M395//WDC** with on-board DC/DC converter
- **M395/NDC** no DC/DC converter, external supply required

2.2. TECHNICAL OVERVIEW

Below an overview of the functionality of the M395 is listed.

- 16x 12-bit digital-to-analog converters.
- Optional 16-bit resolution in continuous update mode using a PWM scheme.
- Common-mode analog voltage outputs.
- Accuracy of 0.4%.
- Simultaneous update capability.
- Output response time of typically 10 µseconds.
- No potentiometers.
- Optional on-board isolated power supply for analog front-end.
- With on-board DC/DC converter the module is operated from a single 5V supply.
- Output updating handled by TMS320C203 DSP, transparent for user.
- Output data corrected by the DSP according to calibration data stored in an onboard EEPROM.
- Each output channel is accessible by a single 16-bit write to dual ported memory.
- Module setup programmable through a command interface in dual ported memory.
- User programmable output range:
  - 0..10V Unipolar (default)
  - +/10V Bipolar.
- A08/D16 M-module interface.
- Host interrupts supported.
- Identification EEPROM.

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 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. CONNECTING THE MODULE

This section describes the pin-assignments of the 25-pole D-sub female front connector and the 24-pole female P2 header.

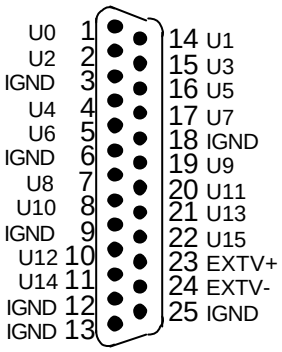


Figure 1 Front view 25 pole D-sub

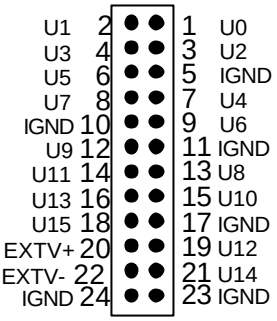


Figure 2 Top view 24 pole female header

The signals labelled with Ux are common-mode voltage outputs where x is the channel number. The IGND pins are connected to the isolated ground of the module. The analog front-end is powered from either an onboard DC/DC converter (M395/WDC) or from an externally applied power supply (M395/NDC) depending on the module version.
If the module has an board DC/DC converter the external power pins EXTV+ and EXTV- must not be connected. If the module has no onboard DC/DC converter an external power supply of +/-15V DC must be applied: +15V DC must be applied to EXTV+ and -15V DC must be applied to EXTV-, the power supply ground must be connected to IGND. For details on the power requirements please refer to section 6.3. For a description of the output circuitry refer to section 4.3.

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

4. FUNCTIONAL DESCRIPTION

This chapter gives a detailed description of the M395. The M-module interface is described as well as the operation.

4.1. BLOCK DIAGRAM

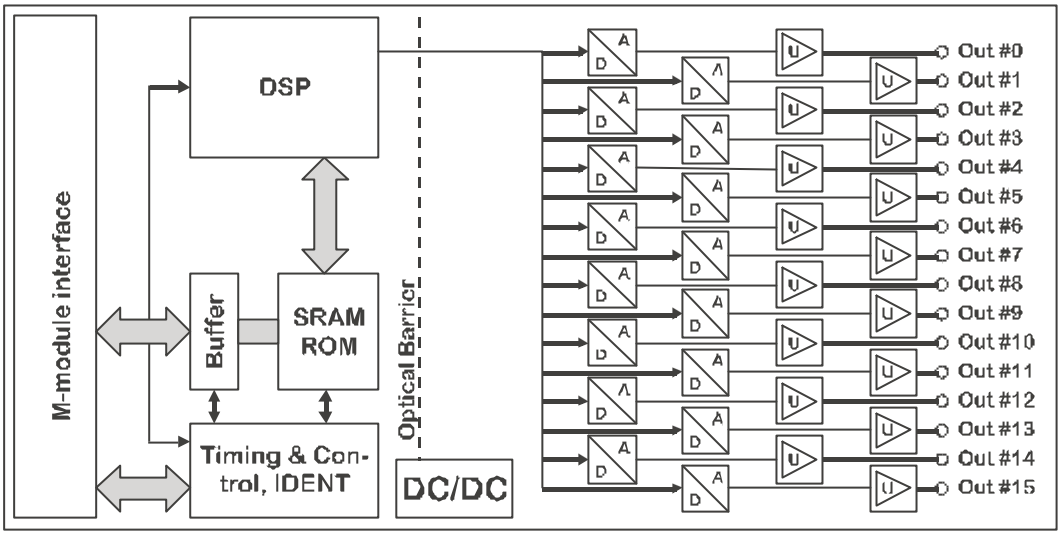


Figure 3 M395 block diagram

4.2. M-MODULE INTERFACE

The M395 has an A08/D16 INT A compliant M-module interface. The following sections give an overview of the M-module interface and memory organization of the M395.

4.2.1. MEMORY MAP OVERVIEW

The following table shows the address map of the M395 module. All addresses are relative to the base address of the module.

Offset	Name	Description
0x00	ODR0	Output data register for channel 0
....	....	....
....	....	....
0x1e	ODR15	Output data register for channel 15
0x20	RES	Reserved
....	....	....
....	....	....
0x3e	RES	Reserved
0x40	C_CMD	Command interface: command field
0x42	C_RES	Command interface: result field
0x44	C_PRM0	Command interface: parameter 0
....	....	....
....	....	....
0x62	C_PRM15	Command interface: parameter 15
0x64	GLOBSTAT	Global status
0x66	RES	Reserved
....	....	....
....	....	....
0x7e	RES	Reserved
0x80	PAGEREG	Page register, must be written with 0x100
0x82	CTRLREG	Host control register
0x84	RES	Reserved
....	....	....
....	....	....
0xfc	RES	Reserved
0xfe	EEPREG	Identification EEPROM register

Caution: Only 16-bit wide accesses are allowed. Do not write reserved fields.

Fields described in the memory map are either located in dual ported memory, offset 0x00 to offset 0x80 (page register set to 0x0100) or are implemented as hardware registers: page register, host control register and the EEPROM register.

The following sections give a detailed description of memory locations and hardware registers.

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

4.2.2. OUTPUT DATA WORDS

Output data must be written to memory locations ODR0 to ODR15 at offset 0x00* to offset 0x20*. The data representation depends on the selected output range, 0..10V unipolar: straight binary format or -10V..+10V bipolar: two's complement format. By default the M395 operates in 12-bit mode, additionally the M395 can be configured for 16-bit resolution mode.
The following table shows the relation between the 12-bit binary codes and the corresponding analog outputs.

Range	Data format	Binary Code **		Output
0..10 V 12-bit (default)	Straight Binary	0x0000 0x0800 0x0fff	0 2048 4095	0 V +5.000 V +9.998 V
+/-10 V 12-bit	Two's Complement	0xf800 0x0000 0x07ff	-2048 0 +2047	-10 V 0 V +9.995 V

The following table shows the relation between the 16-bit binary codes and the corresponding analog outputs.

Range	Data format	Binary Code **		Output
0..10 V 16-bit	Straight Binary	0x0000 0x8000 0xffff	0 32768 65535	0 V +5.000 V +9.999 V
+/-10 V 16-bit	Two's Complement	0x8000 0x0000 0x7fff	-32768 0 +32767	-10 V 0 V +9.999 V

In 12-bit continuous mode (default) a change of the contents of an ODR results in the update of the corresponding output, for details refer to section 4.4. In 16-bit continuous mode the ODRs accept 16-bit data, the upper 12 bits are loaded to the DACs and the lower 4 bits are used to add a PWM component resulting in a 16-bit output resolution.

In on-demand-update mode (12-bit only), requested output data, written to the ODR's must be loaded into the D-to-A converters and the outputs must be updated via host generated commands, for details refer to section 4.2.3.

* The page register must be set to 0x0100
** The MS-Nibble of the output word is discarded.

4.2.3. COMMAND INTERFACE

The M395 features a command interface provided for communication between the host and the local DSP. Through the command interface it is possible to configure special features of the M395, like selecting an alternative update mode, programming ranges etc. For operation in default mode (0..10V outputs, continuous updating) no commands have to be executed.

The command interface consists of a command field, a result field and a parameter field:

Offset *	Name	Description
0x40	command	command field
0x42	result	result field
0x44	parm0	first parameter
….	….	….
….	….	….
0x62	parm15	last parameter

* The page register must be set to 0x0100

The M395 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 DSP, 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 DSP. The DSP is ready to accept new commands.

For a complete overview of the available commands and the usage refer to section 4.8.1.

4.2.4. GLOBAL STATUS WORD

The global status word at offset 0x64 (with the page register set to 0x0100) contains status information of the module. Under normal conditions the global status word is zero. A non-zero status word indicates a possible defect of the module and the output data is unreliable. When after a power-up the problem still exists contact AcQuisition Technology bv for repair.

4.2.5. PAGE REGISTER

The dual ported memory of the M395 is accessed using a page register for the upper address bits. The first 128 bytes of the address space of the M395 M-module is used as a "window" into the dual ported memory as described in the previous section.

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

PAGEREG Offset: 0x80

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

For normal operation the page register must be set to 0x0100. The page register is available for service purposes and must not be set to any other value than 0x0100.

4.2.6. HOST CONTROL REGISTER

CTRLREG Offset: 0x82

15	9	8	7	4	3	2	1	0
not used		BOOT	not used		RDY	RESET	reserved	HIRQ
Reset:		0			0	1	0	0

Read / Write

RDY Ready Bit
This bit must be cleared by the host by writing a logic ‘1'. The bit will be set by the module as soon as it is up and running.

RESET Local reset
When asserted, the M395 will be kept in reset state. After power-up the module is in reset.

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. Currently no interrupts are supported.

BOOT Boot selection
Default the local firmware is booted from ROM, however it is possible to boot the DSP from dual ported memory, therefore the BOOT bit must be set to ‘1' prior to a reset. The boot bit is available for service and debug purposes, so details are beyond the scope of this manual.

CAUTION: Do not use read-modify-write operations on the host control register. (AND/OR instructions).
Reserved bits must be written by a logic ‘0'.

4.2.7. 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
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.

Note: The identification EEPROM on the M395 is only accessible by the host when the module is kept in reset. (Bit #2 in the host control register 'set').

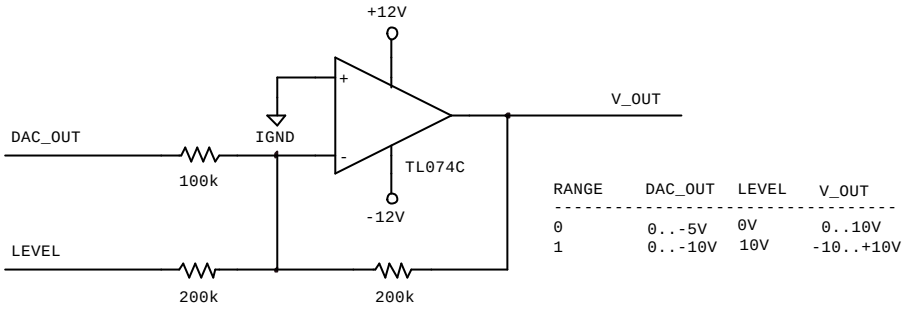
Caution: On the M395 the EEPROM is not only used for identification purposes but also for storage of calibration data used by the local DSP. Therefore writing to or clearing from the EEPROM by the host will result in erroneous behaviour.

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

4.3. OUTPUT CIRCUITRY

This section describes the output circuitry of the M395, this might be useful for connecting actuators.

The picture below shows the simplified schematics of output channel 0 of the M395, channel 1 to channel 15 are similar.



The output circuitry of a channel consists of a level shifter, to obtain the two output ranges the DAC output span and the level voltage for the shifter are software programmable. The range configuration circuitry is omitted in the diagram.
See the table below for the technical specifications of the output circuit:

Maximum output current	2 mA
Slew rate	13 V/µs typical

Detailed information on connecting actuators or other peripherals are beyond the scope of this manual.

4.4. CONTINUOUS OUTPUT UPDATE, 12-BIT

Analog output updating is handled by the onboard TMS320C203 DSP without any user intervention. However this section describes the update process for a better understanding of the operation and usage of the M395.

Default the M395 is configured for continuous output update. In this mode the DSP monitors the dual ported memory (ODR registers, section 4.2.2.). Whenever a requested output value is changed, the DSP will translate and correct the requested output code and will load the corresponding D-to-A converter and will finally update the output. When no output change is requested for a period of at least 250 useconds, the on-board DSP will enter a shutdown state. Any access to the shared memory of the M395 will cause the DSP to come out of the shutdown state.

The flow-chart below shows the output update loop:

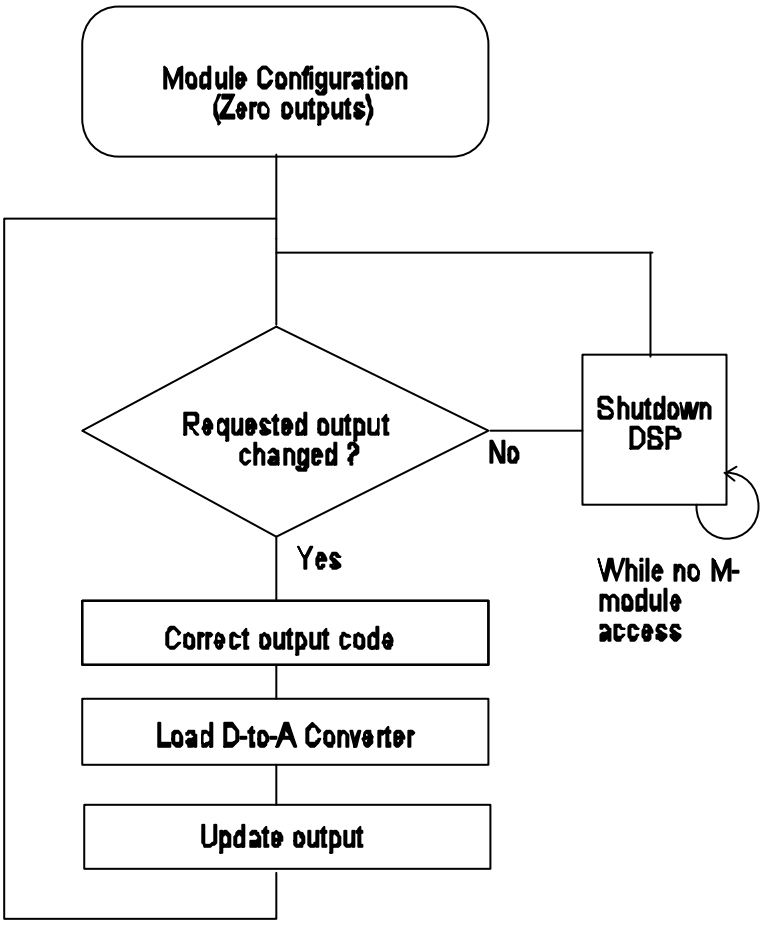


Figure 5 M395 Output update cycle

NOTE: The M395 can be configured for 16-bit continuous mode, in this case the DAC outputs are constantly loaded and updated regardless wether or not new data is requested. The shutdown mode of the DSP is never entered.

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

4.5. ON-DEMAND-UPDATE MODE

The M395 can be configured for 'on-demand-update' mode. This mode is provided for simultaneous output updating. In this mode the user must first write the output data to the ODR registers such as in continuous mode, however the data will not be loaded into the DAC instantaneously. The user must explicitly call the LOAD command to force the DSP to load the corrected output codes to the corresponding DACs, finally the user must call the UPDATE command to instruct the DSP to clock the loaded data to all DAC outputs simulataneously. For details on this refer to section 4.8.4.

NOTE: On-demand-update mode uses always 12-bit output codes.

4.6. CONTINUOUS OUTPUT UPDATE, 16-BIT

The M395 can be configured for 16-bit output resolution. In this mode the overall resolution is 16-bit which is achieved by combining the 12-bit DACs and an additional 4-bit PWM signal generation added to the 12-bit output level. In 16-bit mode the DSP will constantly take data from the ODR registers and generate the 16-bit output values accordingly. The PWM method for achieving 16-bit resolution out of 12-bits DACs is possible thanks to the onboard DPS for the signal generation and thanks to the onboard filtering on each output. Details on the operation are beyond the scope of this manual. Unlike in 12-bit mode, in 16-bit mode the DACs are continuously loaded and updated regardless wether or not the ODR register contents are changed so the onboard DSP will never enter its shutdown state.

4.7. RESET PROCEDURE

After power-up the M395 is kept in reset. The Identification EEPROM can be read by the host. Before any control operation are possible the M395 must be started according to the following reset procedure written in ANSI-C:

```
#define CTRLREG   ((volatile unsigned short *)(<base>+0x82))
#define PAGEREG   ((volatile unsigned short *)(<base>+0x80))
#define GLOBSTAT  ((volatile unsigned short *)(<base>+0x64))

*CTRLREG = 0x0004;                  /* put module in reset */
*PAGEREG = 0x0100;                  /* write 0x0100 to page register */
*CTRLREG = 0x0000;                  /* release reset */

while(!(*CTRLREG & 0x00008));       /* wait until RDY bit is set */

if (GLOBSTAT == 0)                  /* check global status */
      /* M395 ready for usage */
else
      /* Hardware failure */
```

The global status word at offset 0x64 contains status information of the module. Under normal conditions the global status word is zero. A non-zero status word indicates a possible defect of the module and the operation of the module is unreliable. When after a power-up sequence the problem still exists contact AcQuisition Technology for repair.

Make sure only 16-bit write and read accesses are performed. After the reset sequence the M395 is ready for output updating.

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

4.8. ADDITIONAL FUNCTIONS

The M395 is configured through the command interface as described in section 4.2.3. This paragraph gives an overview of the available commands and their usage.

4.8.1. COMMAND SET OVERVIEW

This section lists the commands available to the user, the possible return codes and contains an programming example of command execution on the M395.

Command set:

Command	Hex	Description
DONE	0x0000	command done
SYNC	0x5a5a	synchronize firmware, no action
VERSION	0x1000	get firmware version
RANGE	0x0001	select output range
UMODE	0x0002	set mode (continuous/on-demand/16-bit)
LOAD	0x0003	load D/A converters
UPDATE	0x0004	update analog outputs

Result codes:

Result	Hex	Description
NOERR	0x0000	no error
UNKCMD	0x8001	unknown command
ILLREQ	0x8002	illegal request

Below you can find a software example written in ANSI-C:

```
#define C_CMD     ((volatile unsigned short *)(<base>+0x40))
#define C_RES     ((volatile unsigned short *)(<base>+0x42))
#define C_PRM(x)  ((volatile unsigned short *)(<base>+0x44+((x&0xf)<<1))


*C_PRM(0) = 0;                /* write parameter 0 */
*C_CMD = 5a5a;                /* execute synchronization command */
while (*C_CMD);               /* wait for completion */

if (*C_RES) {                 /* check result */
      /* handle error */
}
else
      /* command successful */
```

4.8.2. GENERAL COMMANDS

SYNC	**Sign of live**

Command: 0x5a5a

Inputs: None

Outputs: None

Function: This function is intended to check whether or not the DSP is running. The local software takes no special action and terminates the execution normally by setting the result field to NOERR and clearing the command field.

VERSION	**Get module information**

Command: 0x1000

Inputs: None

Outputs: Module number (should be 395d)
Firmware version number

Function: This function returns information about the module and the local firmware. When the function returns, parameter 0 contains the module number which should be 395(d). Parameter 1 contains the firmware version number: for example 10(d) which means version 1.0.

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

4.8.3. CONFIGURATION COMMANDS

RANGE — Select output range

Command: 0x0001

Inputs: Output range

Outputs: None

Function: With this function the output range of the M395 can be selected. The table below lists the available ranges:

Input Range	Range Selection
0...10 V unipolar +/-10 V bipolar	0x0000 (default) 0x0001

Note: The range command will set all channels to 0V output regardless of the contents of the ODR registers.

UMODE — Select output update mode

Command: 0x0002

Inputs: Output update mode

Outputs: None

Function: With this function the output update mode of the M395 can be selected. When the function is called with parameter 0 cleared, continuous update mode is selected. If the function is called with parameter 0 filled with 0x0001, on-demand-update mode is selected. Parameter 0 set to 0x0002 selects the 16-bit mode.

Update Mode	Parameter 0
12-bit continuous on-demand 16-bit continous	0x0000 (default) 0x0001 0x0002

In continuous mode output data, written to the ODR registers is automatically loaded to the corresponding DAC and the output is updated. In on-demand-update mode data written to the ODR registers must be explicitly loaded into the DAC's with the LOAD command and the outputs must be updated with the UPDATE command. For details refer to the next section 4.8.4.

4.8.4. ON-DEMAND-UPDATE MODE COMMANDS

The commands described in this section are available when the module is configured for the on-demand-update mode (with the UMODE command as described in section 4.8.3).

LOAD	**Load DAC's**

Command: 0x0003

Inputs: None

Outputs: None

Function: This command will check the ODR registers (see section 4.2.2) for changes, if there are any the corresponding DAC's will be loaded with the corrected output codes. The outputs of the DAC's must be updated with the UPDATE command.

UPDATE	**Update analog outputs**

Command: 0x0004

Inputs: None

Outputs: None

Function: This command is provided for simultaneous output update of the DAC's pre-loaded with the LOAD command.

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

Intentionally left blank.

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 M395 12-bit DAC M-module with 16x Voltage Outputs. The example software is available in ANSI-C source code and consists mainly of an M395 function library which provides functions for easy access to the M395 and a demo program which illustrates the usage of the software library.

The M395 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 M395 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 M395 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
PHA8	volatile unsigned char *	8-bit physical access
PHA16	volatile unsigned short *	16-bit physical access
PHA32	volatile unsigned long *	32-bit physical access
APIS_PATH	unsigned long	APIS physical path ID
APIS_HANDLE	void *	APIS physical path handle
APIS_WIDTH	int	APIS access size in bytes
IDCODE	union {	ID EEPROM contents
	struct {	
	short synccode,	Sync code (0x5346)
	modnum,	Module number (binary coded)
	revision,	revision number
	modchar,	Module characteristics
	res[4];	reserved
	char manstr[16];	manufacturer string
	} id;	ID data
	short reg[16];	raw data
	}	

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

5.3. FUNCTION REFERENCE

This section contains the reference of the functions provided by the M395 software library.

5.3.1. GENERAL FUNCTIONS

m395_open () — Open path to M395

Syntax: `int m395_open(APIS_PATH path_id, APIS_HANDLE *handle)`

Description: Open a hardware path to M395

Arguments: APIS_PATH path_id
Module APIS path ID
APIS_HANDLE *handle
Pointer to hardware path handle

Returns: APIS error code

Example: `m395_open(path_id, &handle);`

m395_close () — Close path to M395

Syntax: `void m395_close(APIS_HANDLE handle)`

Description: Closes hardware path to M395

Arguments: APIS_HANDLE handle
Hardware path handle

Returns: Nothing

Example: m395_close(handle);

m395_boot () — Boot the M395

Syntax: `int m395_boot(APIS_HANDLE handle)`

Description: Boot the M395 from RAM or ROM depending on which is enabled. Default the M395 will boot from ROM, the RAM boot option is mostly used for service purposes.

Arguments: APIS_HANDLE handle
Hardware path handle (returned by apis_open())

Returns: 0 or -1 in case of hardware failure

Example:

```
m395_boot(handle); /* boot M395 */
```


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

m395_cmd() Execute command

Syntax: `int m395_cmd(APIS_HANDLE handle, UINT16 command, UINT16 *pArglist, int ni, int no)`

Description: Copy command parameters to the parameter field in shared ram of the M-module. Execute command and wait for the command to be completed. Copy the result parameters and return with the result code obtained from the firmware.

Below is a table with available commands:

Command	Description
SYNC	Synchronize firmware, sign of live
VERSION	Get firmware version
RANGE	Select output range
UMODE	Set mode (continuous 12-bit/ on-demand/ continuous 16-bit).
LOAD	Load D/A converters
UPDATE	Update analog outputs

Arguments:

APIS_HANDLE handle
Hardware path handle
UINT16 command
Firmware command
void *pArglist
Pointer to the command parameter list
int ni
Number of input command parameters
int no
Number of output command parameters

Returns: Command result code

Command result codes:

Result	Description
NOERR	No error
UNKCMD	Unknown command
ILLREQ	Illegal request

Example:
```
m395_cmd(handle, SYNC, (UINT16 *)(&params), 0, 0);
```

5.3.2. CONFIGURATION FUNCTIONS

m395_select_range() — Select output range

Syntax: `int m395_select_range(APIS_HANDLE handle, UINT16 outp_range)`

Description: Configure output range of all channels. See following table for valid output ranges.

Output range	Description
V10UNI	0 .. 10 V unipolar
V10BI	+/- 10 V bipolar

Arguments: APIS_HANDLE handle
Hardware path handle
UINT16 outp_range
Output range

Returns: NOERR on success and INV_VALUE when output range has an illegal value

Example:
```
m395_select_range(handle, V10UNI);
```

m395_select_upd_mode — Select update mode

Syntax: `int m395_select_upd_mode(APIS_HANDLE handle, UINT16 upd_mode)`

Description: With this function the update mode of the M395 can be selected. In continuous mode outputs are automatically loaded. In on-demand-update mode all channels are updated when m395_updchan() is called. See following table for valid update modes.

Update mode	Description
CONT	Continuous 12-bit
ON_DEM	On-demand
MODE_16BIT	Continuous 16-bit

Arguments: APIS_HANDLE handle
Hardware path handle
UINT16 upd_mode
Update mode

Returns: NOERR on success and INV_VALUE when update mode has an illegal value

Example:
```
m395_select_upd_mode(handle, CONT);
```


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

5.3.3. UPDATE FUNCTIONS

m395_setchan — Set output of one channel

Syntax: `int m395_setchan(APIS_HANDLE handle, UINT16 channel, UINT16 value)`

Description: Sets output of one channel. If update mode is continuous, channel is automatically updated when update mode is on-demand, m395_updchan() must be called to update channels.

Arguments: APIS_HANDLE handle
Hardware path handle
UINT16 channel
Number of channel (0-7).
UINT16 value
Binary code for output value.
Unipolar - Straight binary / Bipolar - Two's complement

Returns: NOERR on success

Example: `m395_setchan(handle, 3, 0x400);`

m395_setallchan — Set output of all channels

Syntax: `int m395_setallchan(APIS_HANDLE handle, UINT16 value)`

Description: Sets output of all channels. If update mode is continuous, the channels are automatically updated when update mode is on-demand, m395_updchan() must be called to update channels.

Arguments: APIS_HANDLE handle
Hardware path handle
UINT16 value
Binary code for output value.
Unipolar - Straight binary / Bipolar - Two's complement

Returns: NOERR on success

Example: `m395_setallchan(handle, 0x400);`

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

5.3.4. ON-DEMAND-UPDATE MODE FUNCTIONS

This function must only be used when the update mode is on-demand.

m395_updchan — Update all channels

Syntax: `int m395_updchan(APIS_HANDLE handle)`

Description: Updates all channels simultaneously and can only be used when update mode is on-demand

Arguments: APIS_HANDLE handle
Hardware path handle

Returns: NOERR on success and ILLREQ when update mode is continuous

Example:
```
m395_updchan(handle);
```

5.4. SOFTWARE DISTRIBUTION

This section gives an overview of the software distribution.

File	Description
M395\SOFTWARE\m395rel.txt	Release notes
M395\SOFTWARE\LIB\m395lib.c	APIS based ANSI-C M395 software library
M395\SOFTWARE\LIB\m395defs.h	Definitions for M395 software library
M395\SOFTWARE\EXAMPLE\m395demo.c	Demo program
M395\SOFTWARE\EXAMPLE\m395demo.mak	Example make file for Borland C
MMODID\SOFTWARE\LIB\modideep.c	APIS based ANSI-C M-module ID EEPROM software library
MMODID\SOFTWARE\LIB\ideeprom.h	M-module ID EEPROM definitions

M395 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 M395 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
     |         +---....
     +---M395                       M395 distribution
     |    +---DOC                   M395 documentation
     |    +---SOFTWARE
     |         | m395rel.txt        Release notes / version info
     |         +---LIB              M395 support libraries
     |         |    m395defs.h
     |         |    m395lib.c
     |         +---EXAMPLE          M395 example software
     |              m395demo.c
     |              m395demo.mak
     +---MMODID
          +---SOFTWARE
```


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.

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

Data Sheet of the TL074C
Texas Instruments
PO Box 655303 Dallas, Texas 75265

6.2. COMPONENT IMAGE

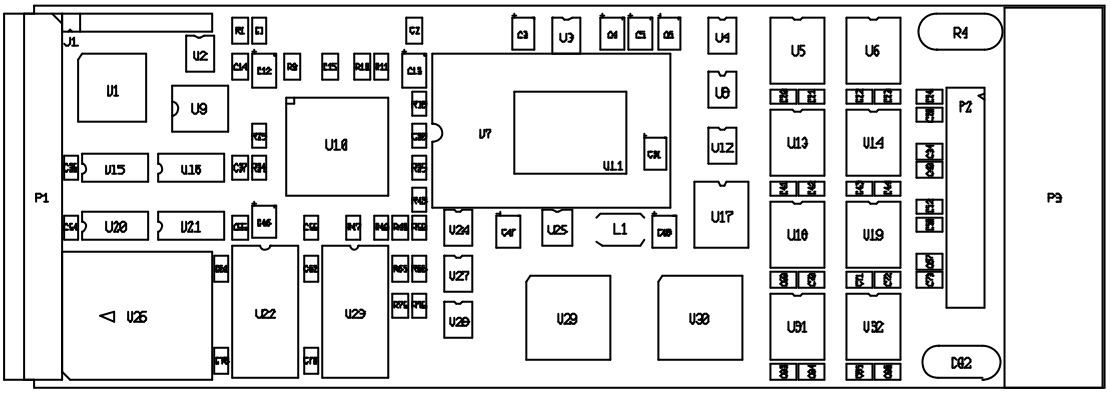


Figure 6 M395 Component Image

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.
To peripheral via 24 pole header

Power supply with on-board DC/DC converter:
+5VDC ±10%, typical 850mA.

Power supply without on-board DC/DC converter:
+5VDC ±10%, typical 250mA (obtained from M-module bus)
+15VDC ±5%, maximal 200mA (externally applied)
-15VDC ±5%, maximal 100mA (externally applied)

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

Humidity:
Class F, non-condensing.

6.4. DOCUMENT HISTORY

! Version 1.0
First release.
! Version 1.1
New layout.
APIS support added.
! Version 1.2
M395/NDC and M395/WDC versions added.
Code examples removed.
! Version 1.3
16-bit output mode added.