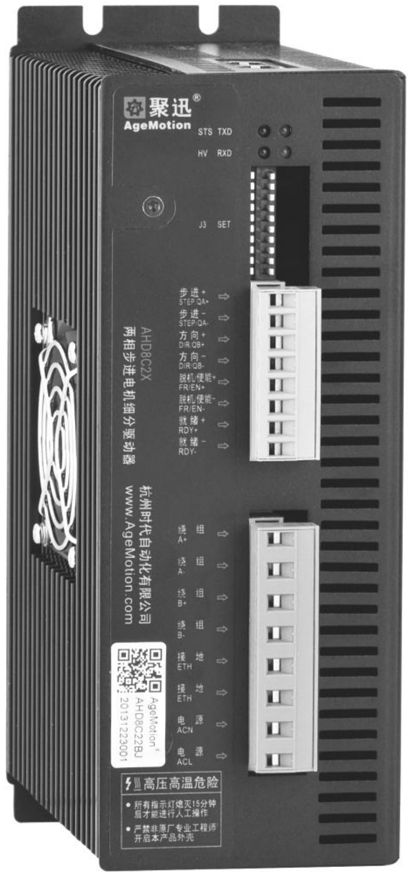
AHD8C2X Series

Modbus AC 2-Phase Stepping Motor Driver

Datasheet V3.50/2015.04.05



Hangzhou Automaticage Co., Ltd.

http://www.AgeMotion.com

IMPORTANT NOTES

Carefully read and observe all safety instructions in Chapter 1.

Carefully read this datasheet entirely for using the product correctly.

Keep the datasheet for future reference.

E-book address：
http:// www.AgeMotion.com/download/ds/AHD8C2XDS_ENG.pdf

Catalogue

1 Safety Information

Read this chapter carefully before using the driver to ensure safety.

⚠DANGER

- **DO NOT use the driver in explosive, inflammable or corrosive environment.**
- **DO NOT open the driver's shell.**
- **Driver may exceed 400VDC internal, so driver and motor must connect to safety earth.**
- **Manual operation such as plugging, connecting, setting must process after high voltage indicator LED is entirely off.**
- **If HV LED is off after power on, the manual operation must do after 1 hour later with power off.**
- **Connectors except RJ45 can't plug and pull with power on.**
- **External resistance is high voltage, high temperature.**
- **The driver maybe very hot, do the manual operation after power off and ensure the temperature is low enough.**

Failure to follow these instructions will result in death or serious injury.

WARNING

Application and setting

- The application of the driver must fit to the driver's parameters.
- Check connecting again before the first power on.
- At first increase the torque, then increase the speed.
- Do not start or stop motor by power on or off.
- Connect the brake resistance when the back EMF is too high.

Connection

- Connecting wire must do with connectors pull off from sockets.
- Power wire can't add soldering tin, otherwise the driver may be damaged forever.

Running

- It needs 20 seconds between two power on.
- Be caution that drivers maybe hot when running.

Maintenance and inspection

- Check the fan every month. Clean the fan when power off if the accumulation of dust is too heavy.
- The driver need to be power on once at least 1 hour every 3 months, to ensure the electronic components in good condition.

Repair

- Do not repair the driver by non-our-company authorized maintainer.
- Send the driver back for checking and repairing.

Drop

- Deal as industry pollutant when obsoleted.

Failure to follow these instructions will result in damage.

2 Product Introduction

AHD8C2X use high speed motion control CPU to realize bipolar constant current chopper sine wave micro step driving based on fast adaptive predictive control. The driver has characteristic of high precision, high torque, low noise, wide speed. The driver has functions of self test, fault detection, status indication, and protection.

AHD8C2X can be applied to varieties of numeric control machines. AHD8C2X support Modbus-RTU/RS485 in industry application.

AHD8C2X supports global voltage AC85~265VAC or UPS power.

AHD8C2X fits 56/57/60/85/86/90/110/130/150 stepping motors or line motors.

2.1 Features

Global Wide AC/DC Input

AC 85~265V / DC 120~350V

Multiple Output Current

0.60~1.65A	2.50~4.25A
1.20~2.25A	3.00~6.50A
1.50~3.25A	5.50~9.00A

Multiple Microstepping

Flexible microstepping setting
Up to 3840000 pulse/rev.
Support dynamic microstepping

Multiple control signal

Pulse/DIR, CW/CCW, QEP
Rise edge, fall edge, both edge
PNP, NPN, differential signal
Low and high position sensor

Wide control signal voltage

Default wide voltage +5V~24V
customize to +3.3V~5V, +3.3V, +5V, +12V, +24V
and 5V common ground contactor

Position Memory

Position memory after power off

Variety grades of pulse smooth

0~31 grades to choose

Flexible auto current low setting

Default after 1 second half phase current
For different machine,
set the time and the current dynamicly

100,000h working life

3 years warranty, 10 years working life

Other features：

- Bipolar constant current chopper sine wave micro step driving based on fast adaptive predictive control
- Support Modbus-RTU/RS485, driving and motion control
- Support 8 level baudrate and 111 nodes
- AHD8C2XB support -20~80℃ working temperature
- 2 seconds adaptive starting control after power on16
- Self-test function, running without controller
- Free/Enable
- Ready output
- Protection and indication of under voltage, over voltage, short, over current, open, cross, low temperature, high temperature and internal fault
- Firm, small, long life, high precision, high torque, low noise and wide speed control
- Customization for special application
- Support register auto store and restore
- Support by AgeMotion6(Android version)

2.2 Product Naming

A HD8 C 2 2 A J

- A — AgeMotion Prefix
- HD8 — Micro Step Driver
- C — Modbus High Voltage
- 2 — 2-Phase
- 2 — Peak Phase Current
  - 1： 0.60~1.65A
  - 2： 1.00~2.10A
  - 3： 1.50~3.25A
  - 4： 2.50~4.25A
  - 6： 3.00~6.50A
  - 9： 5.50~9.00A
- A — Function
  - A： -20~50? Work Temp.
  - B： -20~80? Work Temp.
  - C~G： Reserved
  - H~Z： Customized
- J — Input Type
  - H： 5~24 V Voltage
  - I： 3.3 V Voltage
  - J： 5 V Voltage
  - K： 12 V Voltage
  - L： 24 V Voltage
  - P： 5 V Common Ground Contactor

2.3 Product Selecting

<table>
<tr><th>Model</th><th>Power</th><th>Peak Current</th><th>Wrok Temp.</th><th>Addr.</th><th>Baudra te</th><th>Self Test</th><th>Bus Control</th><th>Input Type</th></tr>
<tr><td>AHD8C21AX</td><td>AC85~265V
DC110~350V</td><td>0.60~1.65A</td><td rowspan="6">-20~50℃</td><td rowspan="10">0Broad
1~111</td><td rowspan="10">9600
19200
38400
57600
115200
250000</td><td rowspan="10">Unidir.4
Bidir.4</td><td rowspan="10">Current
Micro
Step
Input Type</td><td rowspan="10">H: 5~24V
I: 5V
J: 12V
K: 24V
L: 3.3V
P: 5V</td></tr>
<tr><td>AHD8C22AX</td><td>AC85~265V
DC120~350V</td><td>1.00~2.10A</td></tr>
<tr><td>AHD8C23AX</td><td>AC85~265V
DC120~350V</td><td>1.50~3.25A</td></tr>
<tr><td>AHD8C24AX</td><td>AC85~265V
DC120~350V</td><td>2.50~4.25A</td></tr>
<tr><td>AHD8C26AX</td><td>AC85~265V
DC120~350V</td><td>3.00~6.50A</td></tr>
<tr><td>AHD8C29AX</td><td>AC85~265V
DC120~350V</td><td>5.50~9.00A</td></tr>
<tr><td>AHD8C21BX</td><td>AC85~265V
DC120~350V</td><td>0.60~1.65A</td><td rowspan="4">-20~80℃</td></tr>
<tr><td>AHD8C22BX</td><td>AC85~265V
DC120~350V</td><td>1.00~2.10A</td></tr>
<tr><td>AHD8C23BX</td><td>AC85~265V
DC120~350V</td><td>1.50~3.25A</td></tr>
<tr><td>AHD8C24BX</td><td>AC85~265V
DC120~350V</td><td>2.50~4.25A</td></tr>
</table>

Notes:

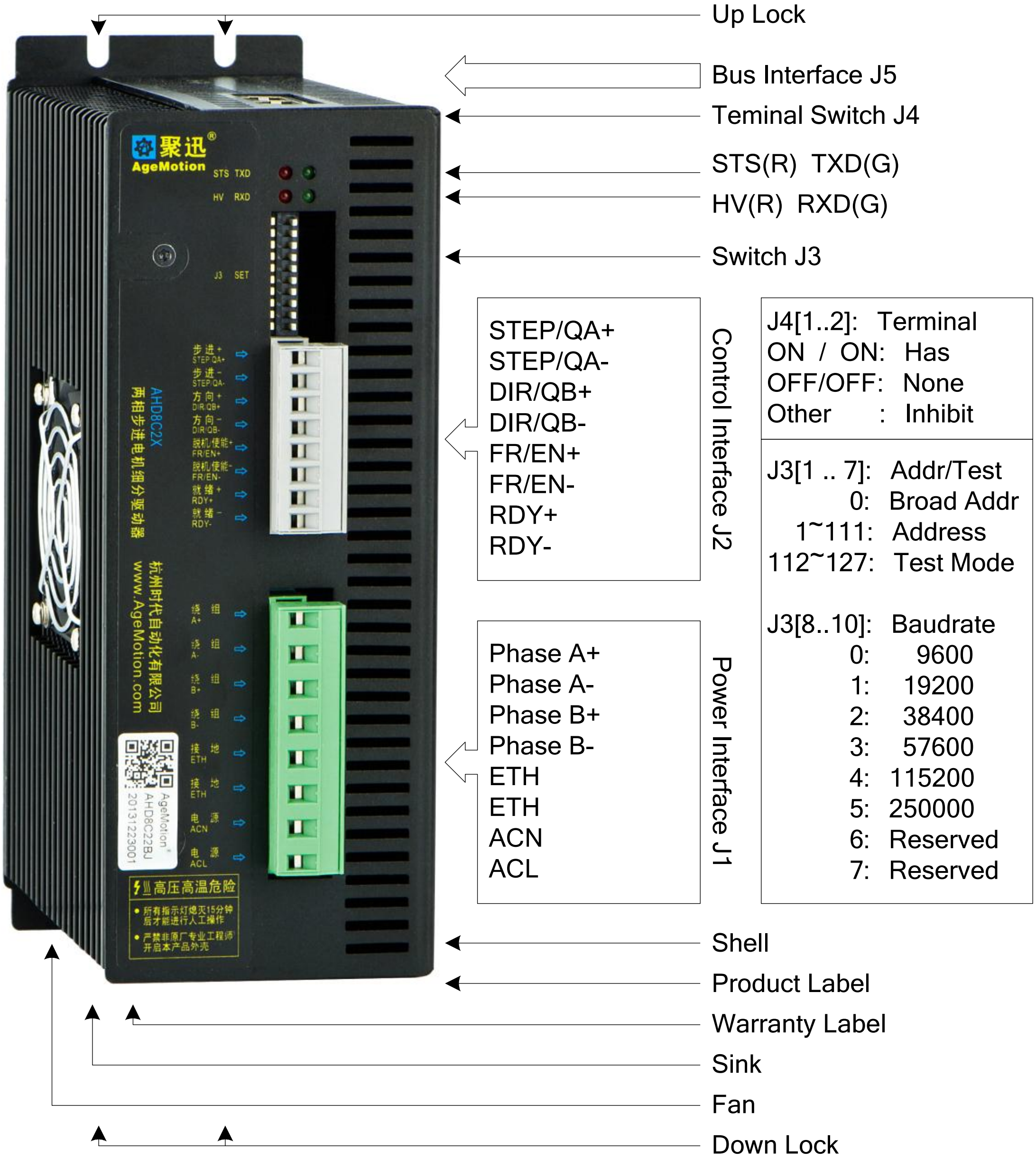
1 X means every sub type.

2 AHD8C2XA uses CPU temperature sensor, AHD8C2XB uses sink and CPU temperature sensor.

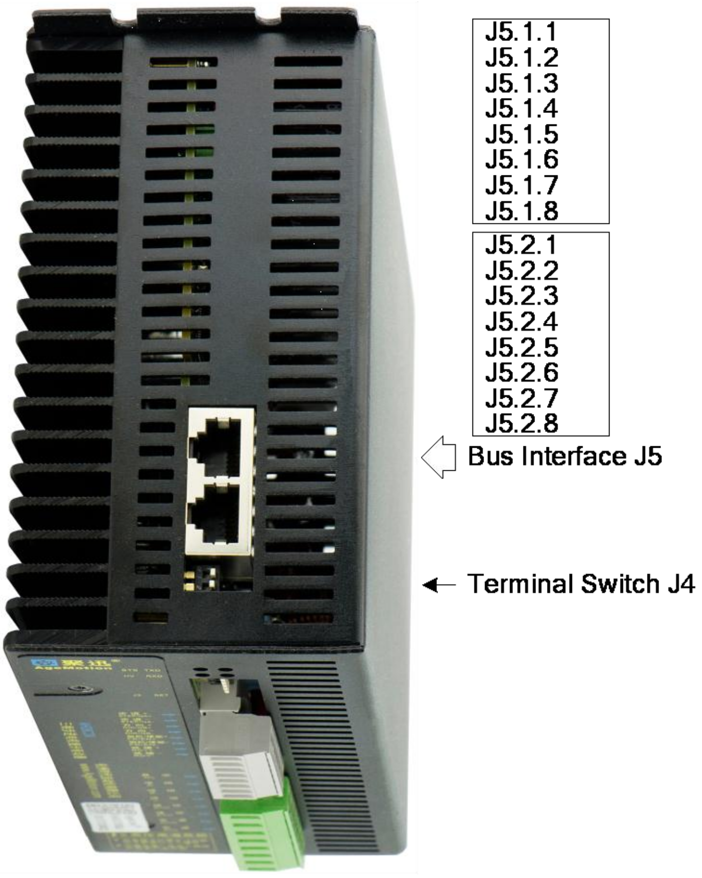
3 AHD8C2X has no break function, select AHD862X if need break.

4 Recommend use QEP if need pulse input type.

2.4 Profile



AHD8C2X



AHD8CXX Modbus Interface

2.5 Customization

AHD8C2X support customization for special application, contact us if needed.

2.6 Serial No.

Warranty label marks product serial number.

Example:

3AJ010702008:

3AJ means AHD8C23AJ, 010702008 means production date 2001.07.02(year.month.day).

2BP110501192:

2BP means AHD8C22BP, 110501192 means production date 2011.05.01(year.month.day).

Each driver has a unique serial number and other information in ROM.

3 Electrical Connector

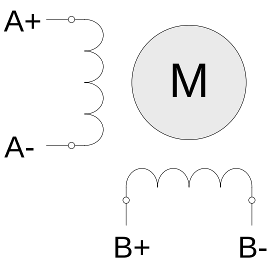
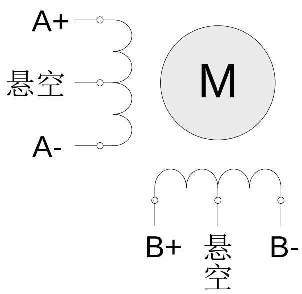
3.1 Power Connector J1

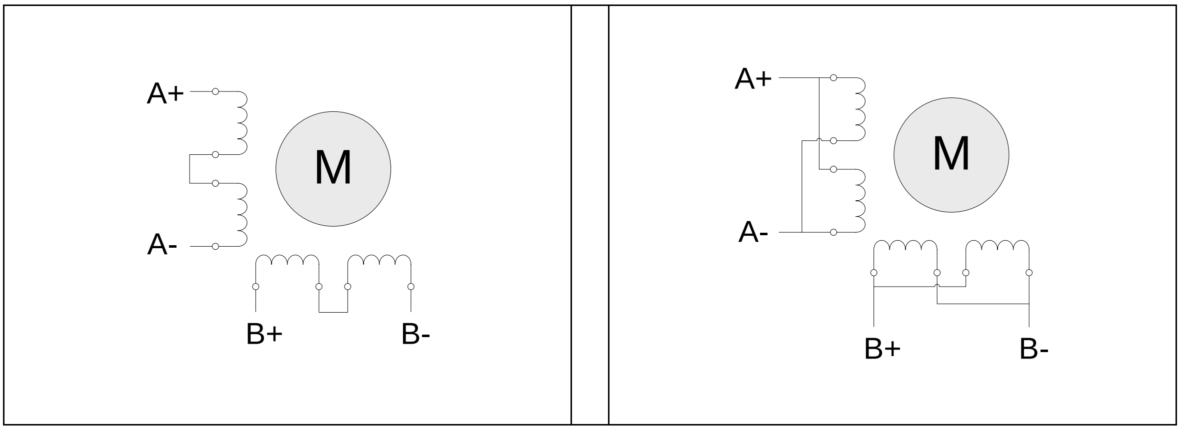
Pin	Name	Description	Connect to
J1.1	Phase A+	Connect to phase A+	2-phase stepping motor
J1.2	Phase A-	Connect to phase A-	
J1.3	Phase B+	Connect to phase B+	
J1.4	Phase B-	Connect to phase B-	
J1.5	ETH	Connect to motor metal shell	
J1.6	ETH	Connect to power input earth	Input AC85~265V
J1.7	ACN	Connect to ACN or UPS	
J1.8	ACL	Connect to ACL or UPS	

3.1.1 Connect to the Motor

The regular 2-phase stepping motor has 4 lead wires: coil A+/A- and coil B+/B-.
J1[1..2] to coil A+/A-;
J1[3..4] to coil B+/B-;
J1.5 to the earth of the motor.

提示 Switch phase A+/A- or switch phase B+/B- can change the running direction of the motor.

Wiring Diagram for Different 2-phase Stepping Motor	
4 wires	6 wires
driver current=motor phase current	driver current=motor phase current
	
8 wires（series connection）	8 wires（parallel connection）
driver current=motor phase current	driver current=motor phase current×2



3.1.2 Connect to Power Supply

J1[6..8] input voltage range AC85~265V/DC120~350V, input power need to be 150%~200% of output power. Driver and motor need connect to power earth.

Power voltage is irrelevant with motor static torque. High torque at high speed can be provided by rising input voltage.

J1.6 must be connect to power earth.

3.1.3 Power Connection Precautions

1. Connect wire after connector has been pulled off from socket and HV LED has been off for 15 minutes.
2. Use thick wire, especially wire for earth.
3. Do not add soldering tin on wire, connect directly or by a copper header.

3.2 Control Connector J2

Pin	Name	Description	Signal Direction
J2.1	STEP/QA+	Pulse(single pulse mode) QA±(QEP mode) Positive pulse(double pulse mode)	Controller→Driver
J2.2	STEP/QA-		
J2.3	DIR/QB+	Direction（single pulse mode) QB±(QEP mode) Negative pulse(double pulse mode)	Controller→Driver
J2.4	DIR/QB-		
J2.5	FR/EN+	Free/Enable	Controller→Driver
J2.6	FR/EN-		
J2.7	RDY+	Ready	Driver→Controller
J2.8	RDY-		

3.2.1 Step Control

J2[1..4] are pulse signal, there are 6 input types:

Control Signal Type	Pin Function	
DIR/STEP, STEP fall edge	J2[1..2]: STEP+/-;	J2[3..4]: DIR+/-
DIR/STEP, STEP rise edge	J2[1..2]: STEP+/-;	J2[3..4]: DIR+/-
DIR/STEP, STEP fall/rise edge	J2[1..2]: STEP+/-;	J2[3..4]: DIR+/-
QEP quadrature	J2[1..2]: QA+/-;	J2[3..4]: QB+/-
Low and high position sensor	J2[1..2]: Low position sensor;	J2[3..4]: High position sensor
CW/CCW, STEP fall edge	J2[1..2]: CW+/-;	J2[3..4]: CCW+/-

In the following time diagram, DIRI means equivalent direction input and STEPI means equivalent step input.

DIRI in high logic means moving positive, B-Phase current ahead 90° of A-Phase current; DIRI in low logic means moving negative, B-Phase current back 90° of A-Phase current. STEPI high means effective micro step input pulse.

QEP is quadrature encoded pulses. QEP cost lowest controller resource, but has highest reliability and pulse bands.

3.2.2 Free/Enable Control

J2[5..6] is Free/Enable input signal.

When free is active, phase current will be decrease to zero, and motor has only hold torque and zero driving torque.

Fan will stop when driver is free.

3.2.3 Ready Signal

Ready signal will output OC low 2 seconds after power on. Driver will response to pulse input or Modbus command after ready output.

Ready signal will output high impedance when fault occurs or free is active.

3.2.4 Control Input Type

Control interface support 6 kinds of control voltage.

J2 input and output are opto-isolated.

When input is left floating, input is equivalent to logic high. 1 and 0 are used to express logic high and low in this datasheet.

Opto-isolation current limiting resistor is listed below:

Control Voltage(V)	R1(Ω)	Opto Driving Current(mA)
5~24 (default)	1500	2~20
3.3~5	680	2~5

3.3	133	10
5	240	10
12	804	10
24	1800	10
5 Common GND	1000	3mA

- +5~24V Voltage Control

+5~24V input is designed to fit many applications that control voltage may be changed.

- +3.3~5V Voltage Control

+3.3~5V input is designed to fit many applications that control voltage may be changed.

- +3.3V Voltage Control

+3.3V input is designed to be compatible with RS422.

- +5V Voltage Control

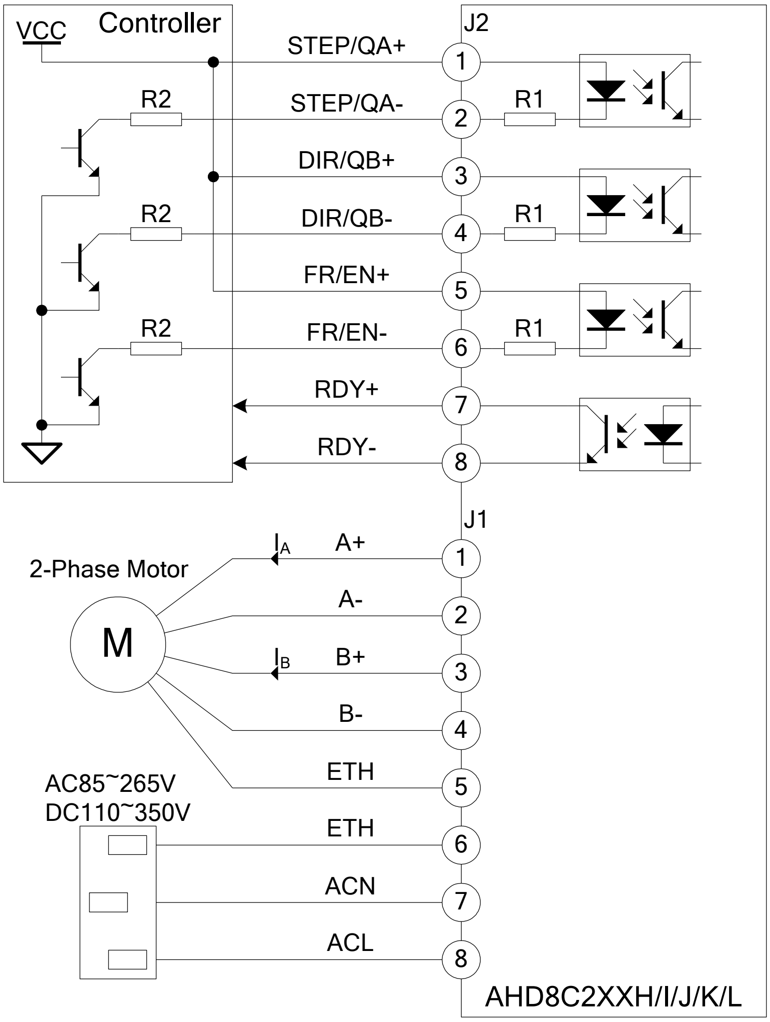
+5V input is designed be controlled by 5V logic.

- +12V Voltage Control

+12V input is designed to fit PLC application.

- +24V Voltage Control

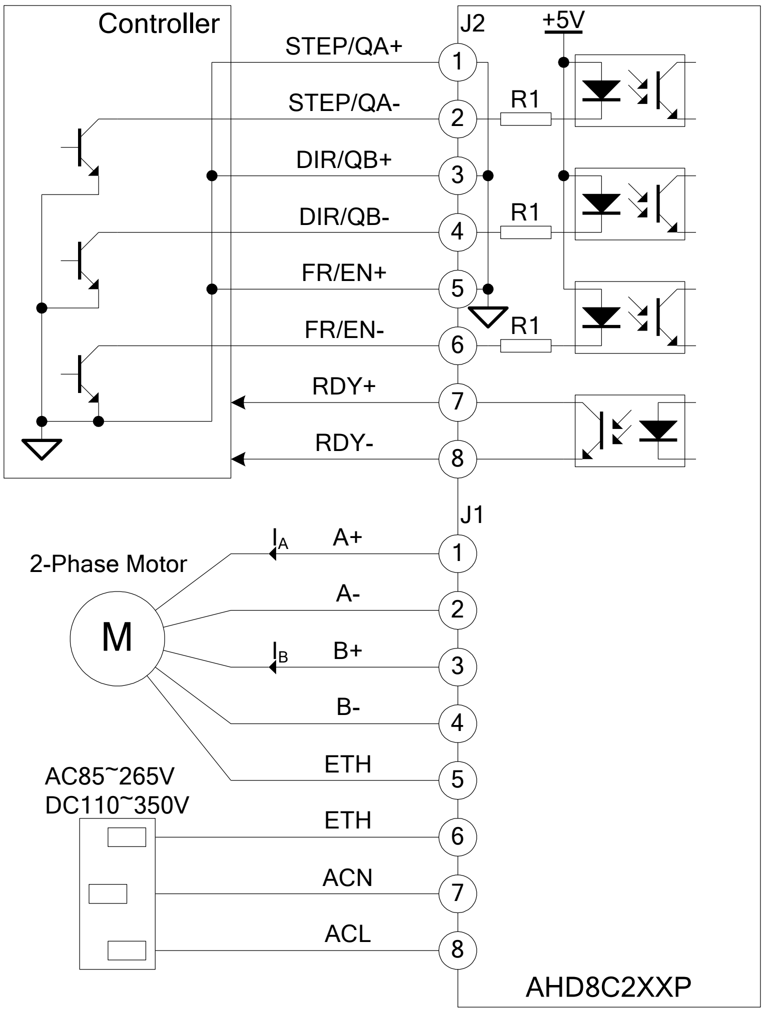
+24V input is designed to fit PLC application.

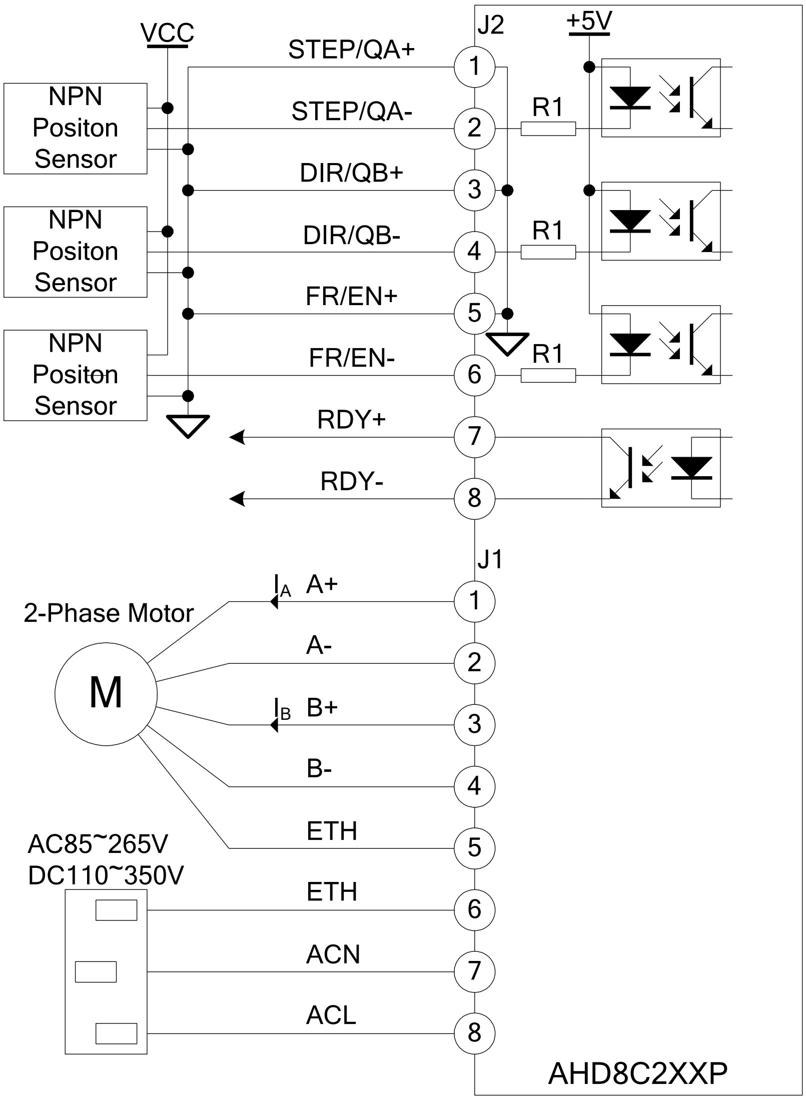

VCC
Controller
J2
STEP/QA+
STEP/QA-
DIR/QB+
DIR/QB-
FR/EN+
FR/EN-
RDY+
RDY-
R2
R1
J1
2-Phase Motor
M
A+
A-
B+
B-
ETH
ETH
ACN
ACL
AC85~265V
DC110~350V
AHD8C2XXH/I/J/K/L

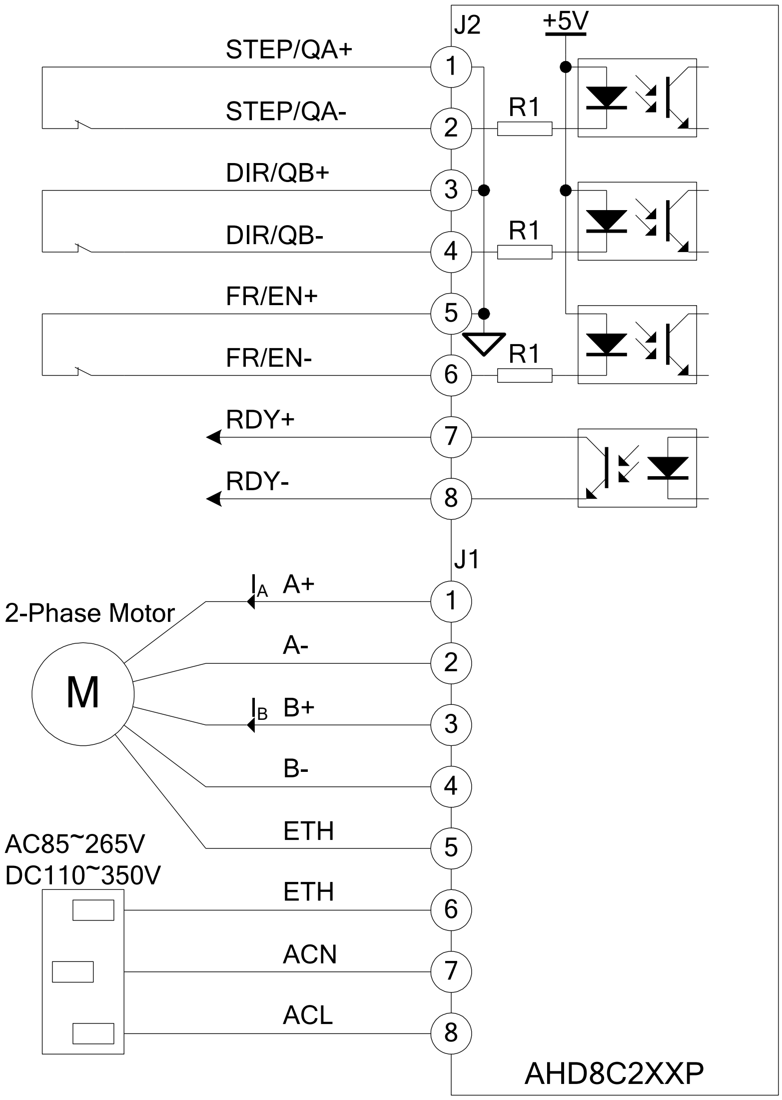
- +5V Common Ground Contactor

This input type is designed for OC/OD/5V logic control. Input circuit has internal isolation power, so mechanical contactor can be direct connect to driver without external power.

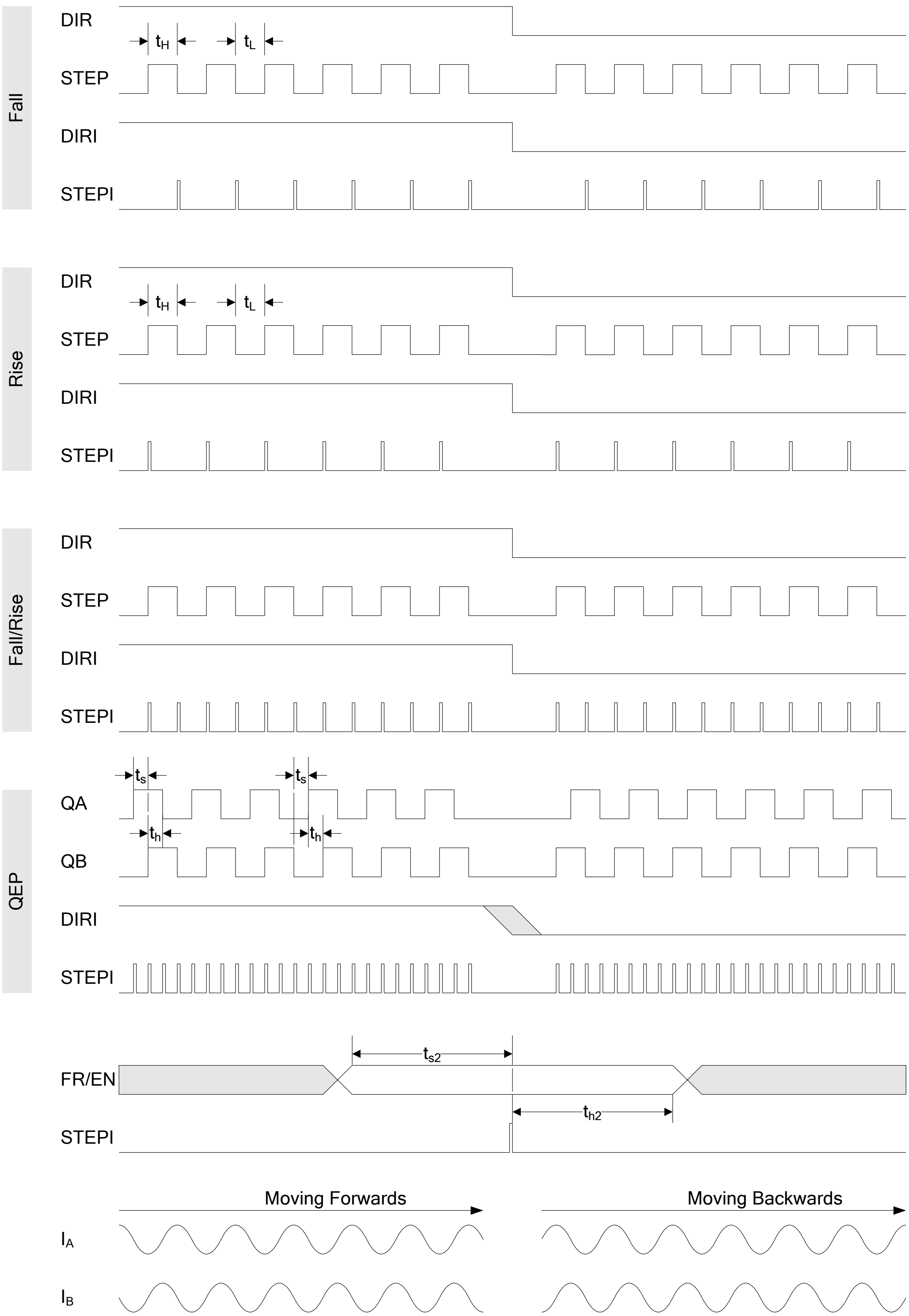
There are 3 kinds of connection for common ground contactor.




VCC
J2
+5V
NPN
Positon
Sensor
NPN
Positon
Sensor
NPN
Positon
Sensor
STEP/QA+
STEP/QA-
DIR/QB+
DIR/QB-
FR/EN+
FR/EN-
RDY+
RDY-
R1
R1
R1
1
2
3
4
5
6
7
8
J1
2-Phase Motor
M
IA A+
A-
IB B+
B-
ETH
ETH
ACN
ACL
AC85~265V
DC110~350V
AHD8C2XXP


J2
+5V
STEP/QA+
STEP/QA-
DIR/QB+
DIR/QB-
FR/EN+
FR/EN-
RDY+
RDY-
R1
R1
R1
1
2
3
4
5
6
7
8
J1
2-Phase Motor
M
IA A+
A-
IB B+
B-
ETH
ETH
ACN
ACL
AC85~265V
DC110~350V
AHD8C2XXP

3.2.5 Sequence Diagram



4 Switch Setting

4.1 Switch J3

Driver Modbus parameters or test status are set by J3.
J3[8..10] set bus baudrate, J3[1..7] set bus address or selftest status.
J3 must set before power on, set after power on will be ignored.

J3[8..10]	J3-8	J3-9	J3-10	Baudrate(bPS)
0	0	0	0	9600
1	1	0	0	19200
2	0	1	0	38400
3	1	1	0	57600
4	0	0	1	115200
5	1	0	1	250000
6	0	1	1	Reserved(250000)
7	1	1	1	Reserved(250000)

J3[1..7]	J3-1	J3-2	J3-3	J3-4	J3-5	J3-6	J3-7	Bus Address or Test Status
0	0	0	0	0	0	0	0	Broad Address 0
1~111	1~111							Bus Address1~111
Test Status 0~15	X	X	X	1	1	1	1	Selftest Status

4.2 Terminal Switch J4

When driver is located at the end of physical end, J4[1..2] need be set to ON to enable terminal resistor. Set to OFF if not located at the two ends. Another set is not permitted.

4.3 Selftest Status

Driver is silent to bus when in selftest mode, when controller command and pulse are ignored.

4.4 Microstepping Setting

Full step is divided into several micro steps.
Full step counts:

$$S = T \times 4$$, T is motor tooth counts

Micro step counts per revolution:

$$S = T \times 4 \times MicroStep$$

Angle of single micro step:

$$D = \frac{360°}{T \times 4 \times MicroStep}$$

1.8° motor has full step 1.8°, that is motor has 50 teeth, 0.9° motor has full step 0.9°, that is motor has 100 teeth.

4.5 Phase Current Setting

Peak phase current of step motor can be set dynamically via Modbus.

Relationship between driver peak current and motor rms current:

$$I_{PDriver} = I_{MotorRMS} \times \sqrt{2}$$

5 Status Indicator

5.1 Status Indicator STS

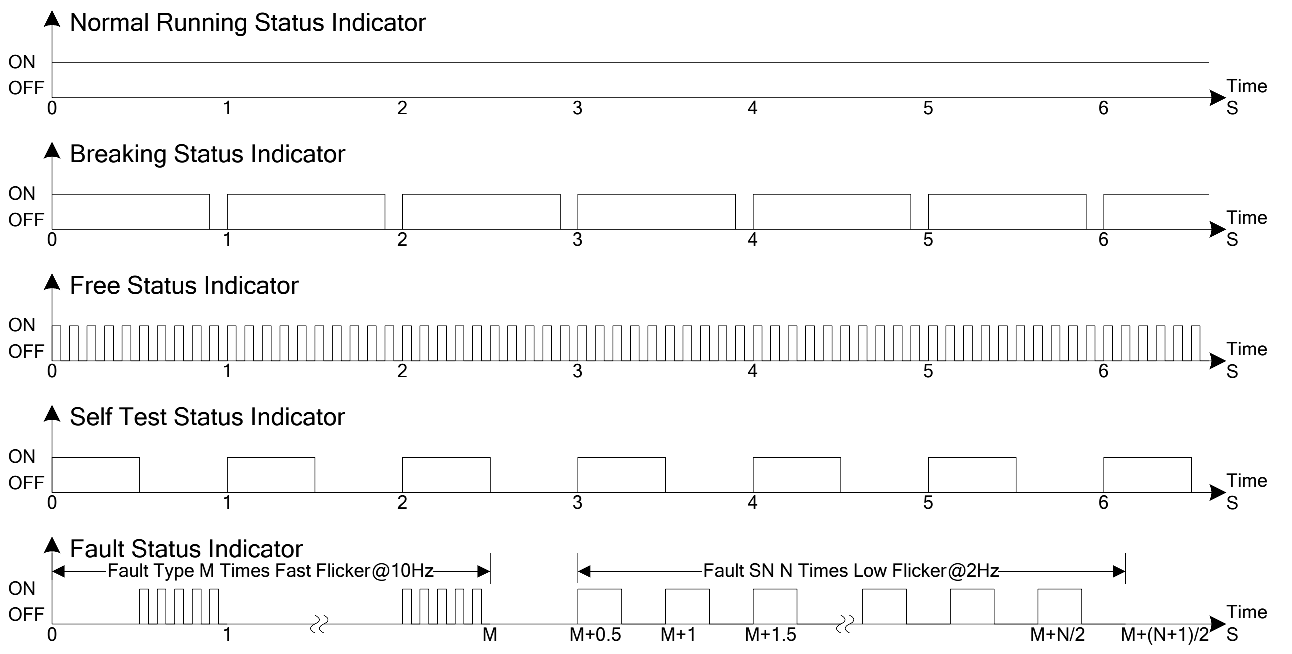
Red Status indicator LED is used to show driver realtime status. There are 7 types of status:

1. Always ON: driver is running normally, enabled and need not break.
2. Short OFF: driver is breaking, LED ON 0.9S, OFF 0.1S.
3. Always fast flicker: driver is free, LED flickers at 10Hz.
4. Always slow flicker: driver is self testing, LED flickers at 2Hz. Driver will not response to external control signal or Modbus command.
5. Fault fast flicker: LED flicker 1S at 10Hz, OFF 0.5S, repeat M times.
6. Fault slow flicker: LED flicker at 2Hz, repeat N times.
7. Fault: when driver has internal or external fault, LED indicator cycle is fault fast flicker M times and fault slow flicker N times, M is fault type, N is fault SN. Faults are list below.

If LED is OFF after power on, then power off for 1 hour before manual operation.

Driver need repower to run normally if fault type is error. Driver will run normally if fault type is warning and the warning is disappeared later

LED Status Indicator Example



LED Status Indicator List					
LED Status		Fault Source	Running Mode	Fault Reason	Solution
Always ON			OK		
Always OFF		External		Serious undervoltage	Check power connecting and voltage
		Internal		Driver internal fault	Power off for 1 hour and send back to factory
Short OFF		External	Breaking		Consider to use AHD86XX
Fast Flicker			Free		
Always Slow Flicker			Self Test		
Fault Flicker					
Fast M	Slow N				
1	1~31	Internal	Error	Internal Fault	Send back to factory
2	1	External	Warning	Undervoltage	Raise power voltage
2	2	External	Warning	Overvoltage	Reduce power voltage, consider add breaking if power is OK
2	3	External	Warning	Temperature Low	Raise environment temperature
2	4	External	Error	Temperature High	Check fan and installation
2	9	External	Error	A/B Cross Phase	Check motor and wire
2	17	External	Error	A-Phase Open	Check motor and wire
2	18	External	Error	B-Phase Open	Check motor and wire
3	1	External	Error	Main Circuit Short	Check motor and wire, send back to factory if everything is OK
3	2	External	Error	Short when start	Check motor and wire, send back to factory if everything is OK
3	3	External	Error	Short when selftest	Check motor and wire, send back to factory if everything is OK
3	15	External	Error	Over Load	Check motor and wire, send back to factory if everything is OK
Other				Reserved Fault	Send back to factory

5.3 High Voltage Indicator HV

Red HV ON means internal high voltage of driver is ON.

5.4 Transmit Indicator TXD

Green TXD ON means the driver is transmitting data to controller.

5.5 Receive Indicator RXD

Green RXD ON means the driver has received a correct frame from controller.

When RXD is ON but TXD is never ON means driver has received correct broad or other address frame, but the driver never response.

6 Registers

6.1 Registers List

Memory	Register	Datatype	Description	Read	Write	Nonvolatile	Authority
0x0000	Control	WORD	Control Register	WORD	WORD		Free
0x0008	InputType	WORD	Input Type Register	WORD	WORD	√	
0x0010	CurrentMax	UINT16	Maximum Current Register	WORD			
0x0011	CurrentMin	UINT16	Minimum Current Register	WORD			
0x0012	CurrentSet	UINT16	Current Set Register	WORD	WORD	√	
0x0013	CurrentLow	UINT16	Current Low Register	WORD	WORD	√	
0x0014	CurrentLowWT	UINT16	Current Low Waiting Time Register	WORD	WORD	√	
0x0020 0x0021 0x0022 0x0023	Position	INT64	Motor Real Position Register	低 DWORD QWORD	低 DWORD QWORD	√	
0x0024 0x0025 0x0026 0x0027	PositionSet	INT64	Motor Position Set Register	低 DWORD QWORD	低 DWORD QWORD		
0x0028 0x0029	TResolution	UINT32	Motor Tooth Resolution Register	DWORD			
0x002A 0x002B	PulseLength	UINT32	Pulse Step Length Register	DWORD	DWORD	√	
0x002C 0x002D	PulsePosition	INT32	Motor Pulse Real Position Register	低 WORD DWORD	低 WORD DWORD	√	
0x002E 0x002F	PulsePositionSet	INT32	Motor Pulse Position Set Register	低 WORD DWORD	低 WORD DWORD		
0x0040	VelSet	UINT16	Motor Velocity Set Register	WORD	WORD	√	
0x0041	VelStart	UINT16	Motor Start Velocity Register	WORD	WORD	√	
0x0042	VelFilter	UINT16	Motor Velocity Filter Register	WORD	WORD	√	
0x0043	KV	UINT16	Motor Velocity Coefficient Register	WORD			
0x0060	BusWDT	UINT16	Bus Watchdog Timer Register	WORD	WORD	√	
0x0080	Port	UINT16	Port Register	WORD			
0x0081	PortHiFlag	UINT16	Port High Flag Register	WORD	WORD		
0x0082	PortLoFlag	UINT16	Port Low Flag Register	WORD	WORD		

0x0083	PortFlipFlag	UINT16	Port Flip Flag Register	WORD	WORD		
0x0300	CpuTemp	INT16	Driver's CPU Temperature Register	WORD			
0x0301	SinkTemp	INT16	Driver's Sink Temperature Register	WORD			

Notes:

1. Memory unit is WORD(2Byte) type;
2. Registers in memory is Little-Endian, that is low byte at low address and high byte at high address;
3. Register transmission is Big-Endian via Modbus-RTU, that is high byte at low address and low byte at high address;
4. Nonvolatile registers will be restored value when power off;
5. Nonvolatile registers will be automatically saved when power off;
6. Recommend that bus controller initialize all writable registers after driver power on;
7. Position registers may have storage offset when power off because of inertia;
8. Only memory listed above can be visited, reserved bits of registers can't write 1;
9. Registers above can be visited freely and need not authority;
10. Only support 6 types of operation mode listed above.

6.2 Register Operation

6.2.1 Operation Mode

Registers support 6 types of operation mode:

1. ReadWORD: Read 1 WORD;
2. ReadDWORD: Read 2 WORD;
3. ReadQWORD: Read 4 WORD;
4. WriteWORD: Write 1 WORD;
5. WriteDWORD: Write 2 WORD;
6. WriteQWORD: Write 4 WORD.

6.2.2 Operation Rules

1. Only 1 register for every operation;
2. Supported operation only;
3. Response error or no response for unauthorized operation.

6.3 Registers Detail

6.3.1 Control – Control Register

Address: 0x0000
Operation: ReadWORD/WriteWORD, volatile

Bits	Name	Type	Value	Default	Description
0	Reset	Bit	0~1	0	R : Always 0 W : 0 No act 1 Reset driver and clear the bit to 0
1	ResetValue	Bit	0~1	0	R : Always 0 W : 0 No act 1 Reset value to default and reset driver, then clear the bit to 0
2	Free	Bit	0~1	0	R/W : 0 Enable driver 1 Free driver
3	Pause	Bit	0~1	0	R/W : 0 Continue 1 Pause
4	MovePSL	Bit	0~1	0	R/W : 0 No act 1 Move to low position sensor, clear when arrived
5	MovePSH	Bit	0~1	0	R/W : 0 No act 1 Move to high position sensor, clear when arrived
6~15	Reserved				

Bit0 has the highest priority, Bit15 has the lowest.

Be careful to use Reset, ResetValue, and Free. Reset and ResetValue using interval must be longer than 20 seconds, function like power on and off. Action counts must less than 10 every day. Free or Enable can be used only when motor is idle.

When free, driver will enter low power mode, motor phase current is 0. Driver will be in normal mode if Control.Free=0 and J2[5..6] is enable.

Pause will stop motor, and motor will continue when Pause is cleared.

Moving low position is inhibited if low position sensor is arrived, but moving high position is permitted.

Moving high position is inhibited if high position sensor is arrived, but moving low position is permitted.

PositionSet = PositionSet when MovePSL or MovePSH is finished, MovePSL and MovePSH will be cleared after action.

MovePSL or MovePSH must be used with InputType.PulseType = 8.

6.3.2 InputType – Input Type register

Address: 0x0008
Operation: ReadWORD/WriteWORD, nonvolatile

Bits	Name	Type	Value	Default	Description
0~3	PulseType	4Bit	0~16	8	R/W : 0 Direction + Pulse Fall Edge 1 Direction + Pulse Rise Edge 2 Direction + Pulse Fall and Rise Edge 3 QEP 8 PSL + PSH Other settings are reserved
4~15	Reserved				

Recommend QEP type when use pulse input type.

6.3.3 CurrentMax – Maximum Current Register

Address: 0x0010
Operation: ReadWORD/WriteWORD, volatile

Bits	Name	Type	Value	Default	Description
0~15	CurrentMax	UINT16	$100I_{PMax}$	$100I_{PMax}$	R : Maximum Peak Phase Current of Driver W : Illegal

I_{PMax} is the maximum peak phase current of driver, I_{PMax} and CurrentMax have been fixed before deliver and can't be changed by user.

Maximum peak current of driver

$$I_{PMax} = 0.01 \times CurrentMax$$, unit: A

That is

$$CurrentMax = 100 \times I_{PMax}$$

Example:

AHD8C26XX: I_{PMax} = 6.50A, CurrentMax = 650.
AHD8C29XX: I_{PMax} = 9.00A, CurrentMax = 900.

6.3.4 CurrentMin – Minimum Current Register

Address: 0x0011
Operation: ReadWORD/WriteWORD, volatile

Bits	Name	Type	Value	Default	Description
0~15	CurrentMin	UINT16	$100I_{PMin}$	$100I_{PMin}$	读 ：驱动器相电流最小峰值 写 ：非法

I_{PMin} is the maximum peak phase current of driver, I_{PMin} and CurrentMax have been fixed before deliver and can't be changed by user.

Minimum peak current of driver

$I_{PMin} = 0.01 \times CurrentMin$, unit: A

That is

$CurrentMin = 100 \times I_{PMin}$

Example:

AHD8C26XX: I_{PMin} = 2.50A, CurrentMin = 250.

AHD8C29XX: I_{PMin} = 5.00A, CurrentMin = 500.

6.3.5 CurrentSet – Current Set Register

Address: 0x0012

Operation: ReadWORD/WriteWORD, nonvolatile

Bits	Name	Type	Value	Default	Description
0~15	CurrentSet	UINT16	$100I_{PMin}$~$100I_{PMax}$	100 I_{PMin}	读写：驱动器相电流峰值设定

Peak phase current of driver I_{PSet} is set by user, it can't greater than I_{PMax} or smaller than I_{PMin}. Notes the relationship between peak phase current of driver and rms current of moter.

Peak phase current of driver set by user

$I_{PSet} = 0.01 \times CurrentSet$, unit: A

That is

$CurrentSet = 100 \times I_{PSet}$

Example:

AHD8C26XX: Peak phase current range 2.50~6.50A.

If user want to set I_{PSet}= 5.31A, then CurrentSet = 531.

If user want to set I_{PSet}= 4.57A, then CurrentSet = 457.

6.3.6 CurrentLow – Current Low Register

Address: 0x0013

Operation: ReadWORD/WriteWORD, nonvolatile

Bits	Name	Type	Value	Default	Description
0~15	CurrentLow	UINT16	30~100	30	R/W : Peak Phase Current Low Percent of Driver

If waiting time in idle exceed the time of waiting low current, the driver will enter low current status to decrease driver and motor temperature.

Driver power will decrease with lower current. Torque is larger and temperature is higher when CurrentLow is larger, torque is smaller and temperature is lower when CurrentLow is smaller.

Low peak phase current

$I_{PLow} = I_{PSet} \times CurrentLow\%$, unit: A

That is

$$CurrentLow = 100 \times \frac{I_{PLow}}{I_{PSet}}$$

Example:

AHD8C26XX: peak phase current range 2.50~6.50A, I_{PMax} = 6.50A, CurrentMax = 650;
User sets I_{PSet}= 5.31A, CurrentSet = 531;
User sets I_{PLow}= 2.00A, CurrentLow = 38.

6.3.7 CurrentLowWT – Current Low Waiting Time Register

Address: 0x0014
Operation: ReadWORD/WriteWORD, nonvolatile

Bits	Name	Type	Value	Default	Description
0~15	CurrentLowWT	UINT16	50~65535	1000	R/W : Waiting Time for Current Low

If waiting time in idle exceed the time of waiting low current, the driver will enter low current status, and then peak phase current will be set to I_{PLow}.

Waiting time for current low

$$T_{CurrentLow} = 0.001 \times CurrentLowWT$$, unit: S

That is

$$CurrentLowWT = 1000 \times T_{CurrentLow}$$

Range of Waiting time: 0.05~65.535 seconds.

6.3.8 Position – Motor Realtime Position Register

Address: 0x0020~0x0023
Operation: Low DWORD support ReadDWORD/WriteDWORD, ReadQWORD/WriteQWORD, nonvolatile

Bits	Name	Type	Value	Default	Description
0~63	Position	INT64	-0x20000000 00000000 ~ +0x1FFFFFFF FFFFFFFF	Stored	R : Motor Realtime Position W : Motor Realtime Position will be set to new value unit: Minimum Micro Step(MMS)

If INT64 is not supported, controller can use ReadDWORD/WriteDWORD to visit low DWORD, but high DWORD can't be visited alone.

Motor revolutions of Minimum micro step MMS

$$Revolution_{MMS} = \frac{1}{TResolution \times T}$$, unit: r

TResolution: Resolution of a single tooth of motor;
T: Tooth counts of motor.

6.3.9 PositionSet – Motion Set Position Register

Address: 0x0024~0x0027

Operation: Low DWORD support ReadDWORD/WriteDWORD, ReadQWORD/WriteQWORD

Bits	Name	Type	Value	Default	Description
0~63	PositionSet	INT64	-0x20000000 00000000 ~ +0x1FFFFFFF FFFFFFFF	Stored	R/W : Motor position set by user Each successful write will active a motion unit: Minimum Micro Step(MMS)

If INT64 is not supported, controller can use ReadDWORD/WriteDWORD to visit low DWORD, but high DWORD can't be visited alone.

Motor will moving from Position to PositionSet at set motion parameters, Position=PositionSet when arrived.

Set Control.Pause first, then set Position and PositionSet, finnally clear Control.Pause. Motor will not move unwanted action during set Position and PositionSet such as set to origin by this methoad.

Example:

Position = 10000, PositionSet = 12000, motor will moving forwards 2000MMS;

Position = 10000, PositionSet = 9000, motor will moving backwards 1000MMS.

6.3.10 TResolution – Motor Single Tooth Resolution Register

Address: 0x0028~0x0029

Operation: ReadDWORD, volatile

Bits	Name	Type	Value	Default	Description
0~31	TResolution	UINT32	1~0xFFFFFFFF	76800	R : Single Tooth Resolution of Motor W : Illegal unit: Minimum Micro Step(MMS)

Single tooth resolution of motor is minimum micro step counts of a tooth of motor.

Motor revolution

$$\text{Revolution} = \frac{\Delta\text{Position}}{\text{TResolution} \times \text{T}}$$, unit: r

That is

$$\Delta\text{Position} = \text{Revolution} \times \text{TResolution} \times \text{T}$$

ΔPosition: Moving length unit MMS;

T : Motor tooth counts, normally 50, 100, 200 or other teech counts.

Example:

T = 50, Position = 10000000, PositionSet = 11920000,

Then ΔPosition = PositionSet – Position = 1920000, Revolution = 0.5r, that is half revolution.

6.3.11 PulseLength – Pulse Length Register

Address: 0x002A~0x002B
Operation: ReadDWORD/WriteDWORD, nonvolatile

Bits	Name	Type	Value	Default	Description
0~31	PulseLength	UINT32	1~3840000	1536	R/W : Pulse Length unit MMS

Motor revolution of a pulse step

$$Revolution_{Step} = \frac{PulseLength}{TResolution \times T}$$, unit: r

That is

$$PulseLength = Revolution_{Step} \times TResolution \times T$$

A pulse moving length range $\frac{1 \sim 3840000}{3840000}$ r, so pulse counts of moving a round may be not an interger, but it can be used with no moving errors.

Example:

T = 50, PulseLength = 1536, then $Revolution_{Step}$ = 0.0004r, that is 2500Pulse/r.

T = 100, PulseLength = 1536, then $Revolution_{Step}$ = 0.0002r, that is 5000Pulse/r.

6.3.12 PulsePosition – Motor Realtime Pulse Position Register

Address: 0x002C~0x002D
Operation: Low WORD support ReadWORD/WriteWORD, ReadDWORD/WriteDWORD, nonvolatile

Bits	Name	Type	Value	Default	Description
0~31	PulsePosition	INT32	-0x80000000 ~ +0x7FFFFFFF	Stored	R : Motor Realtime Pulse Position W : Set Motor Realtime Pulse Position Unit: PulseLength

If INT32 is not supported, controller can use ReadWORD/WriteWORD to visit low WORD, but high WORD can't be visited alone.

Motor revolution of ΔPulsePosition

$$Revolution_{PulsePosition} = \frac{\Delta PulsePosition \times PulseLength}{TResolution \times T}$$, unit: r

Position of PulsePosition

$$Position = PulsePosition \times PulseLength$$, unit: MMS

PulsePosition of Position

$$PulsePosition = \frac{Position}{PulseLength}$$, unit: PulseLength

Write PulsePosition will update Position, read PulsePosition is interger result of equation above.

Position and PositionSet are INT64, they need high performance controller; PulsePosition and PulsePositionSet are INT32, they need low performance controller. Engineer can use such as 1000Pulse/r easily.

Example:

Position = 19, PulseLength = 5, read PulsePosition = 3;

Position = 19, PusleLength = 5, write PulsePosition = 3, then Position = 15.

6.3.13 PulsePositionSet – Motor Pulse Position Set Register

Address: 0x002E~0x002F

Operation: Low WORD support ReadWORD/WriteWORD, ReadDWORD/WriteDWORD

Bits	Name	Type	Value	Default	Description
0~31	PulsePositionSet	INT32	-0x80000000 ~ +0x7FFFFFFF	Stored	R/W：Motor Pulse Position Set Each successful write will active motor moving unit: PulseLength

If INT32 is not supported, controller can use ReadWORD/WriteWORD to visit low WORD, but high WORD can't be visited alone.

Motor will move from Position to PositionSet at set motion parameters, Position=PositionSet when arrived. Notes that moving condition is Position and PositionSet, not PulsePosition and PulsePositionSet. Position maybe not arrived when PulsePosition has arrived, PulsePosition must be arrived when Position has arrived.

Example:

Driver has set to be 1000Pulse/r, then

PulsePosition = 10000, PulsePositionSet = 12000, motor will moving forwards 2000Pulse, that is 2r;

PulsePosition = 10000, PulsePositionSet = 9000, motor will moving backwards 1000Pulse, that is 1r.

6.3.14 VelSet – Motor Velocity Set Register

Address: 0x0040

Operation: ReadWORD/WriteWORD, nonvolatile

Bits	Name	Type	Value	Default	Description
0~15	VelSet	UINT16	1~38400	192	R/W：Motor Velocity Set

Motor revolution speed

$$RSpeed = \frac{VelSet \times KV \times 60000}{TResolution \times T}$$, unit: rpm

That is

$$VelSet = \frac{RSpeed \times TResolution \times T}{KV \times 60000}$$

Example:

T = 50, VelSet = 192, KV = 20, then motor revolution speed is 60rpm. Motor revolution speed range 0.3125~12000rpm when VelSet = 1~38400.

T = 50, VelSet = 38400, KV = 20, then motor revolution speed is 12000rpm. If Position = -0x2000 0000 0000 0000, PositionSet = 0x1FFF FFFF FFFF FFFF, then moving time

$$T_{Run} = \frac{PositionSet - PositionSet}{TResolution \times T \times RSpeed} = \frac{0x4000000000000000}{76800 \times 50 \times 12000} \approx 100079992$$

Minute, about 190 years.

So large position offset will let motor running almost forever.

6.3.15 VelStart – Motor Start Velocity Register

Address: 0x0041

Operation: ReadWORD/WriteWORD, nonvolatile

Bits	Name	Type	Value	Default	Description
0~15	VelStart	UINT16	1~1920	96	R/W：Motor Start Velocity

Motor start revolution speed

$$RSpeed_{Start} = \frac{VelStart \times 1200000}{TResolution \times T}$$, unit: rpm

That is

$$Vel_{Start} = \frac{RSpeed_{Start} \times TResolution \times T}{1200000}$$

Motor velocity will directly jump to start velocity when accelerating from idle, and will directly jump to idle when decreasing from start velocity.

Lower start velocity will set motor start and stop more stable, but need more start and stop time.

Example:

T = 50, default VelStart is 96, so default start velocity is 30rpm,

If VelStart is 16, start velocity is 5rpm.

Start velocity range: 0.3125~600rpm.

6.3.16 VelFilter – Motor Velocity Filter Register

Address: 0x0042

Operation: ReadWORD/WriteWORD, nonvolatile

Bits	Name	Type	Value	Default	Description
0~15	VelFilter	UINT16	0~31	20	R/W：Motor Velocity Filter

Internal velocity filter is more stable than external motion controller. Motor will moving stablly even long length per step(Such as 1r/Pulse) by set a proper filter.

Filter will generate dynamic errors, that is Position and PositionSet have a small offset value.

Filter will not generate stable errors, that is motor will arrive T_{Delay} later, stable error is 0.

Filter will generate motion delay, approximate equation

$$T_{Delay} \approx 0.1 \times 2^{\frac{VelFilter-18}{2}}, \text{ unit: S}$$

Example:

T = 50, $RSpeed_{Start}$ = 30rpm, default delay is 0.2S.

In this case, the maximum dynamic position error

$$Position_{Err} \approx 2^{\frac{VelFilter-18}{2}} \times \frac{RSpeed}{2400}, \text{ unit: r}$$

$Position_{Err}$ is 0.5r when RSpeed = 600rpm.

$Position_{Err}$ decreases when $RSpeed_{Start}$ increasing, $Position_{Err}$ increases when $RSpeed_{Start}$ decreasing.

Example:

VelFilter = 0, then T_{Delay} = 0S.

VelFilter = 16, then $T_{Delay} \approx$ 0.05S.

VelFilter = 18, then $T_{Delay} \approx$ 0.1S.

VelFilter = 20, then $T_{Delay} \approx$ 0.2S.

VelFilter = 22, then $T_{Delay} \approx$ 0.4S.

6.3.17 KV – Motor Velocity Coefficient Register

Address: 0x0043

Operation: ReadWORD, nonvolatile

Bits	Name	Type	Value	Default	Description
0~15	KV	UINT16	1~65535	20	Read : Motor Velocity Coefficient, used to calculate velocity Write : Illegal

6.3.18 BusWDT – Bus Watchdog Timer Register

Address: 0x0060

Operation: ReadWORD/WriteWORD, nonvolatile

Bits	Name	Type	Value	Default	Description
0~15	BusWDT	UINT16	2~65535	0x8000	R/W : Bus Watchdog Timer Unit : mS

If BusWDT ≥ 0x8000, then bus watchdog timer is disabled.

If BusWDT < 0x8000, then driver will be auto paused if no bus command is received in BusWDT millisecond. Driver will auto cancel pause caused by bus.

6.3.19 Port – Port Register

Address: 0x0080

Operation: ReadWORD, volatile

Bits	Name	Type	Value	Default	Description
0	Step	Bit	0~1	0	Read : 0 STEP/QA Photo LED ON 1 STEP/QA Photo LED OFF Write : Illegal
1	Dir	Bit	0~1	0	Read : 0 DIR/QB Photo LED ON 1 DIR/QB Photo LED OFF Write : Illegal
2	Free	Bit	0~1	0	Read : 0 FR/EN Photo LED ON 1 FR/EN Photo LED OFF Write : Illegal
3~7	Reserved				
8	Ready	Bit	0~1	0	Read : 0 RDY OFF 1 RDY ON Write : Illegal
9~15	Reserved				

6.3.20 PortHiFlag – Port High Flag Register

Address: 0x0081
Operation: ReadWORD, volatile

Bits	Name	Type	Value	Default	Description
0	Step	Bit	0~1	0	Read : 0 Port. Step has no rise edge 1 Port. Step has rise edge Write : 0 No Act 1 Clear
1	Dir	Bit	0~1	0	Read : 0 Port. Dir has no rise edge 1 Port. Dir has rise edge Write : 0 No Act 1 Clear
2	Free	Bit	0~1	0	Read : 0 Port. Free has no rise edge 1 Port. Free has rise edge Write : 0 No Act 1 Clear
3~7	Reserved				
8	Ready	Bit	0~1	0	Read : 0 Port. Ready has no rise edge 1 Port. Ready has rise edge Write : 0 No Act 1 Clear
9~15	Reserved				

6.3.21 PortLoFlag – Port Low Flag Register

Address: 0x0082
Operation: ReadWORD, volatile

Bits	Name	Type	Value	Default	Description
0	Step	Bit	0~1	0	Read : 0 Port. Step has no fall edge 1 Port. Step has fall edge Write : 0 No Act 1 Clear
1	Dir	Bit	0~1	0	Read : 0 Port. Dir has no fall edge 1 Port. Dir has fall edge Write : 0 No Act 1 Clear
2	Free	Bit	0~1	0	Read : 0 Port. Free has no fall edge 1 Port. Free has fall edge Write : 0 No Act 1 Clear
3~7	Reserved				
8	Ready	Bit	0~1	0	Read : 0 Port. Ready has no fall edge 1 Port. Ready has fall edge Write : 0 No Act 1 Clear
9~15	Reserved				

6.3.22 PortExFlag – Port Flip Flag Register

Address: 0x0083
Operation: ReadWORD, volatile

Bits	Name	Type	Value	Default	Description
0	Step	Bit	0~1	0	Read : 0 Port. Step has no edge 1 Port. Step has edge Write : 0 No Act 1 Clear
1	Dir	Bit	0~1	0	Read : 0 Port. Dir has no edge 1 Port. Dir has edge Write : 0 No Act 1 Clear
2	Free	Bit	0~1	0	Read : 0 Port. Free has no edge 1 Port. Free has edge Write : 0 No Act 1 Clear
3~7	Reserved				
8	Ready	Bit	0~1	0	Read : 0 Port. Ready has no edge

					1 Port. Ready has edge Write : 0 No Act 1 Clear
9~15	Reserved				

6.3.23 CpuTemp – Driver CPU Temperature Register

Address: 0x0300
Operation: ReadWORD, volatile

Bits	Name	Type	Value	Default	Description
0~15	CpuTemp	INT16	-40~125	0	Read : Driver CPU Temperature, unit: ℃ Write : Illegal

6.3.24 SinkTemp – Driver Sink Temperature Register

Address: 0x0301
Operation: ReadWORD, volatile

Bits	Name	Type	Value	Default	Description
0~15	SinkTemp	INT16	-40~125	0	Read : Driver Sink Temperature, unit: ℃ Write : Illegal

7 Application Guide

7.1 Step Driving

C codes of pulse control for micro step driver are list below.

7.1.1 Fall Edge

```
void DirStepDown(BOOL bDir)  // DIR/STEP, STEP fall edge, moving 1 pulse
{
	if (bDir)
		PinDirQB = 1;	// Forwards
	else
		PinDirQB = 0;	// Backwards

	DelaynS(100);		// Delay 100nS
	PinStepQA = 0;		// STEP fall edge
	DelaynS(100);
	PinStepQA = 1;
}
```

7.1.2 Rise Edge

```
void DirStepUp(BOOL bDir)	// DIR/STEP, STEP rise edge, moving 1 pulse
{
	if (bDir)
		PinDirQB = 1;	// Forwards
	else
		PinDirQB = 0;	// Backwards

	DelaynS(100);		// Delay 100nS
	PinStepQA = 1;		// STEP rise edge
	DelaynS(100);
	PinStepQA = 0;
}
```

7.1.3 Fall/Rise Edge

```
void DirStepUpDown(BOOL bDir)	// DIR/STEP, STEP fall/rise edge, moving 1 pulse
```

```
{
    if (bDir)
        PinDirQB = 1;       // Forwards
    else
        PinDirQB = 0;       // Backwords

    DelaynS(100);           // Delay 100nS
    PinStepQA ^= 1;         // Step rise/fall edge
    DelaynS(100);
}
```

7.1.4 QEP

```
const unsigned char ucQEP[4] = {0, 1, 3, 2};

void DirStepQep(BOOL bDir)    // QEP, moving 1 pulse
{
    static unsigned char ucPos = 0;
    unsigned char ucQEPNow;

    if (bDir)
        ucPos ++;     // Forwards
    else
        ucPos --;     // Backwards

    ucQEPNow= ucQEP[ucPos &3];      // Conver position to QEP
    PinStepQA = ucQEPNow&1;         // Output QA
    PinDirQB = (ucQEPNow>>1)&1;     // Output QB
    DelaynS (100);                  // Delay 100nS
}
```

7.1.5 Step Driving Example

```
void main(void)
{
    int i;

    // Init PinStepQA, PinDirQB here

    for (i=0; i<10000; i++)  // Forwards 10000 pulse
    {
        DirStepDown(1);          // Forwards 1 pulse
        // DirStepUp (1);        // Forwards 1 pulse
        // DirStepUpDown (1); // Forwards 1 pulse
```

```
        // DirStepQep (1);        // Forwards 1 pulse
        DelaynS(1000000);         // Delay1mS, control speed
    }

    for (i=10000; i>0; i--)   // Backwards 10000 pulse
    {
        DirStepDown(0);           // Backwards 1 pulse
        // DirStepUp (0);         // Backwards 1 pulse
        // DirStepUpDown (0);     // Backwards 1 pulse
        // DirStepQep (0);        // Backwards 1 pulse
        DelaynS (1000000);        // Delay1mS, control speed
    }
}
```

7.2 Bus Driving

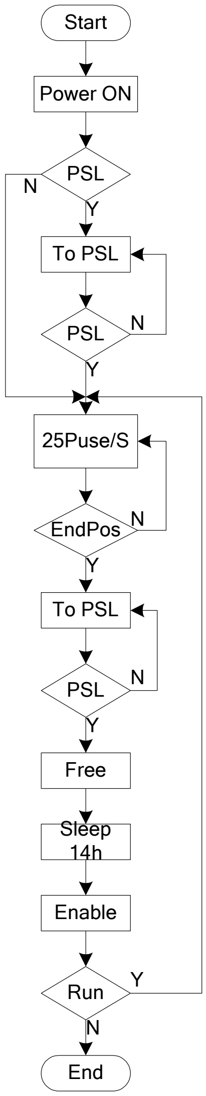
Modbus communication has uncertain factors, so bus control need command to replace pulse. C codes for Modbus driver are list below.

7.2.1 Communication Macro Instruction

Modbus-RTU/RS485 ADU can express as:

```
ReadWADU(ucAddr, wReg, wData)             : Read 1 WORD
WriteWADU(ucAddr, wReg, wData)            : Write 1 WORD
ReadWriteWADU(ucAddr, wReg, wData)        : Read/Write 1 WORD
ReadDWADU(ucAddr, wReg, dwData)           : Read 2 WORD
WriteDWADU(ucAddr, wReg, dwData)          : Write 2 WORD
ReadWriteDWADU(ucAddr, wReg, dwData)      : Read/Write 2 WORD
```

7.2.2 Solar System

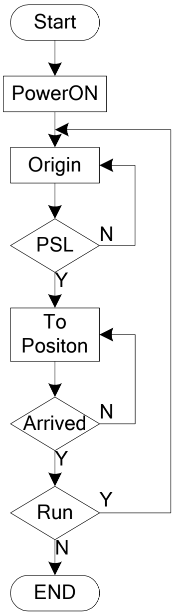


C codes for solar system are listed below.

```
void main(void)
{
    ...

    MovePSL(ucNodeAddr);                                    // Moving to PSL after power on
    WriteWADU(ucNodeAddr, AddrPositon, 0x00000000); // Position = 0

    while(1)
    {
      for (int i=0; i<10*3600; i++)
      {
        dwPos = i*25;                         // 25*60 = 1500Step/Min = 7.5rpm
        MovePos(ucNodeAddr, dwPos);     // Low speed moving
        Sleep(1000);                          // Wait 1000mS
      }
```

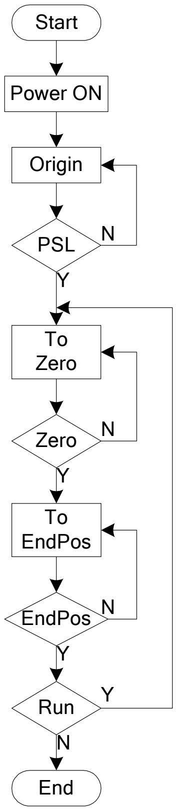
```
        MovePSL(unNodeAddr);// Fast moving to PSL
        Free(unNodeAddr);       // Free in the evening
        Sleep(14*3600*1000);    // Wait 14 hours
        Enable(unNodeAddr);     // Enable in the morning
    }
}

void MovePSL(BYTE ucNodeAddr)
{
    WORD wResult;

    // AHD8CXX moving to PSL position
    WriteWADU(ucNodeAddr, AddrControl, 0x0020);// Control.MovePSL = 1

    while(1)
    {
        ReadWADU(ucNodeAddr, AddrStatus, wResult); //wResult = Status
        if (!(wResult & 0x0002))     // Waiting to PSL position Control.MovePSL = 0
            break;
        Sleep(1000);                 // Wait 1000mS
    }
}

void MovePos(BYTE ucNodeAddr, DWORD dwPos)
{
    WriteWADU(ucNodeAddr, AddrPositionSet, dwPos);// PositionSet = dwPos
}

void Free(BYTE ucNodeAddr)
{
    WriteWADU(ucNodeAddr, AddrControl, 0x0000);//Control.Enable = 0
}

void Enable(BYTE ucNodeAddr)
{
    WriteWADU(ucNodeAddr, AddrControl, 0x0008);//Control.Enable = 1
}
```

7.2.3 Fast Shuttle System Calibrated Each Loop




```
void main(void)
{
    ...
    MovePSL(ucNodeAddr);                                          // Moving to PSL after power on
    WriteWADU(ucNodeAddr, AddrPositon, 0x00000000); // Position = 0

    while(1)
    {
        MovePSL(ucNodeAddr);              // Moving to PSL
        Sleep(100);                       // Delay 100mS
        MovePos(ucNodeAddr, 25*200);  // Forwards 25r
        Sleep(100);                       // Delay 100mS
    }

    ...
}
```

7.2.4 Fast Shuttle System Calibrated First Time




```
void main(void)
{
      ...

      MovePSL(ucNodeAddr);                                    // Moving to PSL after power on
      WriteWADU(ucNodeAddr, AddrPositon, 0x00000000); // Position = 0

      while(1)
      {
         MovePos(ucNodeAddr, 100); // Start point and PSL need proper distance
                                   // When position sensor is active,
                                   // moving to the position sensor is inhibitted
         Sleep(100);               // Delay 100mS
         MovePos(ucNodeAddr, 100+25*200);  // Forwards 25r
         Sleep(100);               // Delay 100mS
      }

      ...
}
```

8 Knowledge

8.1 Modbus-RTU/RS485

Modbus-RTU/RS485 protocol and standard can refer to relative documents and will not be discussed in this datasheet. The datasheet only introduces the content related with the driver.

8.1.1 Communication Bits Mode

AHD8CXX communication bits mode: 1 start bit, 8 data bits(Low bits ahead, High bits behind), 1 even parity bit, 1 stop bit.

Start	Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Even Parity	Stop

8.1.2 Big-Endian/Little-Endian

Big-Endian: for multibyte data, high bytes at low address, low bytes at high address.
Little-Endian: for multibyte data, high bytes at high address, low bytes at low address.

8.1.3 Check Mode

AHD8CXX communication use CRC check.
CRC XOR polynomial value is 0xA001(1010 0000 0000 0001).

8.1.4 ADU/PDU

ADU is application data units, PDU is protocol data units.

ADU				
RTU Address	Function Code	Register Address	Register Data	CRC
BYTE	BYTE	WORD	WORD/DWORD	WORD
	PDU			

Table 0-1

Register Address and Register Data are Big-Endian data type.

AHD8CXX support broadcast address 0 and 1~111 address.

AHD8CXX supported Function Codes are listed below:

Function Code	Function	Data Length
0x04	Read 1 Register/WORD	WORD
0x03	Read Muti Register/WORD	DWORD/QWORD
0x06	Write 1 Register/WORD	WORD
0x10	Write Muti Register/WORD	DWORD/QWORD

Table 0-2

8.1.5 Baud Rates and Communication Distance

Baud rates and communication distance is relevant with field application, typical data are list below:

Baudrate(bps)	Distance(m)
9600	1000
19200	1000
38400	1000
57600	800
115200	500
250000	250

Table 0-3

8.1.6 Communication Nodes

AHD8CXX support 111 bus nodes(Not include controller). How many nodes can be supported also relevant to field application.

8.1.7 Bus Connection

AHD8CXX connected by RJ45 socket of J5, do not use external T connection. Recommend use TECO/AMP's plug and AMP's 的 CAT5e/24AWG shielded twisted cable, shielded cable not shorted to driver frame.

8.1.8 Bus Terminal Resistor

Bus terminal resistor must be used at two ends of cable. If AHD8CXX is located at bus end, just use terminal resistor of AHD8CXX and switch J4[1..2] ON. If AHD8CXX is located at bus middle, switch J4[1..2] OFF.

Terminal resistor must be 2 counts and located at two ends. Only one node can connect shield to earth.

8.1.9 Bus Maintenance

AHD8CXX support bus hot plug. If a node is fault, just unplug the two RJ45 head of the node and connect the two head by a direct connector, then the bus can run without the fault node. After the fault node has been repaired or replaced, connect the node again.

8.2 RTU Frame Example

Suppose RTU address is 0x01, motor tooth counts is 50, xx represents CRC result bytes.

8.2.1 Reset

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	Control Address	HByte	0x00	3	Control Address	HByte	0x00
4		LByte	0x00	4		LByte	0x00
5	Control Data	HByte	0x00	5	Control Data	HByte	0x00
6		LByte	0x01	6		LByte	0x01
7	CRC	LByte	0x48	7	CRC	LByte	0x48
8		HByte	0x0A	8		HByte	0x0A

8.2.2 Reset Value

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	Control Address	HByte	0x00	3	Control Address	HByte	0x00
4		LByte	0x00	4		LByte	0x00
5	Control Data	HByte	0x00	5	Control Data	HByte	0x00
6		LByte	0x02	6		LByte	0x02
7	CRC	LByte	0x08	7	CRC	LByte	0x08
8		HByte	0x0B	8		HByte	0x0B

8.2.3 Free

Request Frame			Response Frame		
No.	Byte Name	Data	No.	Byte Name	Data
1	RTU Address	0x01	1	RTU Address	0x01

2	Function Code		0x06	2	Function Code		0x06
3	Control Address	HByte	0x00	3	Control Address	HByte	0x00
4		LByte	0x00	4		LByte	0x00
5	Control Data	HByte	0x00	5	Control Data	HByte	0x00
6		LByte	0x04	6		LByte	0x04
7	CRC	LByte	0x88	7	CRC	LByte	0x88
8		HByte	0x09	8		HByte	0x09

8.2.4 Enable

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	Control Address	HByte	0x00	3	Control Address	HByte	0x00
4		LByte	0x00	4		LByte	0x00
5	Control Data	HByte	0x00	5	Control Data	HByte	0x00
6		LByte	0x00	6		LByte	0x00
7	CRC	LByte	0x89	7	CRC	LByte	0x89
8		HByte	0xCA	8		HByte	0xCA

8.2.5 Pause

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	Control Address	HByte	0x00	3	Control Address	HByte	0x00
4		LByte	0x00	4		LByte	0x00
5	Control Data	HByte	0x00	5	Control Data	HByte	0x00
6		LByte	0x08	6		LByte	0x08
7	CRC	LByte	0x88	7	CRC	LByte	0x88
8		HByte	0x0C	8		HByte	0x0C

8.2.6 Continue

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	Control Address	HByte	0x00	3	Control Address	HByte	0x00
4		LByte	0x00	4		LByte	0x00

5	Control Data	HByte	0x00	5	Control Data	HByte	0x00
6		LByte	0x00	6		LByte	0x00
7	CRC	LByte	0x89	7	CRC	LByte	0x89
8		HByte	0xCA	8		HByte	0xCA

8.2.7 Moving to Low Position Sensor

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	Control Address	HByte	0x00	3	Control Address	HByte	0x00
4		LByte	0x00	4		LByte	0x00
5	Control Data	HByte	0x00	5	Control Data	HByte	0x00
6		LByte	0x10	6		LByte	0x10
7	CRC	LByte	0x88	7	CRC	LByte	0x88
8		HByte	0x06	8		HByte	0x06

8.2.8 Moving to High Position Sensor

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	Control Address	HByte	0x00	3	Control Address	HByte	0x00
4		LByte	0x00	4		LByte	0x00
5	Control Data	HByte	0x00	5	Control Data	HByte	0x00
6		LByte	0x20	6		LByte	0x20
7	CRC	LByte	0x88	7	CRC	LByte	0x88
8		HByte	0x12	8		HByte	0x12

8.2.9 Read Control

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x04	2	Function Code		0x04
3	Control	HByte	0x00	3	Register Bytes		0x02
4		LByte	0x00	4	Control	HByte	0x00
5	Register WORDs	HByte	0x00	5		LByte	0x08
6		LByte	0x01	6	CRC	LByte	0xB8
7	CRC	LByte	0x31	7		HByte	0xF6

8		HByte	0xCA	8			

8.2.10 Write InputType

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	InputType Address	HByte	0x00	3	InputType Address	HByte	0x00
4		LByte	0x08	4		LByte	0x08
5	InputType QEP	HByte	0x00	5	InputType QEP	HByte	0x00
6		LByte	0x02	6		LByte	0x02
7	CRC	LByte	0x89	7	CRC	LByte	0x89
8		HByte	0xC9	8		HByte	0xC9

8.2.11 Read InputType

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x04	2	Function Code		0x04
3	InputType Address	HByte	0x00	3	Register Bytes		0x02
4		LByte	0x08	4	InputType PSL+PSH	HByte	0x00
5	Register WORDs	HByte	0x00	5		LByte	0x08
6		LByte	0x01	6	CRC	LByte	0xB8
7	CRC	LByte	0xB0	7		HByte	0xF6
8		HByte	0x08	8			

8.2.12 Read CurrentMax

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x04	2	Function Code		0x04
3	CurrentMax Address	HByte	0x00	3	Register Bytes		0x02
4		LByte	0x10	4	I_{PMax}=6.50A	HByte	0x02
5	Register WORDs	HByte	0x00	5		LByte	0x8A
6		LByte	0x01	6	CRC	LByte	0x39
7	CRC	LByte	0x30	7		HByte	0xF7
8		HByte	0x0F	8			

8.2.13 Read CurrentMin

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
9	RTU Address		0x01	9	RTU Address		0x01
10	Function Code		0x04	10	Function Code		0x04
11	CurrentMin Address	HByte	0x00	11	Register Bytes		0x02
12		LByte	0x11	12	I_{PMin}=2.50A	HByte	0x00
13	Register WORDs	HByte	0x00	13		LByte	0xFA
14		LByte	0x01	14	CRC	LByte	0x39
15	CRC	LByte	0x61	15		HByte	0x73
16		HByte	0xCF	16			

8.2.14 Write Current

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	CurrentSet Address	HByte	0x00	3	CurrentSet Address	HByte	0x00
4		LByte	0x12	4		LByte	0x12
5	I_{PSet}=5.00A	HByte	0x01	5	I_{PSet}=5.00A	HByte	0x01
6		LByte	0xF4	6		LByte	0xF4
7	CRC	LByte	0x29	7	CRC	LByte	0x29
8		HByte	0xD8	8		HByte	0xD8

8.2.15 Read Current

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x04	2	Function Code		0x04
3	Current Address	HByte	0x00	3	Register Bytes		0x02
4		LByte	0x12	4	I_{PSet}=3.50A	HByte	0x01
5	Register WORDs	HByte	0x00	5		LByte	0x5E
6		LByte	0x01	6	CRC	LByte	0x39
7	CRC	LByte	0x91	7		HByte	0x58
8		HByte	0xCF	8			

8.2.16 Write CurrentLow

Request Frame	Response Frame

No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	CurrentLow Address	HByte	0x00	3	CurrentLow Address	HByte	0x00
4		LByte	0x13	4		LByte	0x13
5	Low to 30%	HByte	0x00	5	Low to 30%	HByte	0x00
6		LByte	0x1E	6		LByte	0x1E
7	CRC	LByte	0xF8	7	CRC	LByte	0xF8
8		HByte	0x07	8		HByte	0x07

8.2.17 Read CurrentLow

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x04	2	Function Code		0x04
3	CurrentLow Address	HByte	0x00	3	Register Bytes		0x02
4		LByte	0x13	4	Low to 30%	HByte	0x00
5	Register Counts	HByte	0x00	5		LByte	0x1E
6		LByte	0x01	6	CRC	LByte	0x39
7	CRC	LByte	0xC0	7		HByte	0x38
8		HByte	0x0F	8			

8.2.18 Write CurrentLowWT

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	CurrentLowWT Address	HByte	0x00	3	CurrentLowWT Address	HByte	0x00
4		LByte	0x14	4		LByte	0x14
5	Waiting 0.45S	HByte	0x01	5	Waiting 0.45S	HByte	0x01
6		LByte	0xC2	6		LByte	0xC2
7	CRC	LByte	0x49	7	CRC	LByte	0x49
8		HByte	0xCF	8		HByte	0xCF

8.2.19 Read CurrentLowWT

Request Frame		Response Frame			
No.	Byte Name	Data	No.	Byte Name	Data
1	RTU Address	0x01	1	RTU Address	0x01
2	Function Code	0x04	2	Function Code	0x04

3	CurrentLowWT Address	HByte	0x00	3	Register Bytes		0x02
4		LByte	0x14	4	Waiting 0.7S	HByte	0x02
5	Register WORDs	HByte	0x00	5		LByte	0xBC
6		LByte	0x01	6	CRC	LByte	0xB9
7	CRC	LByte	0x71	7		HByte	0xE1
8		HByte	0xCE	8			

8.2.20 Write Position

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x10	2	Function Code		0x10
3	Position Address	HByte	0x00	3	Position Address	HByte	0x00
4		LByte	0x20	4		LByte	0x20
5	Register WORDs	HByte	0x00	5	Register WORDs	HByte	0x00
6		LByte	0x04	6		LByte	0x04
7	Register Bytes		0x08	7	CRC	LByte	0xC0
8	Position is 10r from origin T = 50	HByte	0x00	8		HByte	0x00
9			0x00	9			
10			0x00	10			
11			0x00	11			
12			0x02	12			
13			0x49	13			
14			0xF0	14			
15		LByte	0x00	15			
16	CRC	LByte	0xA3	16			
17		HByte	0xAB	17			

8.2.21 Write Position Low DWORD

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x10	2	Function Code		0x10
3	Position Address	HByte	0x00	3	Position Address	HByte	0x00
4		LByte	0x20	4		LByte	0x20
5	Register WORDs	HByte	0x00	5	Register WORDs	HByte	0x00
6		LByte	0x02	6		LByte	0x02
7	Register Bytes		0x04	7	CRC	LByte	0x40
8	Position is 10r from origin T = 50	HByte	0x02	8		HByte	0x02
9			0x49	9			
10			0xF0	10			

11		LByte	0x00	11	
12	CRC	LByte	0x65	12	
13		HByte	0xD9	13	

8.2.22 Read Position

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x03	2	Function Code		0x03
3	Position Address	HByte	0x00	3	Register Bytes		0x08
4		LByte	0x20	4	Position is 10r from origin T = 50	HByte	0x00
5	Register WORDs	HByte	0x00	5			0x00
6		LByte	0x04	6			0x00
7	CRC	LByte	0x45	7			0x00
8		HByte	0xC3	8			0x02
9				9			0x49
10				10			0xF0
11				11		LByte	0x00
12				12	CRC	LByte	0x01
13				13		HByte	0xB9

8.2.23 Read Position Low DWORD

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x03	2	Function Code		0x03
3	Position Address	HByte	0x00	3	Register Bytes		0x04
4		LByte	0x20	4	Position is 10r from origin T = 50	HByte	0x02
5	Register WORDs	HByte	0x00	5			0x49
6		LByte	0x02	6			0xF0
7	CRC	LByte	0xC5	7		LByte	0x00
8		HByte	0xC1	8	CRC	LByte	0x6E
9				9		HByte	0x5D

8.2.24 Write PositionSet

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x10	2	Function Code		0x10

3	PositionSet Address	HByte	0x00	3	PositionSet Address	HByte	0x00
4		LByte	0x24	4		LByte	0x24
5	Register WORDs	HByte	0x00	5	Register WORDs	HByte	0x00
6		LByte	0x04	6		LByte	0x04
7	Register Bytes		0x08	7	CRC	LByte	0x81
8	PositionSet is 10r from origin T = 50	HByte	0x00	8		HByte	0xC1
9			0x00	9			
10			0x00	10			
11			0x00	11			
12			0x02	12			
13			0x49	13			
14			0xF0	14			
15		LByte	0x00	15			
16	CRC	LByte	0x52	16			
17		HByte	0x64	17			

8.2.25 Write PositionSet Low DWORD

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x10	2	Function Code		0x10
3	PositionSet Address	HByte	0x00	3	PositionSet Address	HByte	0x00
4		LByte	0x24	4		LByte	0x24
5	Register WORDs	HByte	0x00	5	Register WORDs	HByte	0x00
6		LByte	0x02	6		LByte	0x02
7	Register Bytes		0x04	7	CRC	LByte	0x01
8	PositionSet is 10r from origin T = 50	HByte	0x02	8		HByte	0xC3
9			0x49	9			
10			0xF0	10			
11		LByte	0x00	11			
12	CRC	LByte	0x64	12			
13		HByte	0x2A	13			

8.2.26 Read PositionSet

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x03	2	Function Code		0x03
3	PositionSet Address	HByte	0x00	3	Register Bytes		0x08
4		LByte	0x24	4	PositionSet is 10r from origin	HByte	0x00
5	Register	HByte	0x00	5			0x00

6	WORDs	LByte	0x04	6	T = 50		0x00
7	CRC	LByte	0x04	7			0x00
8		HByte	0x02	8			0x02
9				9			0x49
10				10			0xF0
11				11		LByte	0x00
12				12	CRC	LByte	0x01
13				13		HByte	0xB9

8.2.27 Read PositionSet Low DWORD

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x03	2	Function Code		0x03
3	PositionSet Address	HByte	0x00	3	Register Bytes		0x04
4		LByte	0x24	4	PositionSet is 10r from origin T = 50	HByte	0x02
5	Register WORDs	HByte	0x00	5			0x49
6		LByte	0x02	6			0xF0
7	CRC	LByte	0x84	7		LByte	0x00
8		HByte	0x00	8	CRC	LByte	0x6E
9				9		HByte	0x5D

8.2.28 Read TResolution

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x03	2	Function Code		0x03
3	TResolution Address	HByte	0x00	3	Register Bytes		0x04
4		LByte	0x28	4	TResolution Value	HByte	0x00
5	Register WORDs	HByte	0x00	5			0x01
6		LByte	0x02	6			0x2C
7	CRC	LByte	0x44	7		LByte	0x00
8		HByte	0x03	8	CRC	LByte	0xB7
9				9		HByte	0x33

8.2.29 Write PulseLength

Request Frame			Response Frame		
No.	Byte Name	Data	No.	Byte Name	Data
1	RTU Address	0x01	1	RTU Address	0x01

2	Function Code		0x10	2	Function Code		0x10
3	PulseLength Address	HByte	0x00	3	PulseLength Address	HByte	0x00
4		LByte	0x2A	4		LByte	0x2A
5	Register WORDs	HByte	0x00	5	Register WORDs	HByte	0x00
6		LByte	0x02	6		LByte	0x02
7	Register Bytes		0x04	7	CRC	LByte	0x60
8	1536	HByte	0x00	8		HByte	0x00
9			0x00	9			
10			0x06	10			
11		LByte	0x00	11			
12	CRC	LByte	0x72	12			
13		HByte	0x68	13			

8.2.30 Read PulseLength

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x03	2	Function Code		0x03
3	PulseLength Address	HByte	0x00	3	Register Bytes		0x04
4		LByte	0x2A	4	PulseLength	HByte	0x00
5	Register WORDs	HByte	0x00	5			0x00
6		LByte	0x02	6			0x06
7	CRC	LByte	0xE5	7		LByte	0x00
8		HByte	0xC3	8	CRC	LByte	0xF9
				9		HByte	0x93

8.2.31 Write PulsePosition

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x10	2	Function Code		0x10
3	PulsePosition Address	HByte	0x00	3	PulsePosition Address	HByte	0x00
4		LByte	0x2C	4		LByte	0x2C
5	Register WORDs	HByte	0x00	5	Register WORDs	HByte	0x00
6		LByte	0x02	6		LByte	0x02
7	Register Bytes		0x04	7	CRC	LByte	0x80
8	10000	HByte	0x00	8		HByte	0x01
9			0x00	9			
10			0x27	10			
11		LByte	0x10	11			
12	CRC	LByte	0xEB	12			

13		HByte	0xDE	13			

8.2.32 Write PulsePosition Low WORD

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	PulsePosition Address	HByte	0x00	3	PulsePosition Address	HByte	0x00
4		LByte	0x2C	4		LByte	0x2C
5	10000	HByte	0x27	5	10000	HByte	0x27
6		LByte	0x10	6		LByte	0x10
7	CRC	LByte	0x52	7	CRC	LByte	0x52
8		HByte	0x3F	8		HByte	0x3F

8.2.33 Read PulsePosition

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x03	2	Function Code		0x03
3	PulsePosition Address	HByte	0x00	3	Register Bytes		0x04
4		LByte	0x2C	4	PulsePosition 10000	HByte	0x00
5	Register WORDs	HByte	0x00	5			0x00
6		LByte	0x02	6			0x27
7	CRC	LByte	0x05	7		LByte	0x10
8		HByte	0xC2	8	CRC	LByte	0xE0
9				9		HByte	0x0F

8.2.34 Read PulsePosition Low WORD

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x04	2	Function Code		0x04
3	PulsePosition Address	HByte	0x00	3	Register Bytes		0x02
4		LByte	0x2C	4	PulsePosition 10000	HByte	0x27
5	Register WORDs	HByte	0x00	5		LByte	0x10
6		LByte	0x01	6	CRC	LByte	0xA3
7	CRC	LByte	0xF0	7		HByte	0x0C
8		HByte	0x03	8			

8.2.35 Write PulsePositionSet

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x10	2	Function Code		0x10
3	PulsePositionSet Address	HByte	0x00	3	PulsePositionSet Address	HByte	0x00
4		LByte	0x2E	4		LByte	0x2E
5	Register WORDs	HByte	0x00	5	Register WORDs	HByte	0x00
6		LByte	0x02	6		LByte	0x02
7	Register Bytes		0x04	7	CRC	HByte	0x21
8	10000	HByte	0x00	8		LByte	0xC1
9			0x00	9			
10			0x27	10			
11		LByte	0x10	11			
12	CRC	LByte	0x6A	12			
13		HByte	0x07	13			

8.2.36 Write PulsePositionSet Low WORD

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	PulsePositionSet Address	HByte	0x00	3	PulsePositionSet Address	HByte	0x00
4		LByte	0x2E	4		LByte	0x2E
5	10000	HByte	0x27	5	10000	HByte	0x27
6		LByte	0x10	6		LByte	0x10
7	CRC	LByte	0xF3	7	CRC	LByte	0xF3
8		HByte	0xFF	8		HByte	0xFF

8.2.37 Read PulsePositionSet

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x03	2	Function Code		0x03
3	PulsePositionSet Address	HByte	0x00	3	Register Bytes		0x04
4		LByte	0x2E	4	PulsePositionSet 10000	HByte	0x00
5	Register WORDs	HByte	0x00	5			0x00
6		LByte	0x02	6			0x27
7	CRC	LByte	0xA4	7		LByte	0x10

8		HByte	0x02	8	CRC	LByte	0xE0
				9		HByte	0x0F

8.2.38 Read PulsePositionSet Low WORD

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x04	2	Function Code		0x04
3	PulsePositionSet Address	HByte	0x00	3	Register Bytes		0x02
4		LByte	0x2E	4	PulsePosition Set = 10000	HByte	0x27
5	Register WORDs	HByte	0x00	5		LByte	0x10
6		LByte	0x01	6	CRC	LByte	0xA3
7	CRC	LByte	0x51	7		HByte	0x0C
8		HByte	0xC3	8			

8.2.39 Write VelSet

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	VelSet Address	HByte	0x00	3	VelSet Address	HByte	0x00
4		LByte	0x40	4		LByte	0x40
5	192	HByte	0x00	5	192	HByte	0x00
6		LByte	0xC0	6		LByte	0xC0
7	CRC	LByte	0x88	7	CRC	LByte	0x88
8		HByte	0x4E	8		HByte	0x4E

8.2.40 Read VelSet

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x04	2	Function Code		0x04
3	VelSet Address	HByte	0x00	3	Register Bytes		0x02
4		LByte	0x40	4	VelSet 192	HByte	0x00
5	Register Counts	HByte	0x00	5		LByte	0xC0
6		LByte	0x01	6	CRC	LByte	0xB9
7	CRC	LByte	0x30	7		HByte	0x60
8		HByte	0x1E	8			

8.2.41 Write VelStart

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	VelStart Address	HByte	0x00	3	VelStart Address	HByte	0x00
4		LByte	0x41	4		LByte	0x41
5	96	HByte	0x00	5	96	HByte	0x00
6		LByte	0x60	6		LByte	0x60
7	CRC	LByte	0xD9	7	CRC	LByte	0xD9
8		HByte	0xF6	8		HByte	0xF6

8.2.42 Read VelStart

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x04	2	Function Code		0x04
3	VelStart Address	HByte	0x00	3	Register Bytes		0x02
4		LByte	0x41	4	VelStart 96	HByte	0x00
5	Register Counts	HByte	0x00	5		LByte	0x60
6		LByte	0x01	6	CRC	LByte	0xB9
7	CRC	LByte	0x61	7		HByte	0x18
8		HByte	0xDE	8			

8.2.43 Write VelFilter

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	VelFilter Address	HByte	0x00	3	VelFilter Address	HByte	0x00
4		LByte	0x42	4		LByte	0x42
5	20	HByte	0x00	5	20	HByte	0x00
6		LByte	0x14	6		LByte	0x14
7	CRC	LByte	0x29	7	CRC	LByte	0x29
8		HByte	0xD1	8		HByte	0xD1

8.2.44 Read VelFilter

Request Frame	Response Frame

No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x04	2	Function Code		0x04
3	VelFilter Address	HByte	0x00	3	Register Bytes		0x02
4		LByte	0x42	4	VelFilter 20	HByte	0x00
5	Register Counts	HByte	0x00	5		LByte	0x14
6		LByte	0x01	6	CRC	LByte	0xB9
7	CRC	LByte	0x91	7		HByte	0x3F
8		HByte	0xDE	8			

8.2.45 Read KV

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x04	2	Function Code		0x04
3	KV Address	HByte	0x00	3	Register Bytes		0x02
4		LByte	0x42	4	KV 20	HByte	0x00
5	Register Counts	HByte	0x00	5		LByte	0x14
6		LByte	0x01	6	CRC	LByte	0xB9
7	CRC	LByte	0xC0	7		HByte	0x3F
8		HByte	0x1E	8			

8.2.46 Write BusWDT

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x06	2	Function Code		0x06
3	BusWDT Address	HByte	0x00	3	BusWDT Address	HByte	0x00
4		LByte	0x60	4		LByte	0x60
5	1000mS	HByte	0x03	5	1000mS	HByte	0x03
6		LByte	0xE8	6		LByte	0xE8
7	CRC	LByte	0x89	7	CRC	LByte	0x89
8		HByte	0x6A	8		HByte	0x6A

8.2.47 Read BusWDT

Request Frame			Response Frame		
No.	Byte Name	Data	No.	Byte Name	Data
1	RTU Address	0x01	1	RTU Address	0x01
2	Function Code	0x04	2	Function Code	0x04

3	BusWDT Address	HByte	0x00	3	Register Bytes		0x02
4		LByte	0x60	4	1000mS	HByte	0x03
5	Register Counts	HByte	0x00	5		LByte	0xE8
6		LByte	0x01	6	CRC	LByte	0xB9
7	CRC	LByte	0x31	7		HByte	0x8E
8		HByte	0xD4	8			

8.2.48 Read Port

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x04	2	Function Code		0x04
3	Port Address	HByte	0x00	3	Register Bytes		0x02
4		LByte	0x80	4	Port 0x0001	HByte	0x00
5	Register Counts	HByte	0x00	5		LByte	0x01
6		LByte	0x01	6	CRC	LByte	0x78
7	CRC	LByte	0x30	7		HByte	0xF0
8		HByte	0x22	8			

8.2.49 Read CPUTemperature

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x04	2	Function Code		0x04
3	CpuTemp Address	HByte	0x03	3	Register Bytes		0x02
4		LByte	0x00	4	22℃	HByte	0x00
5	Register Counts	HByte	0x00	5		LByte	0x16
6		LByte	0x01	6	CRC	LByte	0x38
7	CRC	LByte	0x31	7		HByte	0xFE
8		HByte	0x8E	8			

8.2.50 Read SinkTemperature

Request Frame				Response Frame			
No.	Byte Name		Data	No.	Byte Name		Data
1	RTU Address		0x01	1	RTU Address		0x01
2	Function Code		0x04	2	Function Code		0x04
3	SinkTemp Address	HByte	0x03	3	Register Bytes		0x02
4		LByte	0x01	4	38℃	HByte	0x00
5	Register	HByte	0x00	5		LByte	0x26

6	Counts	LByte	0x01	6	CRC	LByte	0x38
7	CRC	LByte	0x60	7		HByte	0xEA
8		HByte	0x4E	8			

8.3 Fieldbus Cable

CAT5E twisted pair connecting color standard T568A and T568B are listed below:

Pin / Stand.	1	2	3	4	5	6	7	8
T568A	WhiteGreen	Green	WhiteOrange	Blue	WhiteBlue	Orange	WhiteBrown	Brown
T568B	WhiteOrange	Orange	WhiteGreen	Blue	WhiteBlue	Green	WhiteBrown	Brown

Table 0-4

RJ45 connectors of AHD8CXX Modbus-RTU/RS485 use T568B standard, signals are list below.

AHD8CXX	Signal	NC	NC	RS485+	NC	NC	RS485-	GND	GND
	T568B	WhiteOrange	Orange	WhiteGreen	Blue	WhiteBlue	Green	WhiteBrown	Brown
	J5/RJ45	1	2	3	4	5	6	7	8

Table 0-5

Only one node can connect bus shield to earth, recommend every RJ45 connector do not connect bus shield to earth, AHD8CXX can work well in this mode.

9 Installation and Maintenance

9.1 Installation Preparation

1. Check driver and fitting status are good after opening packages.
2. Install the driver in the control cabinet that has good ventilation and little dusts.
3. Take the necessary measures when environment has inhibitive material to improve reliability and safety
4. Lock driver tightly in vibrative environment.

9.2 Mechanical Installation

1. Driver must keep 5cm space in each direction to ensure good heat dissipation.
2. Recommend installing driver on metal base, especially aluminum base.

9.3 Electrical Installation

1. Use thick wire to connect J1-6/ETH power earth.
2. Connect shield of motor cable to earth and driver J1-5/ETH.
3. Connect shield of control signal cable to earth at controller side.
4. Power cable maximum length is 50m.
5. AHD8C2XA power wire sectional area must greater than 0.2mm^2/A;
   AHD8C2XB power wire sectional area must greater than 0.4mm^2/A, and wire must stand 125℃.
6. Power cable and logic cable must keep proper distance.
7. Surface temperature of motor must be lower than 90℃ when running.

9.4 Test running

1. Check the wiring and setting before initial running.
2. When test on your machine, first increase the torque, then increase the speed.
3. After self-testing, set the driver to normal mode.
4. For better accuracy and lower noise, choose the appropriate motor and acceleration curves.

9.5 Daily Maintenance

1. Check the fan every month. Clean the fan when power off if the accumulation of dust is too heavy.
2. The driver need to be power on once at least 1 hour for every 3 months, to ensure the electronic components in good condition.

9.6 FAQ

FAQ	Solution
Motor has torque but not move	Check control signal voltage, current, time. If driver selftest is OK, then control wire or controller has faults
Motor has no torque or not move	Check Free/Enable wire or motor wire
Motor has little torque	Check current, velocity, filter setting
Motor stall when speed up	Check driver setting or change to large torque motor
Motor run at wrong speed	Check PulseLength or VelSet
Motor run illegally	Check load, connecting and control signal
Motor noise soundly	Check driver current setting and motor connecting
Run wrong direction	Exchange A+/A- or B+/B-
Driver temperature high	Check fan, clean dust
Motor running without control signal	Driver is in self-test mode, set to normal mode
Bus communication error	Check wires, address, baudrate and terminal resistor

10 Performance

10.1 Electrical Parameters

Performance is tested at T_A = +25℃

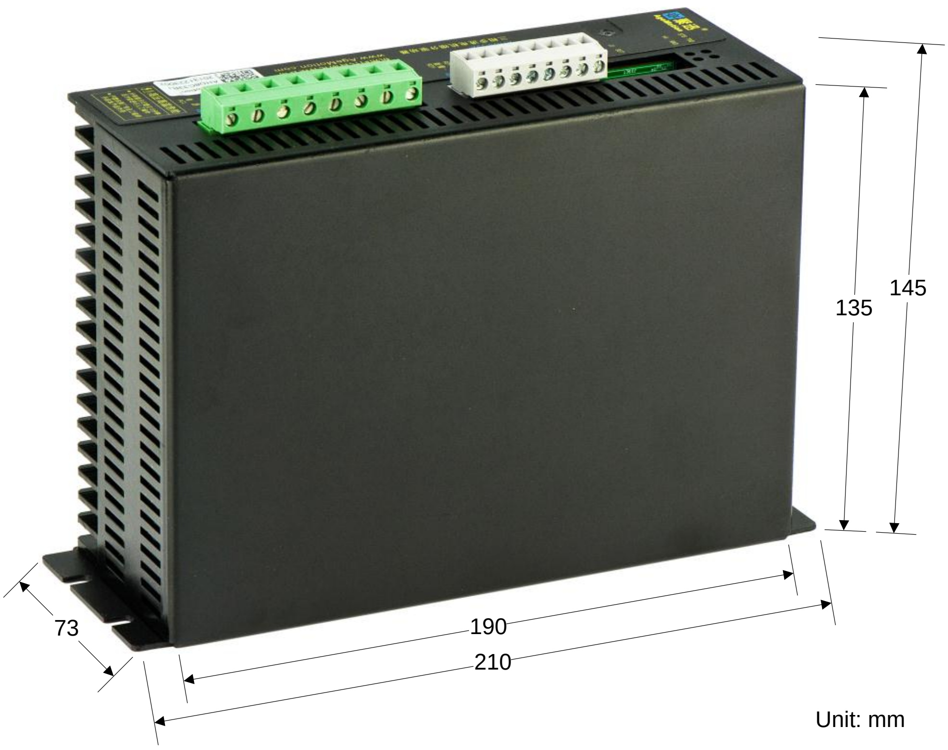
Parameters			Minimum	Typical	Maximum	Unit
Alternating Voltage			85		265	VAC
Direct Voltage			120		350	VDC
Alternating Frequency			47		63	Hz
Input Current	AHD8C21			1		A
	AHD8C22			2		
	AHD8C23			3		
	AHD8C24			4		
	AHD8C26			6.5		
	AHD8C29			9		
Input Power	AHD8C21				300	W
	AHD8C22				600	
	AHD8C23				900	
	AHD8C24				1200	
	AHD8C26				2000	
	AHD8C29				2700	
Idle Power(Driver)				30		
No Load Power				5.6		
Power Free				2.0		
Output Efficiency				90	97	%
Peak Current per Phase I_P	AHD8C21	interval 0.15A	0.60		1.65	A
	AHD8C22	interval 0.25A	1.00		2.10	
	AHD8C23	interval 0.25A	1.50		3.25	
	AHD8C24	interval 0.5A	2.50		4.25	
	AHD8C26	interval 0.5A	3.00		6.50	
	AHD8C29	interval 1.0A	5.50		9.00	
Undervoltage Protection				100		VDC
Overvoltage Protection				410		VDC
Current Low Percent				50		%
Current Low Waiting time				1		S
Insulation strength					500	V@1 min
Insulation Resister					500	MΩ
Optocoupler Input Current			10	13	20	mA
Control Signal	AHD8C2XXXG		3.3		5	V

Voltage	AHD8C2XXXH	5		24	
	AHD8C2XXXI	3.0	3.3	3.6	
	AHD8C2XXXJ	4	5	6	
	AHD8C2XXXK	10	12	14	
	AHD8C2XXXL	20	24	28	
Contact Current	AHD8C2XXP	4	5	10	mA
RDY Withstanding Voltage				30	V
RDY Driver Current				20	mA
RDY On Voltage@10mA			0.5		V
Power on to RDY Output		1.85	1.90	1.95	S
Pulse Frequency	Position Switch	0		10	KHz
	P+D（Fall or Rise Edge）	0	1	5	MHz
	P+D（Both Edge）	0	1	10	MHz
	QEP	0	1	20	MHz
	CW+CCW（Fall or Rise Edge）	0		150	KHz
FullStep Frequency		0		16000	FullStep/S
STEP/QA DIR/QB	t_H	100			nS
	t_L	100			nS
	t_s	50			nS
	t_h	50			nS
FR/EN	t_{s2}	20			uS
	t_{h2}	20			uS
Free/Enable Time			10		mS
Bolt Tightening Torque			0.4		Nm
Cooling Method			Fan Cooling		
Driver Temperature Rise			5		℃
AHD8C2XA	Working Temperature	-20		50	
	Storage Temperature	-20		70	
	CPU LowTemp Alarm		-40		
	CPU HighTemp Alarm		85		
AHD8C2XB	Working Temperature	-20		80	
	Storage Temperature	-40		85	
	CPU LowTemp Alarm		-40		
	CPU HighTemp Alarm		105		
	Sink LowTemp Alarm		-40		
	Sink HighTemp Alarm		105		
AHD8C2XA	Life@50℃		100000		Hour
	MTBF@50℃		1000000		
AHD8C2XB	Life@60℃		100000		
	Life@80℃		25000		
	MTBF@60℃		1000000		
	MTBF@80℃		100000		
Node Counts	AHD8CXXA			224	

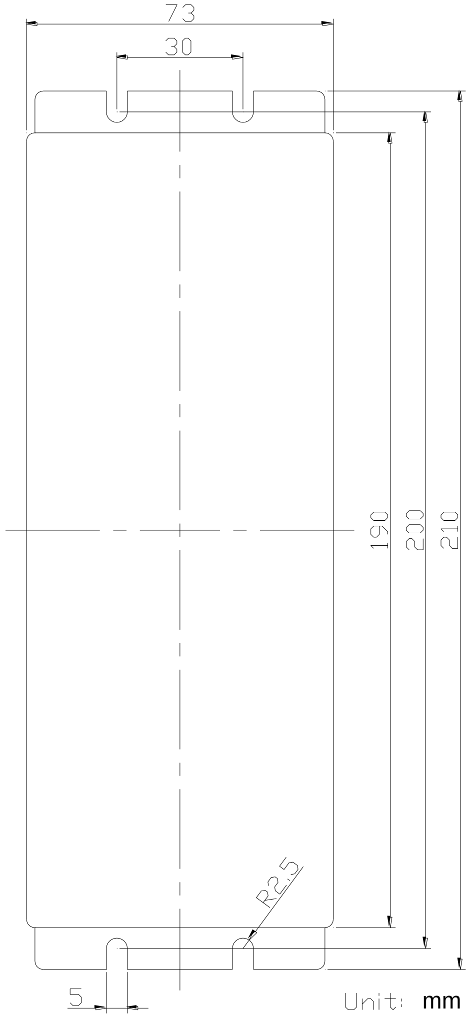
(Include controller CAT5e/24AWG)	AHD8CXXB			112	
Baud Rate(CAT5e/24AWG)		115	19200	250000	bps
Clock Precision				0.5	%
Response Time			1		mS
Communication Distance (CAT5e/24AWG)		0		1000	m
Characteristic Impedance(CAT5e/24AWG)		85	120	140	Ω
Working Humidity		15		85	%
Storage Humidity		15		85	
Vibration Acceleration				5	g
Impact Acceleration				10	
Height				1000	m
Forbidden Occasion		Corrosive, inflammable, explosive, conductive or electrostatic gas, liguid or dust, moisture environment			

10.2 Mechanical Parameters

Dimensions	210mm×135mm×73mm, 210mm×145mm×73m(Include Connectors)
Box	238mm×160mm×98mm(1PCS)
Carton	494mm×338mm×420mm(16Boxes)
Net Weight	1745g
Total Weight	1875g(Box), 31Kg(Carton)



10.3 Installation Dimensions



11 Support

11.1 Service

Thanks for choosing **AgeMotion** products. Each AgeMotion product has unique ID, guarantee period and other manufacturing information inside. The product has ID printed on label before sale, notice to protect it, not tear or pollute the label.

After service items:

1 Guarantee period:

AHD8CXXA guarantee period is 36 months after buying or 40 months after manufacturing. The product guarantee period is considered expired if either period expired.

AHD8CXXB guarantee period is 12 months after buying or 14 months after manufacturing. The product guarantee period is considered expired if either period expired.

Products will be serviced free in guarantee period, but service will not free and we have no responsibility for products when one of following items is fit:

A） Wrong use according to the datasheet.
B） Exceed the maximum ratings of the datasheet.
C） Disassembly, rebuilding or repairing without our authorization.
D） Product label lost, teared, damaged or polluted.
E） Product dropped, pressed, crashed, soaked, insolated or polluted.
F） Earthquake, fire, lightning strike and other irresistible disasters and conic disasters.

2 We will never service if disassembly, rebuilding or repairing without our authorization.

3 The maximum responsibility in any case for products in guarantee period is the value of products, we shoulder freight and customer shoulder insurance.

4 We have no responsibility for products exceed guarantee period, service will take fee, customer shoulder freight and insurance.

5 Customer shoulder freight and insurance if feedback product in guarantee period has no fault after our test.

6 We consider customers bought the product accept all items in the datasheet.

11.2 Company

Hangzhou AutomaticAge Co., Ltd. is located at Zhejiang University Science Park. We are professional manufacturer of NC software, NC system, NC pad and driver system.

Visit our web http://www.AgeMotion.com/ for details.

11.3 Contact Us

Manufacturer: Hangzhou AutomaticAge Co., Ltd.
Address : Room 213, Building A East, Zhejiang University Science Park,
No.525 Xixi Road, Hangzhou, China
Zip : 310013
Tel : 0086-571-85022190
Fax : 0086-571-85022590
Web : http://www.AgeMotion.com/
Email : Sales@AgeMotion.com（Recommend use personnel email）