Contents

Manual Released Version: 3.2

The information provided herein is believed to be accurate and reliable; however, Microcon assumes no responsibility for inaccuracies or omissions. Prices and specifications are subject to change without notice. No license is granted by implication or otherwise under any patent or patent right. No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording for any purpose without the written permission of Microcon. Microcon does not authorize any Microcon products for use as critical components in life support devices or systems.

Warranty

Microcon warrants to its Customers that the products that it manufactures and sells will be free from defects in materials and workmanship for the period of 12 months from the date of shipment. Microcon warrants that its software and firmware products will not fail to execute their programming instructions due to defects in materials and workmanship. If Microcon receives notice of such defects during the warranty period, Microcon shall at its option replace or repair software media or firmware which do not execute their programming instructions due to such defects. Microcon does not warrant that the operation of the software, firmware or hardware shall be uninterrupted or error free. This warranty is given by Microcon in lieu of any other warranties, expressed of implied. Microcon disclaims any implied warranty of merchantability or fitness for a particular purpose. Microcon responsibility to repair or replace a defective product is the sole and exclusive remedy provided to the customer for breach of this warranty. Microcon will not be liable for any indirect, special, incidental or consequential damages.

Trademark Acknowledgements

MICROCON is a registered trademark of MICROCON Co.
IBM PC is a registered trademark of IBM Corp.
MS-DOS is a registered trademark of MICROSOFT Corp.

I Main Features

* FULLY PROGRAMMABLE SINGLE CHIP INDEXER AND SEQUENCER

* MOTOR SPEEDS UP TO 40 000 STEPS/SEC

* UP TO 64 MICROSTEPS PER STEP

* HIGH PERFORMANCE MOTION PROFILES

* ONE STANDARD AND TWO MICROSTEP MODES OF DRIVER INTERFACE:
  - STEP AND DIRECTION
  - DUAL DAC INTERFACE
  - TWO DACs INTERFACE

* POWERFUL INSTRUCTION SET OF 50 COMMANDS

* 16 MILLION STEPS PER MOTION

* INTERNAL MEMORY ENABLES PRE-PROGRAMMED MOVE SEQUENCES TO BE REPEATED OFF-LINE

* UP TO 21 GENERAL PURPOSE I/O

* PROGRAMMABLE START AND SLEW RATE, ACCEL/DECEL RAMPS

* ESPECIALY SMOOTH ACCELERATION AND DECELERATION

* PROGRAMMABLE DYNAMIC MICROSTEPPING

* SERIAL HOST INTERFACE, UP TO 16 DEVICES PER PORT

* LOW POWER CMOS, TTL COMPATIBLE CHIP

* AVAILABLE IN 40-PIN DUAL IN-LINE (DIP) OR 44-PIN PLCC PACKAGES

II General Description

The M1486 controller provides the user with an integration of a precision motion control with other programmable machine control functions such as I/O on a single chip, thus eliminating a need to use additional units for complete machine control. The motion control side of the M1486 provides a fully programmable single-axis indexer and a full step, half step and microstepping sequencer.

The primary purpose of the M1486 is to receive instructions from a host computer and convert them to digital values of phase currents for both phases of a stepper motor. Standard pulse and direction outputs for driving an external drive are also available.

◆ PROGRAMMING LANGUAGE

The controller has its own high level programming language, providing a powerful instruction set of more than 50 commands for stepper motor systems control. This includes jump and conditional jump commands which can be used for program execution branching according to user input and user flag status. Internal memory enables pre-programmed command files sequences to be repeated off-line.

The M1486 commands and data are sent from the host computer via a serial link as ASCII characters (it is possible to use both uppercase and lowercase alpha characters) of the form: command, argument, carriage return. The user can employ any text editor for ASCII command files constructing.

◆ HIGH SPEED CONTROL

Precise continuous high speed control method delivers maximum performance and explores maximum of the stepper motor torque in an open loop system, allowing the use of a smaller motor and more compact driver package. This means substantially lower motor and drive costs or increased safety margin of an open loop stepper motor system.

◆ DYNAMIC MICROSTEPPING

The 1486 provides higher step resolution that virtually eliminates low speed resonance and step-to-step oscillation in stepper motor systems.

The number of microsteps per full step can be programmed from 1 to 64 separately for low speed range and high speed range in order to achieve maximum speed as well as very high resolution at low speed.

◆ SERIAL INTERFACE

The M1486 features full duplex TTL (logic) level serial interface. RS232 , RS422 or RS485 interface can be established using single chip driver/receiver. Driver/receiver ICs which convert TTL voltage levels to RS232, RS422 or RS485 output signals are available from many suppliers.

The baud rate can be set at one of four speeds using pins 33 (BR1) and 34 (BR0), by selecting the required combination of 0 or 1 logic levels according to the following table:

BR1	BR0	M1486 Baud Rate	M1486..2x with 24 MHz crystal Baud Rate
0	0	300	600
0	1	1 200	2 400
1	0	2 400	4 800
1	1	4 800	9 600

Data format: 8 Data Bits, No Parity Bit, One Stop Bit

Due to time needed for EEPROM write, is necessary so that total time for one command transmission to M1486E is 0.5 sec at least.

Easily programmed via a serial link, up to sixteen M1486A or M1486B controllers (hexadecimal addresses from 0 to F) may be daisy-chained by setting up the address of each controller using pins ADDR0 to ADDR3 to allow simultaneous control of multiple axes.

Up to fifteen M1486E controllers (hexadecimal addresses from 1 to F) may be daisy-chained. If a low is held on all of the address inputs - ADDR0, ADDR1, ADDR2 and ADDR3 (hexadecimal address 0) a high on RST input resets the device. A high on RST input is always present at power-up due to recommended power-up 2,2 µF capacitor.

The selection of an indexer which is to receive following commands is made by the "*IndeX*" command. Selected indexer continues to receive commands until another "*IndeX*" command is received. The default setting is receiving enabled. The user needn´t care of address pins and the "*IndeX*" command for the single controller (point-to-point operation).

TDX outputs may be externally pulled low. It is possible to connect RDX and TDX pins of up to four M1486 controllers directly in parallel.

BR0 and BR1 inputs share controller pins with I/O lines B5 and B6. BR0 and BR1 inputs are queried after pulling RST input high only. Because a high on RST input is always present at power-up due to recommended power-up 2,2 µF capacitor, baud rate is always queried at power-up.

◆ MEMORY

Unlimited number of host controller step by step commands can be sent to the M1486 due to fully scrollable internal memory. For stand alone operation internal memory enables pre-programmed move sequences to be repeated off line. When the controller is disconnected from the host computer it can be controlled by external events by means of general purpose inputs and can handle external elements by means of general purpose outputs.

Commands transmitted via serial link in ASCII code are first compressed internally and then stored in command memory in the following fixed structure:

command character - 8 bits

command argument stored in the following binary form:

Number of bits is a multiple of eight and $2^{\text{number of bits}}$ is more or equal to maximum permissible argument value presented in this user manual.

Example:

"R"	8-bit character + 0-bit argument = **8 bits**
"L5"	8-bit character + 8-bit argument (range up to 255) = **16 bits**
"A1000"	8-bit character + 16-bit argument (range up to 65 000) = **24 bits**
"W500"	8-bit character + 24-bit argument (range up to 16 000 000) = **32 bits**

Length of command stored in command memory depends on permissible range of command argument, and is independent of specified argument.

The "*Address*", "*Disable*", "*Enable*" and "*Reset*" commands take no command memory space.
The "*If*" command length is 32 bits including command character.

It is possible to determine necessary command memory space for longer command files in advance and check if command memory capacity is sufficient for specified command file.

Memory capacity:

Type	input memory	command memory
M1486A	96 bits	864 bits
M1486B	256 bits	2032 bits
M1486E1	256 bits	2000 bits
M1486E2	256 bits	4000 bits

Compression of received commands is interrupted and they are stored in small input memory when some commands, e.g. "*Run*", are executed.

Filling up the input buffer memory is reported to the host computer in following way:

When received character is loaded to one of last three free bytes of buffer memory, this character is transmitted back via serial link in inverted form.

When transmission continues regardless this report and attempt is made to write to the first occupied input memory byte, the character transmitted from the host computer won't be received and the controller will transmit error report - "5eH" character - back to the host computer via serial link.

Easiest way to overcome input buffer limitations is to use the *"Disable"* command at the beginning of command file and the *"Enable"* command at the end of command file. Commands will go easily through input buffer to command memory unrestricted by the input buffer size. Command file length is limited only by command memory size.

Command file closed in square brackets can´t include the *"Reset"* command. Otherwise it would delete the *"Disable"* command.

In addition pre-programmed motion & machine control sequences may be stored in an on-chip EEPROM (M1486E). These sequences can be automatically executed on power-up. The program stored in memory can be easily changed after re-connecting of your computer.

In order to overcome input buffer limitations also in on-line control with the M1486 is necessary to split commands to blocks of commands, each block using the *"Disable"* command at the beginning of command file and the *"Enable"* command at the end of command file.

It is advisable to wait till command block is executed before next block transmission. If preceding command block execution is not finished new commands will be stored in input memory and allowable number of transmitted commands would by limited by input memory.

It is possible to include suitable command, e. g. *"Upload"* to the end of each command block. Host computer can then determine end of preceding block execution by the moment when it receives answer from the controller. The controller transmits data requested by the *"Upload"* command subsequent to preceding commands execution.

User outputs can also be used to inform that preceding command block execution is finished.

◆ JOG CONTROL

Manual jog control switches may be connected to the M1486 to provide a flexible manual control facility. Even when used under manual control, absolute position is continually updated and may be uploaded to the host computer.

◆ USER I/O

There are up to 21 digital user input/outputs which are available to initiate or monitor external events. User input and output lines are used for controlling program execution and for general machine control functions such as interfacing to sensors and other external devices.

Each of I/O lines B1 - B7 can be independently used as an input or an open drain output.

I/O lines B8 and B9 are available with Step & Direction and Dual DAC modes of the driver interface. Input/output lines B10 to B17 are available with Step & Direction mode of the driver interface only. In other driver interface modes digital current values are set by means of these pins.

I/O lines B5 and B6 share controller pins with inputs BR0 respectively BR1. BR0 and BR1 inputs are queried after pulling RST input high only. Because a high on RST input is always present at power-up due to recommended power-up 2,2 µF capacitor, baud rate is always queried at power-up.

I/O lines B18 to B21 share controller pins with inputs ADDR0 to ADDR3. Controller address is queried after pulling RST input high only. Because a high on RST input is always present at power-up due to recommended power-up 2,2 µF capacitor, controller address is queried at power-up.

If the M1486A or M1486B controller is in the shutdown mode of operation, power-up reset terminates the shutdown mode and controller address inputs ADDR0 to ADDR3 are not queried.

◆ DRIVER INTERFACE

The driver interface is the part of the M1486 controller that issues the signals to an external motor driver.

The M1486 has one standard and two microstep modes of driver interface:

- Step & Direction
- Dual DAC interface
- Two DACS interface

Standard Step & Direction mode can be selected by tying **S&D** input pin low. Pulse clock is emitted through pin **Step. Direction** pin is motion direction output. Please note that the pulse count per full step corresponds to the selected full step division. See the *"Microstep"* command.

The Dual DAC mode can be selected by tying **DUAL** input pin low. The Dual DAC interface simplifies wiring and reduces component count using one dual DAC for both phases in the microstep mode. **D2.0-D2.6** output pins emit 7bit motor current data for phase 1 and 2. MSB is D2.6. **PHASE 2** output pin selects direction of current flow for phase 1 and 2. **DIR** output pin selects receiving channel (channel 1 for DIR = 0, channel 2 for DIR = 1) of external dual DAC. Active low output **STROBE** latches output data (D2.0-D2.6 and PHASE2) into external dual DAC.

The Two DACS interface uses outputs **D1.0-D1.6** and **PHASE1** for motor phase 1 current value and direction specification and outputs **D2.0-D2.6** and **PHASE2** for motor phase 2 current value and direction specification.

D1.6 and PHASE2 of M1486B are open drain outputs.

III Command Summary

COMMAND	DESCRIPTION
\	"Reset" - set default values
@ (Num)	"Address" - specifies label number on current program line
A (Num)	"Acceleration" - range = 1 to 65 000 steps/sec^2
B (Num)	"Backward" - move (Num) steps in negative direction, range = 1 to 16 000 000
C (Num)	"Clear" - clear output; range = 1 to 21
C (Num)	"Clear" - clear auxiliary function; range = 40 to 63
C (Num)	"Clear Kill" - resume program execution
C (Num)	"Clear" - clear user flag; range = 80 to 87
C100	"Clear position counter"
D	"Direction" - reverse next move direction
E	"End of loop"
F (Num)	"Forward" - move (Num) steps in positive direction, range = 1 to 16 000 000
G (Num)	"Go absolute" - go to absolute position; range = 1 to 16 000 000
G+	"Go positive" - go continuously in positive direction till external interrupt
G-	"Go negative" - go continuously in negative direction till external interrupt
H	"Home" - run home
I (Num)(Value)(Num)	"If" - if (Num) input is equal to (Value) transfers program control to the (Num) label
J (Num)	"Jump" - transfers program control to the (Num) label
K	"Kill" - decelerate immediately, stop program execution
L (Num)	"Loop" - repeat subsequent commands (Num) times
M (Num)	"Microstepping" - number of microsteps per full step at low speed range
N (Num)	"Number" - select current profile characteristic
O (Num)	"One" - wait for input (Num) to go to logical one
P (Num)	"Profile" - set maximum velocity for linear acceleration profile, acceleration continues with parabolic profile
Q (Num)	"Qualification" - number of microsteps per full step at high speed range
R	"Run" - make a move
S (Num)	"Start/stop" - start/stop velocity; range = 16 to 1 950 steps/sec
T (Num)	"Turn on" - set specified output high, range = 1 to 21
T (Num)	"Turn on" - turn specified auxiliary function on, range = 40 to 63
T (Num)	"Turn on" - turn specified user flag, range = 80 to 87

U (Num)	"Upload" - upload absolute position, user variable, user flags and auxiliary functions status
V (Num)	"Velocity" - set maximum velocity
W (Num)	"Wait" - wait for (Num) milliseconds; range = 1 to 16 000 000
X (Num)	"indeX" - select indexer (Num)
Z (Num)	"Zero" - wait for input (Num) to go to logical zero
[	"Disable" - following commands execution
]	"Enable" - previous commands execution
((Num)	"Seek negative" - seek limit in negative direction
) (Num)	"Seek positive" - seek limit in positive direction
= (Num)	"Equal" - load position counter with specified data
: (Num)	"Load" - load internal variable with specified data
? (Num)	"Query" - query data on specified inputs and load them to internal variable
! (Num)	"Order" - Put user variable data to specified outputs
+ (Num)	"Add" - add specified data to user variable
- (Num)	"Subtract" - subtract specified data from user variable
/ (Num)	"Divide" - divide user variable by specified data
* (Num)	"Multiply" - multiply user variable and specified data
< (Num)	"Move to register" - copy data from specified register to user variable
> (Num)	"Move from register" - copy data from user variable to specified register
' (Num)	"Subroutine" - call subroutine starting at specified label
.	"End of subroutine"

IV Electrical Specifications

◆ ABSOLUTE MAXIMUM RATINGS

Input voltage on any pin to Vss	-0,5 to + 6,5 V
Input or output current on any pin	10 mA
Total power dissipation	200 mW
Storage temperature range	-65°C to +150°C
Operating temperature range	
Commercial devices	0°C to +70°C
Industrial devices	-40°C to +85°C

◆ ELECTRICAL CHARACTERISTICS

Parameter	**Min.**	**Max.**	**Unit**
Supply voltage	4,5	5,5	V
Supply current		20	mA
Supply voltage at shutdown - M1486A, M1486B	2	5,5	V
Supply current at shutdown - M1486A, M1486B		50	µA
Input low voltage	-0,5	0,2Vcc-0,1	V
Input high voltage	0,2Vcc + 0,9	Vcc + 0,5	V
Output low voltage		0,45	V
Output low current		1,6	mA
Output low current, outputs Step, Direction		4	mA
Output high voltage	2,4		V
Output high current		60	µA
Data valid to Strobe transition	960		ns
Step pulse width	1,8		µs

ESD WARNING; NORMAL HANDLING PRECAUTIONS SHOULD BE TAKEN TO AVOID STATIC DISCHARGE.

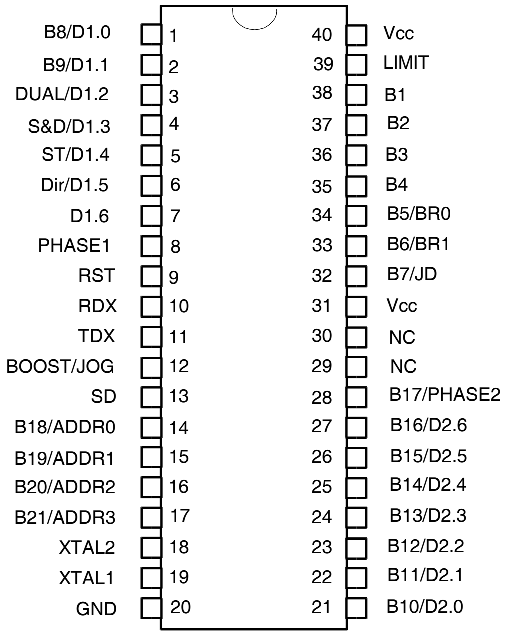
◆ SHUTDOWN MODE

Active low input SD - No. 13 invokes Shutdown mode of M1486A and M1486B controllers which is used for battery backup. Nevertheless the M1486E is regularly used for applications with command file backup. The M1486E contains EEPROM memory and it doesn´t need battery for command file backup.

In the shutdown mode of operation typical power consumption sinks to 5 µA, machine & motion program is retained but its execution is stopped. No communication with the outside is possible. Hold the reset pin high for 10 ms to restore operation after shutdown. Vcc can be reduced to as low as 2 V in the shutdown mode, especially useful for battery operation.

V Pin Connection

(DIL VERSION)



VI Pin Description (DIL versions)

Designation	Pin No.	Description
D1.0-D1.6	1 - 7	7-bit output current data for phase 1 (Two DACs mode) MSB = D1.6
PHASE 1	8	Logic output selects direction of current flow for phase 1 (Two DACs mode)
B8-9	1 - 2	General purpose input/outputs (Step & Direction and Dual DAC modes)
$\overline{\text{DUAL}}$	3	Active low input selects Dual DAC mode
$\overline{\text{S\&D}}$	4	Active low input selects Step & Direction mode
$\overline{\text{STEP}}$	5	Pulse clock output (Step & Direction mode)
DIRECTION	6	Motion direction output, a high on this output selects negative direction (Step & Direction mode)
$\overline{\text{STROBE}}$	5	Active low output latches output data (D2.0-D2.6 and PHASE2) into external dual DAC (Dual DAC mode)
DIR	6	Logic output selects receiving channel (channel 1 for DIR = 0, channel 2 for DIR = 1) of external dual DAC. (Dual DAC mode)
RST	9	Reset; high on this pin resets the device. A 2,2 μF capacitor should be connected between this input and Vcc for automatic power-up reset. A high on RST input resets the device on condition that a low is held on all of the address inputs - ADDR0, ADDR1, ADDR2 and ADDR3. If any of the address inputs is held high then a high on RST input stops command file execution and command file execution starts again from the beginning.
RDX	10	Serial port receiver data input
TDX	11	Serial port transmitter data output
JOG	12	Falling edge triggered external pulse clock input - Jog operation. Jog operation is not possible when the *"Run"* or *"Seek"* command is executed.

Designation	Pin No.	Description
$\overline{\text{BOOST}}$	12	When the controller is executing the *"Run"* or the *"Seek"* commands BOOST output is cleared internally. Boost output enable the user to raise motor current when motor is not moving.
$\overline{\text{SHUTDOWN}}$	13	Active low input invokes Shutdown mode
ADDR0-3	14 - 17	Address inputs select controller address in the multipoint operation
B18-21	14 - 17	General purpose input/outputs
XTAL1, XTAL2	19, 18	External 12 MHz (24 MHz for 2x versions of the controller) crystal should be connected to XTAL1 and XTAL2. 20 pF capacitors should be connected between these inputs and ground
Vss	20	Ground: 0 V reference
D2.0-D2.6	21 - 27	7-bit output current data for phase 2 (Two DACs mode) 7-bit output current data for phase 1 and 2 (Multiplexed, Dual DAC mode) MSB = D2.6
PHASE2	28	Logic output selects direction of current flow for phase 2 (Two DACs mode) Logic output selects direction of current flow for phase 1 and 2 (Multiplexed, Dual DAC mode)
B10-17	21 - 28	General purpose input/outputs (Step & Direction mode)
Vcc	31, 40	+5 V Logic supply voltage
JOG DIRECTION	32	Motion direction input, a high on this input selects negative direction (Jog operation)
B1-B7	38 - 32	General purpose input/outputs
$\overline{\text{LIMIT}}$	39	Active low input stops motion. Motor decelerates immediately at speeds outside the start/stop range with the same slope it has accelerated with. LIMIT discontinues program execution according to the auxiliary function *"Recover"*. Setting this input high resumes program execution.
BR0, BR1	34, 33	Baud rate 0 and Baud Rate 1 inputs select baud rate for the serial interface at power-up
NC	29, 30	Not connected. These pins should be left floating.

VII Command Description

1 Program Control Commands

@	*Address*
L	*Loop*
E	*End of loop*
J	*Jump*
I	*If*
K	*Kill*
C	*Clear kill*
U	*Upload*
W	*Wait*
[	*Disable*
]	*Enable*
'	*Subroutine*
.	*End of subroutine*

◆ ADDRESS

SYNTAX: **@ (Num)**

Type	**M1486A**	**M1486B, M1486E**
Range	1 to 15	1 to 80

The "*Address*" command specifies label number on the current program line. Program execution may be transferred on a program line specified by the "*Address*" command by the "*Jump*" or by the "*If*" command.

EXAMPLE:

@2

This command assigns label number 2 to current program line.

LOOP

SYNTAX: **L** (**Num**) **RANGE:** 1 to 255

The "*Loop*" command repeats subsequent commands till the "*End of loop*" command (Num) times. The user can nest four "*Loop*" commands on the M1486.

EXAMPLE: See Motion Program Example No. 2

END OF LOOP

SYNTAX: **E**

The "*End of loop*" command terminates program loop; program execution continues from the command following corresponding the "*Loop*" command. Number of loops is specified in the "*Loop*" command, afterwards program execution continues from the command following the "*End of loop*" command. It is not possible to jump outside the loop. Contingent branch directed by the "*Jump*" or by the "*If*" command must also include the "*End of loop*" command.

EXAMPLE: See Motion Program Example No. 2

IF

SYNTAX:	**I** (Input or flag **Num**),	(**Value**),	(Label **Num**)	
RANGE:	(**1** to **87**),	(**Low** or **High**),	(**1** to **15**)	M1486A
			(**1** to **80**)	M1486B, M1486E

Conditional jump command can be used for program execution branching according to user input or user flag status. The "*If*" command transfers program control to the specified label if the specified input or flag is equal to the specified value. Input lines B8 and B9 are available with Step & Direction and Dual DAC modes of the driver interface. Input lines B10 to B17 are available with Step & Direction mode of the driver interface only.

EXAMPLE:

```
[
@1
w3000
i7h3
b64000
r
j1
@3
f6400
r
j1
]
```

If input No. 7 is Low, program execution continues from the next command - b64000.

If input No. 7 is High, program flow is transferred to the command following the command @3 - f6400.

See Motion Program No. 3 for another example.

◆ JUMP

SYNTAX: **J (Num)**

Type	**M1486A**	**M1486B, M1486E**
Range	1 to 15	1 to 80

The "*Jump*" command transfers program control to the specified label.

EXAMPLE:

J6

Program flow is transferred to the command following the command "*Address*" 6.

◆ KILL

SYNTAX: **K**

The "*Kill*" command stops motion. Motor decelerates immediately at speeds outside the start/stop range with the same slope it has accelerated with.

The "*Kill*" command discontinues program execution according to the auxiliary function "*Recover*".

◆ CLEAR KILL

SYNTAX: **C75**

Resume program execution stopped by the "*Kill*" command. See also the auxiliary function "*Recover*" for options.

◆ UPLOAD

SYNTAX: **U** (**Num**) **RANGE:** 1 to 18

This command uploads specified byte from the M1486 controller to the host computer according to the following table. For 24-bits parameters (absolute position counter, internal variable) the following rule is valid: the higher number the more significant byte.

Register No.	Description
1	Absolute position - Low Significant Byte
2	Absolute position - Mid Significant Byte
3	Absolute position - Most Significant Byte
4	Internal Variable - Low Significant Byte
5	Internal Variable - Mid Significant Byte
6	Internal Variable - Most Significant Byte
7	Reserved
8	Reserved
9	Reserved
10	Reserved

Register No.	Description
11	Reserved
12	Reserved
13	Reserved
14	User flags No. 80-87 status
15	Driver interface mode
16	No. of the controller production series
17	Auxiliary functions No. 40-47 status
18	Auxiliary functions No. 48-55 status
19	Auxiliary functions No. 56-63 status
20	Controller type ASCII code M1486A 41H M1486B 42H M1486E1 45H M1486E2 46H

EXAMPLE:

U3

The M1486 transmits the most significant byte of the absolute position counter to the host controller via the serial link.

◆ WAIT

SYNTAX: **W (**Num**)** **RANGE:** 1 to 16 000 000 ms

Dwells can be programmed in the broad range from 1 to 16 000 000 ms.

Commands received via the serial interface are processed during the "*Wait*" command execution. Though this feature effectively increases input command buffer the "*Disable*" and "*Enable*" commands are used for file transmission not limited by the input buffer size.

At the beginning of the command file execution after power-up external logic levels connected to controller inputs may not be stabilized yet, e.g. some sensors can react to power-up after a fixed time delay. That's why it's useful to use the "*Wait*" command (e.g. W1000) in the beginning of each command file placed immediately after the "*Disable*" command.

Using the "*Wait*" as first command of the command file reduces also power supply requirements because while electrolytic filtering capacitors are charging, motor motion don't occur yet and power consumption is not further increased.

EXAMPLE:

W3000

The M1486 waits for 3 seconds prior to the next command execution.

- ## DISABLE

SYNTAX: **[**

Delay execution of the following set of commands. Following commands are processed but not executed and so don't raise demand for input memory size.

EXAMPLE: See Motion Program Example No. 4

- ## ENABLE

SYNTAX: **]**

Enable execution of the commands delayed by the corresponding the "*Disable*" command.

EXAMPLE: See Motion Program Example No. 4

◆ SUBROUTINE

SYNTAX: ' (Num)

Type	M1486A	M1486B, M1486E
Range	1 to 15	1 to 80

Subroutine command character is the apostrophe - ' - (character 27H).

This command transfers program control to the specified label. Program flow is transferred back to the command following the *"Subroutine"* command after finishing subroutine by the *"End of subroutine"* command.

It's not possible to use another *"Subroutine"* command in subroutine.

EXAMPLE:

```
[
@1
i1l6
f12800
'3
z3
j1
@6
b25600
'3
z4
j1
@3
r
w2000
d
r
.
]
```

Depending on input 1 status there are two different program branches.

Command file part starting with @3 contains commands shared by both of the branches.

◆ END OF SUBROUTINE

SYNTAX: **.**

Character of the *"End of subroutine"* command is the dot - . - (character 2EH).

Program flow is transferred back to the command following the *"Subroutine"* command after finishing subroutine by the *"End of subroutine"* command.

2 Motion Control Commands

Move distance specifications commands

B	*Backward*
D	*Direction*
F	*Forward*
G	*Go absolute*
G+	*Go positive*
G-	*Go negative*
H	*Home*
=	*Equal*

Move velocity specifications commands

A	*Acceleration*
S	*Start/stop*
P	*Profile*
V	*Velocity*

Move execution commands

R	*Run*
)	*Seek positive*
(	*Seek negative*

◆ BACKWARD

SYNTAX: **B (Num)** **RANGE:** 1 to 16 000 000

Move relative, specifies distance relative to current position in negative direction. The unit is minimum microstep = full step/64. The default setting is 64 000 minimum microsteps = 1 000 full steps = 5 revolutions for 1.8° stepper motor.

Note:
Distance or position used as argument of the *"Backward"*, *"Forward"* and *"Go absolute"* commands must be a integer multiply of minimum microstep defined by the *"Microstep"* command.

The default setting is 4 microsteps per full step in Two DACs and Dual DAC modes of driver interface, i.e. microstep size is 16 minimum microsteps. With this setting distance or position used as argument of the *"Backward"*, *"Forward"* and *"Go absolute"* commands must be a integer multiply of 16.

EXAMPLE:

```
B640      ;move 10 full steps in negative direction
R         ;Run, move execution command - make a move
```

◆ DIRECTION

SYNTAX: **D**

The *"Direction"* command reverses next move direction.

EXAMPLE:

```
B640      ;move 10 full steps in negative direction
R         ;Run, move execution command - make a move
D         ;reverse next move direction to positive direction
R         ;Run, move execution command - make a move
```

◆ FORWARD

SYNTAX: **F (Num)** **RANGE:** 1 to 16 000 000

Move relative, specifies distance relative to current position in positive direction. The unit is minimum microstep = full step/64. The default setting is 64 000 minimum microsteps = 1 000 full steps = 5 revolutions for 1.8° stepper motor.

EXAMPLE:

```
F640      ;move 10 full steps in positive direction
R         ;Run, move execution command - make a move
```

◆ GO ABSOLUTE

SYNTAX: **G (Num)** **RANGE:** 1 to 16 000 000

Go to absolute position

Distance or position used as argument of the *"Backward"*, *"Forward"* and *"Go absolute"* commands must be a integer multiply of minimum microstep defined by the *"Microstep"* command.

The default setting is 4 microsteps per full step in Two DACs and Dual DAC modes of driver interface, i.e. microstep size is 16 minimum microsteps. With this setting distance or position used as argument of the *"Backward"*, *"Forward"* and *"Go absolute"* commands must be a integer multiply of 16.

EXAMPLE:

Current absolute position is 9 000 minimum microsteps. The following command:

G10000

specifies move distance as 1 000 minimum microsteps at positive direction.

◆ GO POSITIVE

SYNTAX: **G+**

Go continuously in positive direction, move at specified velocity until stopped by the *"Kill"* command or by tying the LIMIT input low.

◆ GO NEGATIVE

SYNTAX: **G-**

Go continuously in negative direction, move at specified velocity until stopped by the *"Kill"* command or by tying the LIMIT input low.

◆ HOME

SYNTAX: H

Go home, go to absolute position = 0

◆ EQUAL

SYNTAX: = (Num) **RANGE**: 1 to 16 000 000

Load position counter with specified value.

Position used as argument of the *"Equal"* commands must be a integer multiply of minimum microstep defined by the *"Microstep"* command. See the *"Microstep"* command.

Position counter value is not valid when motion defined by the *"Backward"*, *"Forward"* and *"Go absolute"* commands was stopped by the *"Kill"* command or by setting low the Limit input.

Position counter value is valid even if motion was stopped by the *"Kill"* command or by setting low the Limit input low on condition that this motion was defined by the *"Go positive"* or *"Go negative"* command.

Position counter contains 24-bit unsigned integer. It is advisable to set initial values in such a way that position counter values are not "negative" after any motion.

EXAMPLE:

=1200

New absolute position equals to 1 200 minimum microsteps.

◆ CLEAR POSITION COUNTER

SYNTAX: C100

Sets absolute position counter to 0.

◆ ACCELERATION

SYNTAX: **A (Num)** **RANGE:** 1 to 65 000 full steps/sec^2
1 to 128 000 full steps/sec^2
(M1486..2x with 24 MHz crystal)

The *"Acceleration"* command sets acceleration/deceleration in the range of 1 to 65 000 full steps/sec^2 at trapezoidal velocity profile. The default setting is 64 full steps/sec^2. The actual acceleration range is determined by the *"Start/stop"* command according to the following table:

START/STOP RANGE full steps/sec	ACCELERATION RANGE full steps/ sec^2
16 to 30	1 to 116
31 to 61	1 to 464
62 to 122	1 to 1 856
123 to 244	1 to 7 424
245 to 488	1 to 29 694
489 to 976	2 to 65 000
977 to 1 950	8 to 65 000

EXAMPLE:

A2000

Acceleration is 2 000 full steps/sec^2, 200 steps per revolution motor will increase its speed with the addition of 10 revolutions/sec every second at linear acceleration.

◆ START/STOP

SYNTAX: **S (Num)** **RANGE:** 16 to 1 950 full steps/sec

Sets start/stop (base) velocity. The default setting is 200 full steps/sec.

EXAMPLE:

S150

The base speed at which acceleration begin is 150 full steps/sec.

◆ PROFILE

SYNTAX: **P (Num)** **RANGE:** 20 to 7 000 full steps/sec

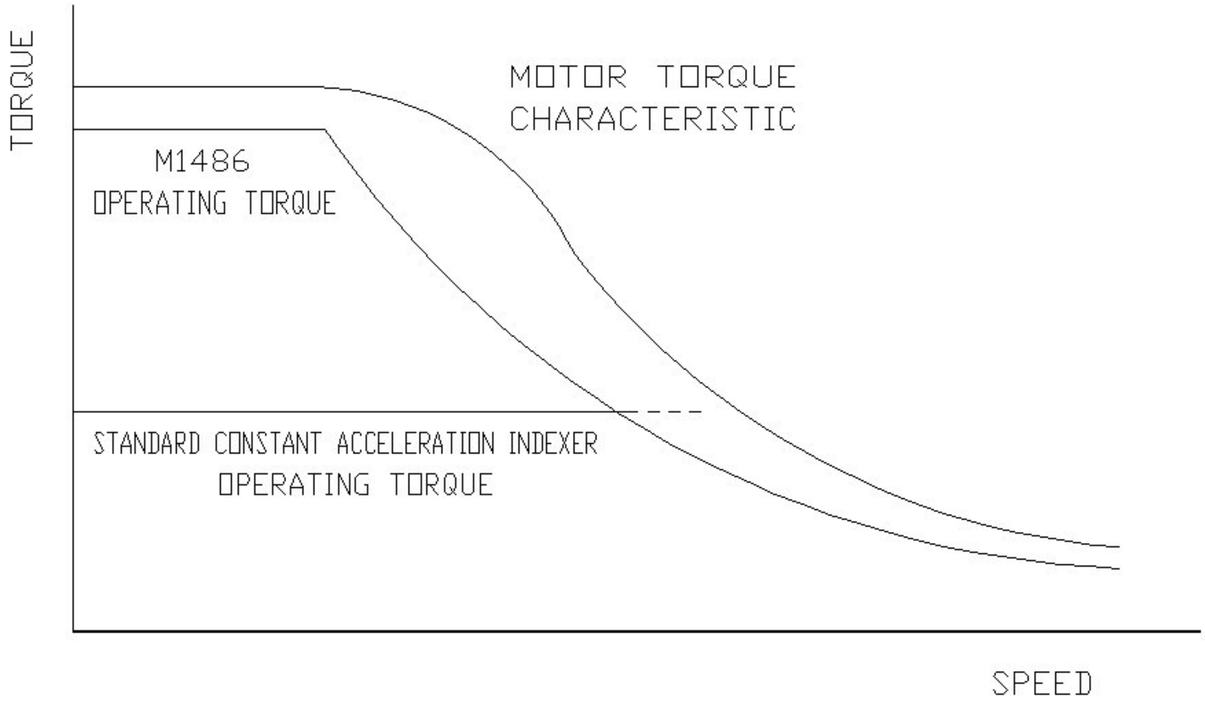
While most motion controllers available today use the trapezoidal velocity profile, this profile has serious drawbacks at a speed range where motor torque decreases. The *"Profile"* command enables the programmer to set a point where acceleration characteristic changes from linear to parabolic and back at deceleration. The user can set the *"Profile"* command data out of range to a value greater then maximum velocity to disable the parabolic acceleration option. The default setting is 25 000 full steps/sec.

EXAMPLE:

P2500

The M1486 uses linear acceleration up to speed of 2 500 full steps/sec, then continues with parabolic acceleration.

An example of acceleration characteristic for the motion program using the *"Profile"* command is shown in the following figure:



◆ VELOCITY

SYNTAX: **V (Num)**

RANGE: 16 to 20 000 full steps/sec standard (S&D mode)
16 to 40 000 full steps/sec M1486..2x with 24 MHz crystal (S&D mode)

Sets maximum velocity to be used in profile. The default setting is 1 000 full steps/sec. Maximum velocity specified by the *"Velocity"* command should be greater than start/stop velocity specified by the *"Start/Stop"* command.

EXAMPLE:

V12500

Specifies maximum velocity to 12 500 full steps/sec, acceleration can continue at the most to the velocity of 12 500 full steps/sec.

◆ RUN

SYNTAX: **R**

This is the move execution command. The *"Run"* command starts execution of motion. There are two ways to override current settings and stop motion with deceleration, the first one is sending the *"Kill"* command from the host computer, the second one is tying the LIMIT input low.

Motion parameters - speed, travel distance can be specified anywhere in command file. Lastly executed motion specification command specify respective motion parameter.

◆ SEEK POSITIVE

SYNTAX: **) (Num)** **RANGE:** 1 to 21

Seek limit in positive direction. The motor is moving in positive direction at start/stop speed until there is a low (inputs No. 1, 2, 3; 8, 9, 10, 11; 16, 17, 18, 19) or a high (inputs No. 4, 5, 6, 7; 12, 13, 14, 15; 20, 21) on the specified input. Absolute position counter is updated.

EXAMPLE:

)7 ; Go in positive direction at start/stop speed until there is a high on input No. 7.

◆ SEEK NEGATIVE

SYNTAX: **((Num)** **RANGE:** 1 to 21

Seek limit in negative direction. The motor is moving in negative direction at start/stop speed until there is a low (inputs No. 1, 2, 3; 8, 9, 10, 11; 16, 17, 18, 19) or a high (inputs No. 4, 5, 6, 7; 12, 13, 14, 15; 20, 21) on the specified input. Absolute position counter is updated.

EXAMPLE:

(2

Go in negative direction at start/stop speed until there is a low on input No. 2.

3 Commands for User I/O

C	*Clear*
O	*One*
T	*Turn on*
Z	*Zero*

Input/output lines B8 and B9 are available with Step & Direction and Dual DAC modes of the driver interface. Input/output lines B10 to B17 are available with Step & Direction mode of the driver interface only.

◆ CLEAR

SYNTAX: **C(Num)** **RANGE:** 1 to 21

Clear output, set selected user I/O bit low. The default setting of the user output bits is high.

EXAMPLE:

C7

Set user bit No. 7 low

◆ ONE

SYNTAX: **O(Num)** **RANGE:** 1 to 21

Wait for selected user I/O bit to go to logical one

EXAMPLE:

O2

The M1486 waits for user bit No. 2 to go high, then program execution continues from the following command.

◆ TURN ON

SYNTAX: **T(Num)** **RANGE:** 1 to 21

Turn on output, set selected user I/O bit high. Default setting of user outputs is high.

EXAMPLE:

T1

Set user bit No. 1 high

◆ ZERO

SYNTAX: **Z(Num)** **RANGE:** 1 to 21

Wait for selected user I/O bit to go to logical zero

EXAMPLE:

Z3

The M1486 waits for user bit No. 3 to go low, then program execution continues from the following command.

4 Sequencer Commands

M	*Microstep*
Q	*Qualification*
N	*Number of characteristic*

◆ MICROSTEP

SYNTAX: **M (Num)** **RANGE:** 1, 2, 4, 8, 16, 32, 64

The M1486 provides higher step resolution that virtually eliminates low speed resonance and step-to-step oscillation in stepper motor systems. The number of microsteps per full step can be programmed separately for low speed range and high speed range in order to achieve maximum speed as well as very high resolution at low speed.

The *"Microstep"* command selects microstep number for low speed range in the range from 1 to 64. The "Qualification" command selects microstep number for high speed range in the range from 1 to 4. M1486 controller at acceleration and deceleration automatically converts resolution at low range speed limit.

Low range speed limit - maximum speed when step division is defined by the *"Microstep"* is 12 000/(Num) [steps per second], where (Num) is selected number of microsteps per full step. The maximum speed is doubled for the M1486..2x with 24 MHz crystal.

The default setting is 4 microsteps per full step in Two Dacs and Dual DAC modes of driver interface, i.e. microstep size is 16 minimum microsteps. With this setting distance or position used as argument of the *"Backward"*, *"Forward"* and *"Go absolute"* commands must be a integer multiply of 16.

The default setting is 1 microsteps per full step in Step & Direction mode of driver interface, i.e. microstep size is 64 minimum microsteps. With this setting distance or position used as argument of the *"Backward"*, *"Forward"* and *"Go absolute"* commands must be a integer multiply of 64.

Available combinations of the *"Microstep"* and *"Configuration"* commands:

M64	Q1, Q2, Q4
M32	Q1, Q2, Q4
M16	Q4
M8	Q1, Q2
M4	Q1, Q4,
M2	Q2

◆ QUALIFICATION

SYNTAX: **Q (Num)** **RANGE:** 1, 2, 4

In order to boost high speed range while maintaining high resolution at low speed for low speed resonance suppression it is possible to program lower high speed resolution by the *"Qualification"* command. The range is from 1 to 4 microsteps per full step. The M1486 at acceleration and deceleration automatically converts resolution at low range speed limit.

Low range speed limit - maximum speed when step division is defined by the *"Microstep"* command is 12 000/(Num) [steps per second], where (Num) is selected number of microsteps per full step. The maximum speed is doubled for the M1486..2x with 24 MHz crystal.

Step division is defined by the *"Qualification"* command for speeds higher than low range speed limit.

Highest available speed at high speed range (step division defined by the *"Qualification"* command) is 3000 full steps per second when Q4 is selected and 6000 full steps per second when Q2 is selected. The highest available speed is doubled for the M1486..2x with 24 MHz crystal.

The default setting is 1 microstep per full step. Maximum speed when Q1 is selected is defined by the *"Velocity"* command.

◆ NUMBER OF CHARACTERISTIC

SYNTAX: **N (Num)** **RANGE:** 1 to 7

The user can choose motor current characteristic for smoothest rotation and/or best static positioning when the type of the motor and expected load torque are taken into consideration. The optimum characteristic varies from one application to another. The M1486 controller provides 7 alternative characteristics. Since suitable measuring equipment is seldom available, empirical methods which quickly reveal the useful range of characteristics are normally used. In general filled characteristics increase motor torque and trimmed down characteristics reduce motor noise. The following types of waveform shaping are implemented. The default setting is sinewave characteristic.

Num	Waveform shaping
1	Sinewave
2	Trapezoid wide
3	Trimming slight
4	Moderate
5	Triangle
6	Filling slight
7	Moderate

It is possible to run the motor in half step mode when N2 (Trapezoid wide) and M2 is selected.

5 Configuration Commands

\	*Reset*
X	*IndeX*

◆ RESET

SYNTAX: \

The "Reset" command for the M1486 controller, starts M1486 initiation. Internal registers are loaded with default values. Configuration settings are queried.

M1486A and M1486B controllers execute the *"Reset"* command immediately and other commands can continuously follow. The *"Reset"* command thus can be included in a command file which is send together.

Internal EEPROM is cleared when M1486E type controller receives the *"Reset"* command. It is necessary to wait one second for M1486E1 and two seconds for M1486E2 after the *"Reset"* command sending before another command can be sent. The *"Reset"* command thus can't be included in a command file which is send together.

◆ INDEX

SYNTAX: X (Num)

Type	**M1486A**	**M1486B**	**M1486E**
Range	0 to 9, A, B, C, D, E, F	0 to 9, A, B, C, D, E, F	1 to 9, A, B, C, D, E, F

Easily programmed via a serial link, up to sixteen M1486A or M1486B controllers (hexadecimal addresses from 0 to F) may be daisy-chained by setting up the address of each controller using pins ADDR0 to ADDR3 to allow simultaneous control of multiple axes.

Up to fifteen M1486E controllers (hexadecimal addresses from 1 to F) may be daisy-chained. If a low is held on all of the address inputs - ADDR0, ADDR1, ADDR2 and ADDR3 (hexadecimal address 0) a high on RST input resets the device. A high on RST input is always present at power-up due to recommended power-up 2,2 μF capacitor.

Address inputs ADDR0 to ADDR3 are queried after pulling high RST input only. Therefore corresponding controller pins can be used as outputs B18 - B21. Address inputs are not queried after software reset.

The selection of an indexer which is to receive following commands is made by the *"IndeX"* command. Selected indexer continues to receive commands until there is another *"IndeX"* command. *"XG"* selects all axis mode, following commands are for all axes.

The default setting is receiving enabled. The user needn´t care of address pins and the *"IndeX"* command for the single controller (point-to-point operation).

EXAMPLE:

Five controllers are daisy-chained, address settings selected by address inputs pins are from 0 to 4. The following command:

X3

disables any other command receiving by M1486 controllers with addresses 0, 1, 2 and 4. The M1486 with address 3 receives the following commands from the host computer.

◆ AUXILIARY FUNCTIONS

SYNTAX:	**T(Num)**	**RANGE:**	40 to 63
	C(Num)	**RANGE:**	40 to 63

It is possible to configure the controller to match a specific application using auxiliary functions. The auxiliary functions have two states only: ON and OFF.

The default setting of auxiliary functions No. 40 to 47 and 56 to 63 is OFF and the default setting of auxiliary functions 48 to 55 is ON.

40 Runsign

ON: Output pin B1 is held low when motor is running, high when motor is stopped.

OFF: B1 is general purpose input/output.

41 Return

ON: Every character received via serial channel is transmitted back to the host computer for verification.

OFF: The controller transmits only as directed by the *"Upload"* command.

42 Limitoff

ON: The limit input is disabled.

OFF: The limit input is enabled.

43 Edge

ON: The limit input is falling edge sensitive.

OFF: The limit input is low level sensitive.

44 Continue

ON: Program execution continues when wrong data received.

OFF: Program execution stops when wrong data are received.

45 Outputs

ON: Outputs held their values at shutdown.

OFF: All outputs are tied low.

46 Input

The *"Input"* function defines second operand of numeric operations. First operand is always user variable. See chapter Numeric unit.

ON: Second operand are data read on inputs specified by numeric command.

Controller inputs are specified by numeric command argument the same way as they are specified by the *"Query"* command.

OFF: Second operand is the argument of a numeric command.

Command "+14" increases user variable value by adding fourteen to the current user variable value.

47 Complement

Enables data read on specified inputs to be processed in true or inverted form.

ON: Data read on specified inputs are inverted prior to further processing.

OFF: Data read on specified inputs are not inverted prior to further processing.

48 Reserved for future use

49 Reserved for future use

50 Fullstep

ON: Motion continues after the end of forced deceleration till full step position is reached.

OFF: Immediate stop at the end of forced deceleration (caused by the *"Kill"* command or triggered limit input).

51 Recover

ON: Program execution continues after the *"Kill"* command or triggered limit input. Only currently executed motion is stopped.

OFF: Program execution is stopped after the *"Kill"* command or triggered limit input.

52 Fullstepss

ON: The *"Seek"* command stops the motor in a position corresponding to a multiple of 4 full steps.

OFF: The *"Seek"* command stops the motor in a microstep position according to step division defined by the *"Microstep"* command.

53 Reserved for future use

54 Reserved for future use

55 Reserved for future use

56 Reserved for future use

57 Report

The *"Report"* function enables to provide information to the host computer that command stored in input buffer have been compressed. It provides possibility to restart command transmission from the host computer to input buffer when input buffer is empty.

ON: If input buffer is empty (no character is stored in input buffer) ASCII character 07FH is transmitted via serial link.

If three bytes of input buffer become empty ASCII character 07EH is transmitted via serial link.

OFF: If input buffer is empty no character is transmitted via serial link.

58 Register

The *"Register"* function defines second operand of numeric operations. First operand is always user variable. See also chapter Numeric unit.

ON: Second operand are data stored in the register specified by the numeric command argument.

Command "+7" increases user variable value by adding register No. 7 to the user variable.

OFF: Second operand is the numeric command argument.
Command "+7" increases user variable value by adding seven to the user variable.

See also the auxiliary function No. 46 *"Input"*.

59 Remainder

The *"Remainder"* function defines data written on specified outputs at the *"Order"* command execution.

See also Numeric unit - the *"Order"* command

ON: Remainder after executed the *"Divide"* command is written on specified outputs at the *"Order"* command execution.

OFF: User variable value is written on specified outputs at the *"Order"* command execution.

60 Reserved for future use

61 Reserved for future use

62 Reserved for future use

63 Reserved for future use

EXAMPLE:

T51 Sets the *"Recover"* function ON
C43 Sets the *"Edge"* function OFF

◆ USER FLAG

SYNTAX:	**T(Num)**	**RANGE:**	80 to 86
	C(Num)	**RANGE:**	80 to 86

The user flags have two states only: ON - high and OFF - low. The default setting of user flags is OFF. User flags status can be queried by the *"Upload"* command (U14).

Program execution can be branched according to user flags status by the *"If"* command. User flags can be used as a memory of user inputs status.

EXAMPLE:

T85 Sets the user flag No. 85 ON
C85 Sets the user flag No. 85 OFF

◆ CARRY FLAG

The Carry flag (No. 87) is set automatically according the result of numeric operations Add and Subtract.

The Carry flag is set by hardware if the *"Subtract"* command operand is greater than user variable and cleared otherwise.

The Carry flag is set by hardware if the result of the *"Add"* command is equal or greater than 2^{24} and cleared otherwise.

The default setting of the Carry flag is OFF.

Program execution can be branched according to Carry flag status by the *"If"* command. This feature is useful for compare function implementation.

Carry flag status can be queried by the *"Upload"* command.

EXAMPLE:

```
[
@5
?14
-10
i87H5
r
]
```

The *"Run"* command is only executed if data read on inputs No. 1, 2, 3 and 4 are greater or equal to 10.

7 Numeric Unit

The M1486 provides numerical unit for basic arithmetic operations on internal 24-bit variable using 8-bit operands. The actual value of the internal variable may be used as argument of M1486 commands. Commands using internal variable as argument use simple syntax of the form: command, carriage return (ASCII character 0dH).

Please note no argument is used. No argument given implies the internal variable value should be used as argument for the respective command.

For user inputs binary value acquisition should be used the *"Query"* command and the *"Order"* command puts internal variable value in binary form on user outputs.

◆ LOAD

SYNTAX:	**:(Num)**	**RANGE:**	0 to 16 000 000

Load user variable with specified data.

◆ QUERY

SYNTAX:	**?(Num)(Num)**	**RANGE:**	(1 to 7)(1 to 7) (10 to 17)(1 to 8) (18 to 21)(1 to 4)

The last figure specifies number of user bits which are to be queried. The other figure (figures) specify low significant bit of queried used bits.

EXAMPLE:

?23 ;Query the value of three user inputs, starting at user input B2 and load it to the user variable.
;User inputs B2, B3 and B4 are queried, B2 being the LSB.
;For B2=0, B3=1 and B4=0 value '2' is loaded to user variable.

J ;The *"Jump"* command is specified without argument.
;The controller uses internal variable value as argument consequently and executes
;command **J2**.

◆ ORDER

SYNTAX: **!(Num)(Num)** **RANGE:** (1 to 7)(1 to 7)
(10 to 17)(1 to 8)
(18 to 21)(1 to 4)

Write internal variable value on outputs in binary form.

The last figure specifies number of user bits which are to be set or reset according to user variable value. The other figure (figures) specify low significant user bit to be written.

EXAMPLE:

```
!14     ;User variable value will be written to four user outputs B1 being LSB, i.e.
        ;outputs No. 1, 2, 3, 4.
```

For user variable value = 5 (2^3 = 0, 2^2 = 1, 2^1 = 0, 2^0 = 1) outputs will be set to 0101 according to the following table:

Output No.	B4	B3	B2	B1
Value after the *"Order"* command execution	0	1	0	1

Note

If the auxiliary function No. 59 *"Remainder"* is set ON Remainder after executed the *"Divide"* command is written (instead of user variable value) on specified outputs at the *"Order"* command execution. This option is useful for user variable output on four output pins in BCD format.

EXAMPLE:

User variable value is equal to 67. The following command file writes figure 7 to user outputs B1 to B4 and figure 6 to user outputs B18 to B21.

```
T59     ;Set the auxiliary function "Remainder" ON
/10     ;Divide internal variable by 10
!14     ;Write remainder value (7) to four user outputs B1 being LSB, i.e. outputs No. 1, 2, 3 and 4.
        ;(67 / 10 = 6, remainder is 7).
C59     ;Set the auxiliary function "Remainder" OFF
!184    ;Write user variable value (6) to four user outputs B18 being LSB, i.e. outputs No. 18, 19, 20
        ;and 21.
```

Registers

Numeric unit of M1486E and M1486B features 10 registers (Register No. 1 to 10).

Registers No. 1 to 8 are general purpose 16-bit registers. It is possible to move data to registers from user variable and to user variable from registers. The other possibility is to use their contents as operand for numeric operations provided the auxiliary function No. 58 *"Register"* is switched ON.

Register No. 9 is 24-bit position counter (see the command *"Equal"*). Only transfers with user variable are possible.

8-bit register No. 10 stores division remainder. Only transfers with user variable are possible.

Note
It is also possible to use registers with M1486A, but utilization is limited to applications which do not need last 16 bytes of command memory. There is shared memory space for command memory and registers in M1486A.

Command files with *"Reset"* as the first command and with lengths up to 736 bits may use registers without limitation on M1486A.

Register No. 9 can be used without limitation on M1486A, too.

◆ MOVE TO REGISTER

SYNTAX: **>(Num)** **RANGE:** 1 to 10

Move (copy) data from user variable to specified register.

◆ MOVE FROM REGISTER

SYNTAX: **<(Num)** **RANGE:** 1 to 10

Move (copy) data from specified register to user variable.

EXAMPLE:

<3

Data stored in register No. 3 are written to user variable. Data in register No. 3 remain unchanged.

Numeric operations

Numeric operation is executed with two operands, leaving the result in the internal variable. First operand is the internal variable.

As the second operand can be used:

1) simply number used as command argument

on condition that the "Input" and *"Register"* auxiliary functions are turned off (auxiliary functions No. 46 and 58).

See also Auxiliary functions description in chapter Auxiliary functions.

2) numerical value read on controller inputs which are specified by command argument

on condition that the auxiliary function No. 46 *"Input"* is turned on and the auxiliary function No. 58 *"Register"* is turned off.

Specification of controller inputs to be read is the same as input specification defined for the *"Query"* command.

The *"Add"* command argument for example is not added to the internal variable in the case that the auxiliary function No. 46 *"Input"* is turned on. Value read on inputs specified by the *"Add"* command argument is added instead. It can be used when reading up to 24-bits values using only small number of user inputs. It is possible to read BCD coded data, too.

3) content of the register which is specified by command argument

on condition that the auxiliary function No. 58 *"Register"* is turned on and the auxiliary function No. 46 *"Input"* is turned off.

Provided that the auxiliary function No. 58 *"Register"* is turned on, the second operand of numerical operations add and subtract is 16-bit value of the specified register.

Provided that the auxiliary function No. 58 *"Register"* is turned on, the second operand of numerical operations multiply and divide is lower byte of the specified register.

EXAMPLE:

T58	;The auxiliary function No. 58 *"Register"* is turned on
+5	;Add register No. 5 to internal variable
C58	;The auxiliary function No. 58 *"Register"* is turned off.
***2**	;Multiply internal variable by 2

◆ ADD

SYNTAX: **+(Num)** **RANGE:** 1 to 255

◆ SUBTRACT

SYNTAX: **-(Num)** **RANGE:** 1 to 255

◆ MULTIPLY

SYNTAX: ***(Num)** **RANGE:** 1 to 255

◆ DIVIDE

SYNTAX: **/(Num)** **RANGE:** 1 to 255

Note: Internal variable range is restricted to 65 000 only for the *"Divide"* command. The whole range of internal variable can be used only when dividing by the following numbers: 2, 4, 8, 16, 32, 64.

The division remainder is saved in register No. 10.

EXAMPLE 1:

***200**	;Multiply internal variable with specified number (200), load result to internal variable.
F	;The *"Forward"* command is specified without argument, its parameter is internal variable value.

EXAMPLE 2:

Internal variable is loaded with data in BCD format (range 0 to 99). High order of BCD data is read from user inputs B1, B2, B3 and B4 and low order of BCD data is read from user inputs B18, B19, B20 and B21.

Input data (BCD format) value is “74” in this example.

[	;Disable - delay execution of the following set of commands
?14	;Query data from user inputs B1 to B4. Internal variable value: 7
***10**	;Multiply by 10. Internal variable value: 70
t46	;The auxiliary function *"Input"*+ (No. 46) is turned on. Data on specified inputs will be added to
	;the internal variable value at the *"Add"* command execution.
+184	;Add data queried from user inputs B18 to B21 to internal variable value.
	;Internal variable value after addition: 74
a	;The *"Acceleration"* command is specified without argument. The controller executes command
	;A74 - set velocity to 74 steps/sec^2
]	;Enable - enable execution of the delayed commands, program execution
	;at the the command following the *"Disable"* command - *"Start/stop"*

Note: The *"Input"* function defines second operand of numeric operations. First operand is always user variable.

If the auxiliary function *"Input"* is switched on then second operand are data read on inputs specified by numeric command argument.

The user can load data up to 24-bit wide to the internal variable using a few user bits with help of this function. BCD coded data can be loaded too, see example No. 5.

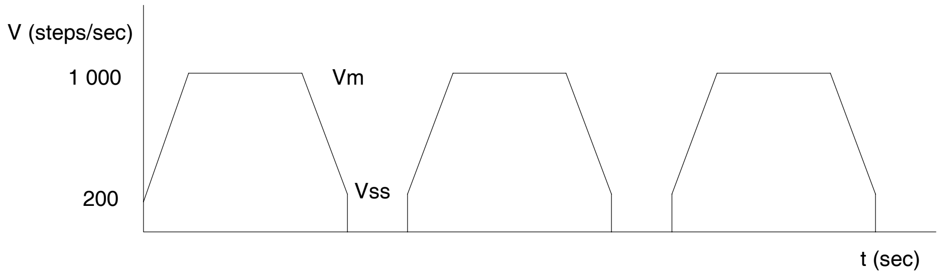
VIII Motion program examples

Motion program example No. 1

```
\           ;Reset - default settings are loaded (Default settings are also automatically loaded at
            power up)
            ;The "Reset" command can't be included in a command file which is send together to
            ;M1486E type controller, the following commands can be send after EEPROM memory
            ;erasure is finished. See the "Reset" command description.
f12800      ;Forward - 200 full steps - one revenue for 1.8° step motors (12800 minimum
            ;microsteps) in positive direction
r           ;Run - make a move of 200 full steps with default settings of start/stop velocity,
            ;maximum velocity, full step division and acceleration
```

Motion program example No. 2

```
[           ;Disable - delay execution of the following set of commands
S200        ;Start/stop velocity set to 200 steps/sec
V1000       ;Maximum velocity set to 1000 steps/sec
A5000       ;Acceleration - set to 5000 full steps/sec^2
F39616      ;Forward distance - 3150 steps in positive direction
L3          ;Loop - repeat the following set of commands 3 times
R           ;Run
W250        ;Wait - wait 500 milliseconds
E           ;End of loop
]           ;Enable - enable execution of the delayed commands, program execution
            ;at the command following the "Disable" command – "Start/stop"
```



Motion program example No. 3

The following motion program indicates the possibility to change motion parameters using the *"If"* command and user pins without any host computer intervention. This feature is very useful for stand alone operation.

```
[
@5
I2H2
I1H1
V370
J6
;
@1
V1230
J6
;
@2
I1H3
V1450
J6
;
@3
V1920
;
@6
R
J5
]
```

As will be seen from the table below, user input status selects maximum velocity of the motion.

User input No.		**Velocity**
1	**2**	**steps/sec**
0	0	370
1	0	1 230
0	1	1 450
1	1	1 920

Motion program example No. 4

Motion program No. 3 could become much simpler using numeric unit. User inputs No. 1 and 2 data loaded into internal variable are used as the *"Jump"* command argument. The command *"If"* is eliminated and motion program structure is streamlined.

```
[
@5
?12             ;Query binary data from user inputs B1 and B2.  Internal variable value is one of the
                ;following: 0, 1, 2, 3 according to user inputs B1 and B2.
+1              ;Add 1 to user variable. Internal variable value one of the following 1, 2, 3 and 4.
J               ;The "Jump" command is specified without argument, its parameter is internal variable
                ;value.
@1
V370
J6
;
@2
V1230
J6
;
@3
V1450
J6
;
@4
V1920
;
@6
R
J5
]
```

Whole part (not only velocity) of motion program which is specific for given type of product manufactured using M1486 type of controller can be specified on selected label.

By this way type of product can be selected for example using external thumbwheel switches or using master PLC and different parts of motion program can be executed for various products.

It is possible to specify up to 8 types of products with three inputs, up to 16 types with four inputs.

Motion program example No. 5

Another example for stand alone-operation without any host computer intervention using user I/O and internal variable.

Internal variable is loaded with data in BCD format (range 0 to 99). External quad 2-input data selector/multiplexer (74xxx157) selects four bits of data from two sources under the control of a select input (S). Input data (BCD format) value is ´83´ in this example.

```
[             ;Disable - delay execution of the following set of commands
?14           ;Query data from user inputs B1 to B4. Internal variable value: 8
*10           ;Multiply by 10. Internal variable value: 80
c8            ;User I/O B8 is connected to select input of an external multiplexer. Select low
              ;order of BCD data
t46           ;The auxiliary function "Input"+ (No. 46) is turned on. Data on specified inputs will be
              ;added to the internal variable value at the "Add" command execution.
+14           ;Add data queried from user inputs B1 to B4 to internal variable value. Internal
              ;variable value after addition: 83
a             ;The "Acceleration" command is specified without argument. The controller executes
              ;command A83 -  set acceleration at 83 steps/sec2
]             ;Enable - enable execution of the delayed commands, program execution
              ;at the command following the "Disable" command - "Start/stop"
```

Using two user I/O for data selection end external quad 4-input data selector/multiplexer (2x 74xxx153 or 2x 74xxx257) an amount of input data in BCD format is increased to the range from 0 to 9 999.

Using 16 user inputs it is possible to read data in the range from 0 to 9 999 without external multiplexer.

Motion program example No. 6

```
x0          ;command sequence for No. 0 controller
:0          ;set internal variable to zero
l20         ;Loop - repeat following commands twenty times
[           ;Disable command - delay following commands execution till the Enable command
            ;reception
c1          ;Clear - set user output No. 1 low
t1          ;Turn On - set user output No. 1 high
w2000       ;wait for 2 sec
+64         ;add to internal variable value 64 (64 microsteps = one full step)
f           ;set distance is equal to internal variable value
r           ;Run - make move with actual parameter values
e           ;End of loop
]           ;Enable - start preceding commands execution
;
x7          ;controller No. 7 command sequence
[
s70         ;Start/stop - initial speed 70 full steps/sec
a1000       ;Acceleration 1000 full steps/sec²
f25600      ;Forward - distance 400 full steps in positive direction
@2          ;Address - assigns label number 2 to current program line
z3          ;Zero - wait for user input No. 3 to go to low
r           ;Run - make move with actual parameter values
d           ;Direction - reverse next move direction
j2          ;Jump - continue with command following to "@2" command
]
```

Controller No. 0 output No. 1 is connected to user input No. 3 of controller No. 7.

IX M1486 Application Development Support

1 Inmotion PC Utilities Software

Inmotion PC Utilities software is available for M1486 indexer and microstepping sequencer program generation, editing and validation. As a learning tool, the software filters each command for proper format and parameter values. The software communicates with the stepper controller via serial interface COM1, COM2, COM3 or COM4 of the PC.

Part of this software is the motion calculator which provides the possibility to calculate any motion parameter after remaining motion parameters specification. For example, motion calculator gives total time of movement and times of individual parts of movement after start/stop velocity, maximum velocity, acceleration and distance specification.

The program can also be controlled using a mouse. The function "Keys" are used for frequently used commands, for example open file, close file and help. All dialog boxes that require the user to enter text keep a scrolling history list, so you can reuse previous entries. Furthermore, Inmotion provides integrated calculator, ASCII table, and directory browser.

Inmotion PC Utilities enables programs to be created off-line on a PC and then downloaded to the controller as well as motion commands to be sent immediately after typing.

2 CD30M Development Board

Development board CD30M is a system design board dedicated to the Microcon M1486 family of stepper motor controllers. The CD30M board is equipped with all necessary components for stepper motors driving. On board advanced microstepping pulse width modulated power drivers feature up to 3,3 amps/phase at supply voltages up to 48 volts. A thermal protection circuit is included. The drivers are designed for high efficiency and reliability. Automatic standby reduces motor current when motor is not moving.

To complete stepper motor system the user needs to add standard two phase stepper motors only.

The CD30M board can be directly linked to IBM PC or any host equipped with RS232 serial interface. Unlimited number of host controller step by step commands can be sent to the M1486 due to fully scrollable internal memory.

The board can be use in stand alone applications. For stand alone operation internal memory enables pre-programmed move sequences to be repeated off line. The program stored in memory can be easily changed after re-connecting of your computer. In addition pre-programmed motion & machine control sequences may be stored in an on-chip EEPROM (M1486E). These sequences can be automatically executed on power-up.

The board provides an easy to use flexible instrument which enables the user to build a controller for specific application. On board wire-wrap area can support user specific hardware for prototyping. User specified board modifications for volume requirements are available.

Please consult the Microcon sales office to confirm availability of full spectrum of boards and modules supporting M1486 controller.

X M1486 Family Members

Unless otherwise stated all data in this manual refer to the M1486A Stepper Motor Controller.

M1486A

96 bits input buffer
RAM command memory 864 bits, speed up to 20 000 steps/sec (up to 40 000 steps/sec – M1486A..2x)

M1486A2x

Identical with M1486, in addition it can run also with 24 MHz crystal. The controller runs two times faster in that case.

It has impact especially on serial link speed as well as commands specifying speed, acceleration and dwell times.

For example *"W2000"* command specifies dwell time 1 000 ms only (the controller runs faster) and *"V2000"* specifies 4 000 steps/sec velocity.

M1486B

Pin and function compatible with M1486A
80 labels, 256 bits input buffer, RAM command memory 2 032 bits

M1486C

Pin and function compatible with M1486A
RAM command memory 4 000 bits, four integrated 8-bit PWM outputs

M1486E1

Pin and function compatible with M1486A, 80 labels, 256 bits input buffer

On-chip 2 000 bits EEPROM command memory. Command files beginning with the *"Disable"* command and with the *"Enable"* command as the last one are stored in the EEPROM are automatically executed on power-up without battery backup.

A high on RST input resets the device on condition that a low is held on all of the address inputs - ADDR0, ADDR1, ADDR2 and ADDR3.

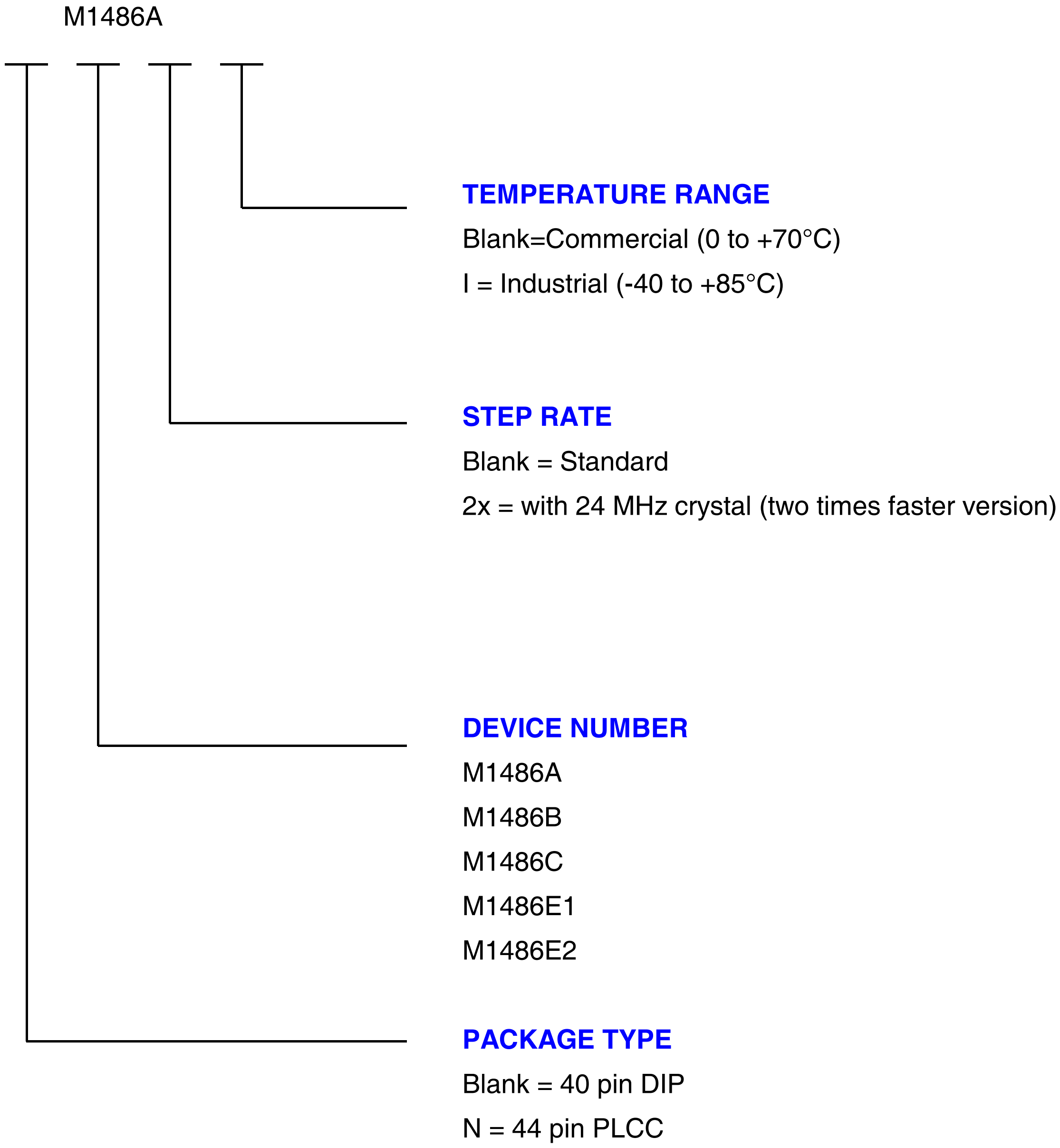
If any of the address inputs is held high then a high on RST input stops command file execution and command file execution starts again from the beginning.

M1486E2

Pin and function compatible with M1486E1
On-chip 4 000 bits EEPROM command memory

All types of the M1486 controller are also delivered in two times faster versions, that can use 24 MHz crystal.

XI Ordering Information



M1486A

TEMPERATURE RANGE

Blank=Commercial (0 to +70°C)

I = Industrial (-40 to +85°C)

STEP RATE

Blank = Standard

2x = with 24 MHz crystal (two times faster version)

DEVICE NUMBER

M1486A

M1486B

M1486C

M1486E1

M1486E2

PACKAGE TYPE

Blank = 40 pin DIP

N = 44 pin PLCC

Valid Combinations

Consult the Microcon sales office to confirm availability of specific combinations and to check on newly released options.

XII Application Hints

◆ Serial link

Please note that inputs BR0, BR1 define baud rate at power-up. They can be used as general purpose input/outputs B5, B6 after power-up. It is necessary to determine B5, B6 levels at power-up to establish correct serial communication speed if they are used as inputs.

◆ Limited input command buffer size

Easiest way to overcome input buffer limitations is to use the *"Disable"* command at the beginning of command file and the *"Enable"* command at the end of command file. Commands will go easily through input buffer to command memory unrestricted by the input buffer size. Command file length is limited only by command memory size.

Command file closed in square brackets can´t include the *"Reset"* command. Otherwise it would delete the *"Disable"* command.

◆ Speed and travel distance units

Speed is always defined at full steps/sec regardless of step division specified by the *"Microstep"* command.

Travel distance is always defined at minimum microsteps (full step/64) regardless of step division specified by the *"Microstep"* command.

EXAMPLE:

F64 ; Forward 64 specifies always one full step travel distance in positive direction.

◆ Acceleration range

Acceleration range is defined according to specified start/stop speed (see table on page 27). Make sure not to get out of the range at acceleration specification. While updating start/stop specification make sure that specified acceleration value is in the range corresponding to new start/stop speed.

◆ The "Reset" command

No M1486E command file can include the *"Reset"* command. Due to time interval necessary for EEPROM erase commands following after the *"Reset"* command may not be received correctly.

Notes

Notes

Notes