Dobot Magician ALARM Description

issue：V1.0.0

Date：2018-12-07

Shenzhen Yuejiang Technology Co., Ltd

Disclaimer

To the maximum extent permitted by applicable law, the products described (including its hardware, software and firmware, etc.) in this document are provided AS IS, which may have flaws, errors or faults. Yuejiang makes no warranties of any kind, express or implied, including but not limited to, merchantability, satisfaction of quality, fitness for a particular purpose and non-infringement of third party rights. In no event will Yuejiang be liable for any special, incidental, consequential or indirect damages resulting from the use of our products and documents.

Before using our product, please thoroughly read and understand the contents of this document and related technical documents that are published online, to ensure that the robotic arm is used on the premise of fully understanding the robotic arm and related knowledge. Please use this document with technical guidance from professionals. Even if follow this document or any other related instructions, Damages or losses will be happen in the using process, Dobot shall not be considered as a guarantee regarding to all security information contained in this document.

The user has the responsibility to make sure following the relevant practical laws and regulations of the country, in order that there is no significant danger in the use of the robotic arm.

Shenzhen Yuejiang Technology Co., Ltd

Address: 3F, Building NO.3, Tongfuyu Industrial Town, Nanshan District, Shenzhen, China

Website: www.dobot.cc

Preface

Purpose

This document describes the alarms of Dobot Magician and provides the solutions to clear alarm.

Intended Audience

This document is intended for:

- Customer Engineer
- Installation and Commissioning Engineer
- Technical Support Engineer

Change History

Date	Description
2018/11/09	Added over speed alarm, lost step alarm
2016/11/30	Created the document

Contents

1. ALARM Description

1.1 Getting the Alarm Status of System

Table 1.1 The description of getting system alarm

Prototype	int GetAlarmsState(uint8_t *alarmsState, uint32_t *len, unsigned int maxLen)
Description	Get the alarm status of system
Parameter	alarmsState: The first address of the array. Each byte in the array **alarmsState** identifies the alarms status of the eight alarm items, with the MSB (Most Significant Bit) at the top and LSB (Least Significant Bit) at the bottom len: The byte occupied by the alarm maxLen: Maximum array length, to avoid overflow
return	DobotCommunicate_NoError: The command returns with no error DobotCommunicate_Timeout: The command does not return, resulting in a timeout

📖NOTE

Each byte in the array **alarmsState** identifies the alarms status of the eight alarm items, with the MSB (Most Significant Bit) at the top and LSB (Least Significant Bit) at the bottom.

1.2 Clearing All Alarms

Table 1.2 Clear all alarms

Prototype	int ClearAllAlarmsState(void)
Description	Clear all alarms
Parameter	None
return	DobotCommunicate_NoError: The command returns a value with no error DobotCommunicate_Timeout: The command does not return any value, resulting in a timeout

1.3 Alarm Content

```
enum {
    // Common error
    ERR_COMMON_MIN = 0x00,
    ERR_COMMON_RESET = ERR_COMMON_MIN,

    ERR_COMMON_MAX = 0x0f,

    // Plan error
    ERR_PLAN_MIN = 0x10,
    ERR_PLAN_INV_SINGULARITY = ERR_PLAN_MIN,
```



```
ERR_PLAN_INV_CALC,
ERR_PLAN_INV_LIMIT,
ERR_PLAN_PUSH_DATA_REPEAT,
ERR_PLAN_ARC_INPUT_PARAM,
ERR_PLAN_JUMP_PARAM,

ERR_PLAN_MAX = 0x1f,

// Move error
ERR_MOVE_MIN = 0x20,
ERR_MOVE_INV_SINGULARITY = ERR_MOVE_MIN,
ERR_MOVE_INV_CALC,
ERR_MOVE_INV_LIMIT,

ERR_MOVE_MAX = 0x2f,

// Over speed error
ERR_OVERSPEED_MIN = 0x30,
ERR_OVERSPEED_AXIS1 = ERR_OVERSPEED_MIN,
ERR_OVERSPEED_AXIS2,
ERR_OVERSPEED_AXIS3,
ERR_OVERSPEED_AXIS4,

ERR_OVERSPEED_MAX = 0x3f,

// Limit error
ERR_LIMIT_MIN = 0x40,
ERR_LIMIT_AXIS1_POS = ERR_LIMIT_MIN,
ERR_LIMIT_AXIS1_NEG,

ERR_LIMIT_AXIS2_POS,
ERR_LIMIT_AXIS2_NEG,

ERR_LIMIT_AXIS3_POS,
ERR_LIMIT_AXIS3_NEG,

ERR_LIMIT_AXIS4_POS,
```



```
    ERR_LIMIT_AXIS4_NEG,

    ERR_LIMIT_AXIS23_POS,
    ERR_LIMIT_AXIS23_NEG,

    //ERR_LIMIT_SINGULARITY,

    ERR_LIMIT_MAX = 0x4f,

    // Lose Step error
    ERR_LOSE_STEP_MIN = 0x50,
    ERR_LOSE_STEP_AXIS1 = ERR_LOSE_STEP_MIN,
    ERR_LOSE_STEP_AXIS2,

    ERR_LOSE_STEP_AXIS3,
    ERR_LOSE_STEP_AXIS4,

    ERR_LOSE_STEP_MAX = 0x5f,
```

1.4 Alarm Index Calculation Description

Table 1.3 Alarm Index Calculation description

Alarm command	0x AA AA 02 14 00 EC Protocol command consists of packet header, payload length, payload frame, and check. AAAA: Packet header 02: Payload length 14: ID Payload ID 00: Payload data(read\write status and queue command status) EC: Check
Returned command	0x AA AA 12 14 00 01 00 00 00 00 00 00 00 08 00 00 00 00 00 00 00 EC
Analysis process	In the returned command, 01 00 00 00 00 00 00 00 08 00 00 00 00 00 00 00 is an array with 16 bytes. One byte identifies eight alarm items, and the payload frame uses little endian mode, we should calculate alarm index by little endian mode Binary : 0x 00000000…...00001000……00000001 When alarm item is 1, this means there is an alarm. So we can calculate that the first alarm is at the 0^{th} bit and the second alarm is at the 67^{th} bit. Transform decimal to hex: 0x00, 0x43
Analysis result	0x00: Reset alarm 0x43: Joint 2 negative limitation alarm

2. Public Alarm

2.1 Resetting Alarm

Table 2.1 Resetting alarm

Index	0x00
Trigger condition	The alarm will be triggered after resetting system
Reset condition	The protocol instruction is cleared

Description

The alarm will be triggered after resetting system.

Reason

The alarm is triggered because of the resetting of system.

Solution

Click **Script** module on DobotStudio interface, execute script **ClearAllAlarmsState** to reset alarm.

📖NOTE

For the Script module details, please see the chapter **Scripting** in **Dobot Maigican User Guide**.

2.2 Undefined Instruction Alarm

Table 2.2 Undefined instruction alarm

Index	0x01
Trigger condition	Receive undefined instruction
Reset condition	The protocol instruction is cleared

Description

Receive undefined instruction

Reason

Receive undefined instruction. For example, an instruction with an error ID which is an undefined instruction.

Solution

Click **Script** module on DobotStudio interface, execute script **ClearAllAlarmsState** to reset alarm.

2.3 File System Alarm

Table 2.3 File system alarm

Index	0x02
Trigger condition	File system error
Reset condition	Reset system, if file system is initialized successfully, the alarm is cleared automatically

2.4 Communication Alarm Between MCU and FPGA

Table 2.4 Communication alarm between MCU and FPGA

Index	0x03
Trigger condition	There is a failed communication between MCU and FPGA when system is initializing
Reset condition	Reset system, if the communication is successful, the alarm is cleared automatically

2.5 Angle Sensor Alarm

Table 2.5 Angle Sensor Alarm

Index	0x04
Trigger condition	Get an error value of angle sensor
Reset condition	Re-power Dobot Magician，if the value is correct, that indicates reset alarm successfully

3. Planning Alarm

3.1 Inverse Resolve Abnormal Alarm

Table 3.1 Inverse resolve abnormal alarm

Index	0x10
Trigger condition	The target point is in abnormal position in Cartesian coordinate system
Reset condition	The protocol instruction is cleared

📖NOTE

Inverse resolve is to get joint angle according to the position of Dobot Magician.

Description

The target point (beginning point or ending point) is in abnormal position in Cartesian coordinate system.

Reason

- The target point is in abnormal position in MOVL mode.
- The target point or middle point is in abnormal position in ARC mode.
- The target point is in abnormal position in CP mode.
- The target point is in abnormal position in JUMP_MOVL mode.

Solution

1） Check the target point whether is abnormal, teach and save point again

2） Click **Script** module on DobotStudio interface, execute script **ClearAllAlarmsState** to reset alarm.

3.2 Inverse Resolve Alarm

Table 3.2 Inverse resolve alarm

Index	0x11
Trigger condition	The target point is out of the workspace, that causes inverse resolve alarm
Reset condition	The protocol instruction is cleared

Description

The target point is out of the workspace, which causes inverse resolve alarm.

Reason

The target point is out of the workplace in all motion mode.

Solution

1） Check the target point whether is abnormal, teach and save point again. For the workplace of Dobot Magician detail, please see Dobot Magician User Guide > Introduction > workplace.

2） Click Script module on DobotStudio interface, execute script ClearAllAlarmsState to reset alarm.

3.3 Inverse Resolve Limitation Alarm

Table 3.3 Inverse resolve limitation alarm

Index	0x12
Trigger condition	The inverse resolve of target is out of the limitation
Reset condition	The protocol instruction is cleared

Description

The inverse resolve of target point is out of the limitation.

Reason

The target point is out of the workplace in all motion mode.

Solution

1） Check the target point whether is out of the limitation, teach and save point again.

Table 3.4 The description of limitation

轴 Joint	负限位 Negative limitation	正限位 Positive limitation
J1	-90°	90°
J2	0°	85°
J3	-10°	90°
J4	-90°	90°

2） Click **Script** module on DobotStudio interface, execute script **ClearAllAlarmsState** to reset alarm.

3.4 Data Repetition Alarm

Table 3.5 Data repetition alarm

Index	0x13
Trigger condition	There are some repetitive points in ARC or JUMP_MOVEL mode
Reset condition	The protocol instruction is cleared

Description

There are some repetitive points in ARC or JUMP_MOVEL mode.

Reason

- Because of two points or three points are repetitive, that can't constitute ARC in ARC mode.
- JThe starting point and the ending point are same in JUMP_MOVL.

Solution

1） Check the teaching point whether are repetitive, teach and save point again.

2） Click **Script** module on DobotStudio interface, execute script **ClearAllAlarmsState** to reset alarm.

3.5 JUMP Parameter Alarm

Table 3.6 Jump parameter alarm

Index	0x15
Trigger condition	Set a wrong parameter in JUMP mode
Reset condition	The protocol instruction is cleared

Description

Set a wrong parameter in JUMP mode.

Reason

Height is a negative value.

Solution

1） Check JUMP parameter and set it again.

2） Click **Script** module on DobotStudio interface, execute script **ClearAllAlarmsState** to reset alarm.

4. Motion Alarm

4.1 Inverse Resolve Singularity Alarm

Table 4.1 Inverse resolve singularity alarm

Index	0x20
Trigger condition	Motion trajectory is in singularity position that causes inverse resolve alarm in Cartesian coordinate system
Reset condition	The protocol instruction is cleared

Description

Motion trajectory is in singularity position that causes inverse resolve alarm in Cartesian coordinate system

Reason

- Move to singularity position in Cartesian coordinate system.
- Move to singularity position in MOVL mode.
- Move to singularity position in ARC mode.
- Move to singularity position in CP mode.
- Move to singularity position in JUMP_MOVL.

Solution

- When move to abnormal position in Cartesian coordinate system, you can move point to the positive direction.
- Check whether the motion trajectory is abnormal, teach and save point again.

4.2 Inverse Resolve Alarm

Table 4.2 Inverse resolve alarm

Index	0x21
Trigger condition	Motion is out of workspace that causes inverse resolve alarm when Dobot Magician moves
Reset condition	The protocol instruction is cleared

Description

Motion is out of workspace that causes inverse resolve alarm when Dobot Magician moves.

Reason

Motion is out of the workplace in all motion mode.

Solution

1） Check whether motion is out of the workspace when Dobot Magician moves, teach and save point again.

2） Click **Script** module on DobotStudio interface, execute script **ClearAllAlarmsState** to reset alarm.

4.3 Inverse Resolve Limitation Alarm

Table 4.3 Inverse resolve limitation alarm

Index	0x22
Trigger condition	The motion is out of the limitation when Dobot Maigican moves
Reset condition	The protocol instruction is cleared

Description

The inverse resolve is out of the limitation when Dobot Magician moves

Reason

The motion is out of the limitation in all motion modes.

Solution

1） Check whether motion is out of the limitation in motion process, teach and save point again

2） Click **Script** module on DobotStudio interface, execute script **ClearAllAlarmsState** to reset alarm.

5. Overspeed Alarm

5.1 Joint 1 Overspeed

Table 5.1 Joint 1 overspeed alarm

Index	0x30
Trigger condition	Joint 1 is overspeed
Reset condition	The protocol instruction is cleared

Description

The joint 1 is overspeed when Dobot Magician moves.

Reason

The speed of motor is bigger than the max value in Cartesian motion. Such as MOVL, ARC and so on.

Solution

1） Decrease speed ratio to make joint speed smaller than the max value.

2） Click **Script** module on DobotStudio interface, execute script **ClearAllAlarmsState** to reset alarm.

5.2 Joint 2 Overspeed

Table 5.2 Joint 2 overspeed alarm

Index	0x31
Trigger condition	Joint 2 is overspeed
Reset condition	The protocol instruction is cleared

Description

The joint 2 is overspeed when Dobot Magician moves.

Reason

The speed of motor is bigger than the max value in Cartesian motion. Such as MOVL, ARC and so on.

Solution

1） Decrease speed ratio to make joint speed smaller than the max value.

2） Click **Script** module on DobotStudio interface, execute script **ClearAllAlarmsState** to reset alarm.

5.3 Joint 3 Overspeed

Table 5.3 Joint 3 overspeed alarm

Index	0x32
Trigger condition	Joint 3 is overspeed
Reset condition	The protocol instruction is cleared

Description

The joint 3 is overspeed in running process.

Reason

The speed of motor is bigger than the max value in Cartesian motion. Such as MOVL, ARC and so on.

Solution

1） Decrease speed ratio to make joint speed smaller than the max value.

2） Click **Script** module on DobotStudio interface, execute script **ClearAllAlarmsState** to reset alarm.

5.4 Joint 4 Overspeed

Table 5.4 Joint 4 overspeed alarm

Index	0x33
Trigger condition	Joint 4 is overspeed
Reset condition	The protocol instruction is cleared

Description

The joint 4 overspeed.

Reason

The speed of motor is bigger than the max value in Cartesian motion. Such as MOVL, ARC and so on.

Solution

1） Decrease speed ratio to make joint speed smaller than the max value.

2） Click **Script** module on DobotStudio interface, execute script **ClearAllAlarmsState** to reset alarm.

6. Limit Alarm

6.1 Joint 1 Positive Limitation Alarm

Table 6.1 Joint 1 positive limitation alarm

Index	0x40
Trigger condition	Joint 1 moves to the positive limitation area
Reset condition	Jog the Joint1 coordinate towards the opposite direction

6.2 Joint 1 Negative Limitation Alarm

Table 6.2 Joint 1 negative limitation alarm

Index	0x41
Trigger condition	Joint 1 moves to the negative limitation area
Reset condition	Jog the Joint1 coordinate towards the opposite direction

6.3 Joint 2 Positive Limitation Alarm

Table 6.3 Joint 2 positive limitation alarm

Index	0x42
Trigger condition	Joint 2 moves to the positive limitation area
Reset condition	Jog the Joint2 coordinate towards the opposite direction

6.4 Joint 2 Negative Limitation Alarm

Table 6.4 Joint 2 negative limitation alarm

Index	0x43
Trigger condition	Joint 2 moves to the negative limitation area
Reset condition	Jog the Joint2 coordinate towards the opposite direction

6.5 Joint 3 Positive Limitation Alarm

Table 6.5 Joint 3 positive limitation alarm

Index	0x44
Trigger condition	Joint 3 moves to the positive limitation area
Reset condition	Jog the Joint3 coordinate towards the opposite direction

6.6 Joint 3 Negative Limitation Alarm

Table 6.6 Joint 3 negative limitation alarm

Index	0x45
Trigger condition	Joint 3 moves to the negative limitation area
Reset condition	Jog the Joint3 coordinate towards the opposite direction

6.7 Joint 4 Positive Limitation Alarm

Table 6.7 Joint 4 positive limitation alarm

Index	0x46
Trigger condition	Joint 4 moves to the positive limitation area
Reset condition	Jog the Joint4 coordinate towards the opposite direction

6.8 Joint 4 Negative Limitation Alarm

Table 6.8 Joint 4 negative limitation alarm

Index	0x47
Trigger condition	Joint 4 moved to the negative limitation area
Reset condition	Jog the Joint4 coordinate towards the opposite direction

6.9 Parallelogram Positive Limitation Alarm

Table 6.9 Parallelogram positive limitation alarm

Index	0x48
Trigger condition	Parallelogram is stretched to the limitation area
Reset condition	Quit the limitation area

6.10 Parallelogram Negative Limitation Alarm

Table 6.10 Parallelogram negative limitation alarm

Index	0x49
Trigger condition	Parallelogram is stretched to the limitation area
Reset condition	Quit the limitation area

7. Losing-Step Alarm

7.1 Joint 1 Losing-Step

Table 7.1 Joint 1 Losing-Step

Index	0x50
Trigger condition	Joint 1 loses step
Reset condition	Execute the homing function

Description

The joint 1 loses step when Dobot Magician moves

Reason

Dobot Magician is hit in running process.

Solution

Execute the homing function of Dobot Magician.

7.2 Joint 2 Losing-Step

Table 7.2 Joint 2 losing-step

Index	0x51
Trigger condition	Joint 2 loses step
Reset condition	Execute the homing function

Description

The joint 2 loses step in running process

Reason

Dobot Magician is hit in running process.

Solution

Execute the homing function of Dobot Magician.

7.3 Joint 3 Losing-Step

Table 7.3 Joint 3 losing-step

Index	0x52
Trigger condition	Joint 3 loses step
Reset condition	Execute the homing function

Description

The joint 3 loses step in running process.

Reason

Dobot Magician is hit in running process.

Solution

Execute the homing function of Dobot Magician.

7.4 Joint 4 Losing-Step

Table 7.4 Joint 4 Losing-Step

Index	0x53
Trigger condition	Joint 4 loses step
Reset condition	Execute the homing function

Description

The joint 4 loses step in running process.

Reason

Dobot Magician is hit in running process.

Solution

Execute the homing function of Dobot Magician.

