Part 1 Vision Robotic Arm Introduction

1. Vision Robotic Arm Introduction

On the basis of JetAuto, JetAuto Pro is added **a 6DOF vision robotic arm** making it a **2-in-1 robot**.

It can **pick and sort stuffs**. In addition, **combined with Lidar**, it can implement **fixed-point navigation transport**, **follow line to clear obstacle** and many more AI games.

2. Structure Introduction

Vision robotic arm is loaded with five 35kg intelligent serial bus servos, one 20kg serial bus servo and one HD wide-angle camera.

Servo: 35kg*5 + 20kg*1 high voltage serial bus servos	Servo accuracy: 0.2°
Robotic arm material: aluminium alloy	Control method: UART serial port command
Robotic arm DOF: 4DOF + gripper	Baud rate: 115200
Grab weight with arm straightened: 200g	Storage: save data when power off
Grab weight during transporting: 500g	Angle read back function: support
Arm span: 410mm	Protection: avoid stalling and overheat
Reach: radius≤30cm	Data feedback: temperature, voltage and position
Camera	
Camera: HD wide-angle camera	Support: windows, Linux, Openwrt
Pixel: 300,000 pixels	Field of view: 120°
Resolution: 480p (640*480)	Frame rate: 30fps
Connection method: USB driver-free	Focus method: adjust focus manually

Part 2 Serial Bus Servo Introduction and Precaution

1. Serial Bus Servo Introduction

Serial bus servo is **derivative of digital servo**. **Different from traditional PWM servo**, **it adopts asynchronous serial communication** that is **servo can be controlled through sending and receiving command packs**, which **belongs to close-looped control.** Therefore, **multiple serial bus servos can be connected together bringing clean wiring and occupying less serial ports.**

Before controlling them to move, you need to **set their ID to tell them apart**, then you can **send command to control it**, for example, ID1 servo, rotate 30 degrees,ID2 servo, rotate 40 degrees

Serial bus servo adopts high-precision potentiometer which provides position, temperature and voltage feedback. With **high accuracy and good linearity**, this potentiometer has longer service life and enables robot to move stably.

In addition, **servos need to be centered before assembly.**

Center servo is to **make it rotate back to initial position**. **Serial bus servo rotates positive angle and negative angle taking servo mid-point as "zero point"**.

The software assumes servo mid-point as "**zero point**" since rotating parts drives potentiometer to move when servo is rotating, otherwise potentiometer will enter "**blind area**" and fail to work, which makes robot cannot reach specific angle and implement right action group.

Note: servos on JetAuto were centered and its IDs were set before delivery.

The serial bus servo communication protocol is stored in the same directory, and you can learn how servos communicate with each other.

2. Precautions

Servos on JetAuto were centered and its IDs were set before delivery, and you can skip this step.

1) Set servo IDs before using serial bus servos. Don't set repetitive ID when you use several servos at the same time.

2) For detailed instructions on how to set servo ID, please refer to the file in "**6. Jetson Nano and Jetson Nano Expansion Board Part->2. Jetson Nano Expansion Board Part->Part 12 Control Serial Bus Servo**"

3) Don't let servo be blocked or robot work at high load power, which leads to short service life and damage.

4) Keep away from robot before starting robot, otherwise you will get hurt.

Part 3 Bus Servo Communication Protocol

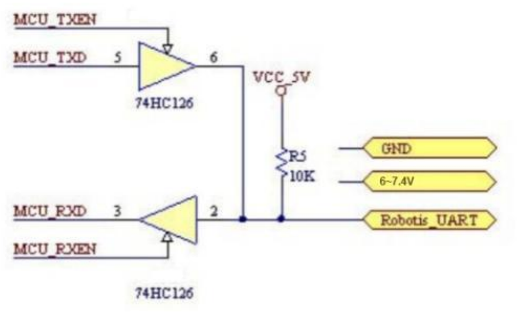
1. Summary

Using asynchronous serial bus communication method, theoretically, **up to 253 robot servos can be connected into chain through the bus**, you can **unify control them through the UART asynchronous serial interfaces.**
Each servo can be set to a different node address, **multiple servos can be unified or controlled independently**.
Communicating with the user's host computer software(controller or PC) through the asynchronous serial interface, you can set its parameters, function control.
Sending instructions to servo through the asynchronous serial interface, the servo can be set to the motor control mode or position control mode.
In the motor control mode, **servo can be used as a DC reduction motor with adjustable speed;**
In the position control mode, servo has 0-240 degrees of rotation range with Plus ± 30 ° deviation adjustable range, in this range with precise position control performance, speed adjustable.

Half-duplex UART asynchronous serial interface which conforms to the protocol can communicate with the servo and control servo in different ways.

2. UART Interface schematic diagram

Servo uses the program code to perform the timing control to the UART asynchronous serial interface, realizes the half-work asynchronous serial bus communication, the communication baud rate is 115200bps, and the interface is simple, the protocol is simplified. In your own designed controller, the UART interface for communication with the servo must be handled as shown below.



3. Command Packet

Command packet format

Table 1:

Header	ID number	Data Length	Command	Parameter	Checksum
0x55 0x55	ID	Length	Cmd	Prm 1... Prm N	Checksum

Header: **Two consecutive 0x55 are received indicating the arrival of data packets.**

ID：Each servo has an ID number. **ID number ranges from 0 ~ 253**, **converted to hexadecimal 0x00 ~ 0xFD**.

Broadcast ID: **ID No. 254 (0xFE) is the broadcast ID. If the ID number issued by the controller is 254 (0xFE)**, **all servos will receive instructions, but they all do not return the response message**, (except for reading the servo ID number, Please refer to the following instructions for detailed instructions) to prevent bus conflict.

Length(data): **Equal to the length of the data that is to be sent (including its own one byte). That is, the length of the data plus 3 is equal to the length of this command packet, from the header to the checksum.**

Command: **Control the various instructions of servo**, such as **position, speed control.**

Parameter: **In addition to commands, parameter are control information that need to add**.

Checksum:The calculation method is as follows:
Checksum=~(ID+ Length+Cmd+ Prm1+...PrmN) If the numbers in the brackets are calculated and exceeded 255,Then take the lowest one byte, "~" means Negation.

4. Command type

There are two kinds of commands, **write command** and **read command**.
Write command: normally, **it followed by parameters, write the parameters of the corresponding function into the servo to complete a certain action.**
Read command: normally, **it will not followed by parameters,** when the servo received “read command”, it will return the corresponding data immediately , the returned command value with parameters is the same as the "read command" value that sent to the servo. So the PC software must immediately prepare to change itself to “read condition” after it sends the reading command.

The following table is command that the PC software send to servos.

Command name	Command value	Length
SERVO_MOVE_TIME_WRITE	1	7
SERVO_MOVE_TIME_READ	2	3
SERVO_MOVE_TIME_WAIT_WRITE	7	7
SERVO_MOVE_TIME_WAIT_READ	8	3

SERVO_MOVE_START	11	3
SERVO_MOVE_STOP	12	3
SERVO_ID_WRITE	13	4
SERVO_ID_READ	14	3
SERVO_ANGLE_OFFSET_ADJUST	17	4
SERVO_ANGLE_OFFSET_WRITE	18	3
SERVO_ANGLE_OFFSET_READ	19	3
SERVO_ANGLE_LIMIT_WRITE	20	7
SERVO_ANGLE_LIMIT_READ	21	3
SERVO_VIN_LIMIT_WRITE	22	7
SERVO_VIN_LIMIT_READ	23	3
SERVO_TEMP_MAX_LIMIT_WRITE	24	4
SERVO_TEMP_MAX_LIMIT_READ	25	3
SERVO_TEMP_READ	26	3
SERVO_VIN_READ	27	3
SERVO_POS_READ	28	3
SERVO_OR_MOTOR_MODE_WRITE	29	7
SERVO_OR_MOTOR_MODE_READ	30	3
SERVO_LOAD_OR_UNLOAD_WRITE	31	4
SERVO_LOAD_OR_UNLOAD_READ	32	3
SERVO_LED_CTRL_WRITE	33	4
SERVO_LED_CTRL_READ	34	3
SERVO_LED_ERROR_WRITE	35	4
SERVO_LED_ERROR_READ	36	3

Command name:**Just for easy identification**, the user can also set according to their own habits. Command name suffix "_WRITE" which represents write command, and the suffix "_READ" represents read command.

Command value: That is, the command Cmd in command packet of Table 1

Length: that is the length(data length) in table 1

1. Command name: **SERVO_MOVE_TIME_WRITE** Command value:1 Length: 7

Parameter 1: lower 8 bits of angle value

Parameter 2: **higher 8 bits of angle value.range 0~1000. corresponding to the servo angle of 0 ~ 240 °, that means the minimum angle of the servo can be varied is 0.24 degree.**

Parameter 3: lower 8 bits of time value

Parameter 4: **higher 8 bits of time value. the range of time is 0~30000ms. When the command is sent to servo, the servo will be rotated from current angle to parameter angle at uniform speed within parameter time. After the command reaches servo, servo will rotate immediately**

2. Command name: **SERVO_MOVE_TIME_READ** Command value: 2 Length: 3

Read the angle and time value which sent by SERVO_MOVE_TIME_WRITE to the servo For the details of the command packet that the servo returns to host computer, please refer to the description of Table 4 below.

3. Command name: **SERVO_MOVE_TIME_WAIT_WRITE** Command value: 7 Length : 7

Parameter1: lower 8 bits of preset angle

Parameter2: **higher 8 bits of preset angle. range 0~1000. corresponding to the servo angle of 0 ~ 240 °. that means the minimum angle of the servo can be varied is 0.24 degree.**

Parameter3: lower 8 bits of preset time

Parameter4: **higher 8 bits of preset time. the range of time is 0~30000ms. The function of this command is similar to this “SERVO_MOVE_TIME_WRITE” command in the first point. But the difference is that the servo will not immediately turn when the command arrives at the servo,the servo will be rotated from current angle to parameter angle at uniform speed within parameter time until the command name SERVO_MOVE_START sent to servo**(command value of 11) **, then the servo will be rotated from current angle to setting angle at uniform speed within setting time**

4. Command name: **SERVO_MOVE_TIME_WAIT_READ** Command value: 8 Length: 3

Read the preset angle and preset time value which sent by SERVO_MOVE_TIME_WAIT_WRITE to the servo For the details of the command packet that the servo returns to host computer, please refer to the description of Table 3 below.

5. Command name: **SERVO_MOVE_START** Command value: 11 Length: 3

With the use of command SERVO_MOVE_TIME_WAIT_WRITE, described in point 3

6. Command name: **SERVO_MOVE_STOP** Command value: 12 Length: 3

When the command arrives at the servo, it will stop running immediately if the servo is rotating, and stop at the current angle position.

7. Command name: **SERVO_ID_WRITE** Command value: 13 Length: 4

Parameter 1: **The servo ID, range 0 ~ 253, defaults to 1. The command will re-write the ID value to the servo and save it even when power-down.**

8. Command name: **SERVO_ID_READ** Command value: 14 Length: 3

Read servo ID, For the details of the command package that the servo returns to host computer,please refer to the description of Table 4 below.

9. Command name: **SERVO_ANGLE_OFFSET_ADJUST** Command value: 17 Length: 4

Parameter 1: **servo deviation, range -125~ 125, The corresponding angle of -30 ° ~ 30 °**, when this command reach to the servo, **the servo will immediately rotate to adjust the deviation**.

Note 1:The adjusted deviation value is not saved when power-down by this command, if you want to save please refer to point 10.

Note 2: Because the parameter is “signed char” type of data, and the command packets to be sent are “unsigned char” type of data, **so before sending, parameters are forcibly converted to “unsigned char” data and then put them in command packet.**

10. Command name: **SERVO_ANGLE_OFFSET_WRITE** Command value: 18 Length: 3

Save the deviation value, and support “power-down save”. The adjustment of the deviation is stated in point 9

11. Command name: **SERVO_ANGLE_OFFSET_READ** Command value: 19 Length: 3

Read the deviation value set by the servo, For the details of the command packet that the servo returns to host computer, please refer to the description of Table 4 below.

12. Command name: **SERVO_ANGLE_LIMIT_WRITE** Command value: 20 Length: 7

Parameter 1: lower 8 bits of minimum angle

Parameter 2: higher 8 bits of minimum angle, range 0~1000

Parameter 3: lower 8 bits of maximum angle

Parameter 4: **higher 8 bits of maximum angle, range 0~1000** And the minimum angle value should always be less than the maximum angle value. **The command is sent to the servo, and the rotation angle of the servo will be limited between the minimum and maximum angle.** And the angle limit value supports ‘power-down save’.

13. Command name: **SERVO_ANGLE_LIMIT_READ** Command value: 21 Length: 3

Read the angle limit value of the servo, for the details of the instruction packet that the servo returns to host computer, please refer to the description of Table 4 below.

14. Command name: **SERVO_VIN_LIMIT_WRITE** Command value: 22 Length: 7

Parameter 1: lower 8 bits of minimum input voltage

Parameter 2: **higher 8 bits of minimum input voltage, range 4500~12000mv**

Parameter 3: lower 8 bits of maximum input voltage

Parameter 4: higher 8 bits of maximum input voltage, range 4500~12000mv And the minimum input voltage should always be less than the maximum input voltage. **The command is sent to the servo, and the input voltage of the servo will be limited between the minimum and the maximum.** If the servo is out of range, the led will flash and alarm (if an LED alarm is set). In order to protect the servo, the motor will be in the unloaded power situation, and the servo will not output torque and the input limited voltage value supports for power-down save.

15. Command name: **SERVO_VIN_LIMIT_READ** Command value: 23 Length: 3

Read the angle limit value of the servo, for the details of the instruction packet that the servo returns to host computer, please refer to the description of Table 4 below.

16. Command name: **SERVO_TEMP_MAX_LIMIT_WRITE** Command value: 24 Length: 4

Parameter 1: **The maximum temperature limit inside the servo range 50~100°C**, the default value is 85°C, if the internal temperature of the servo exceeds this value the led will flash and alarm (if an LED alarm is set). In order to protect the servo, the motor will be in the unloaded power situation, and the servo will not output torque until the temperature below this value of the servo, then it will once again enter the working state.and this value supports for power-down save.

17. Command name: **SERVO_TEMP_MAX_LIMIT_READ** Command value: 25 Length: 3

Read the maximum temperature limit value inside the servo, for the details of the command package that the servo returns to host computer, please refer to the description of Table 4 below.

18. Command name: **SERVO_TEMP_READ** Command value: 26 Length: 3
Read the Real-time temperature inside the servo, for the details of the instruction packet that the servo returns to host computer, please refer to the description of Table 4 below.

19. Command name: **SERVO_VIN_READ** Command value: 27 Length: 3

Read the current input voltage inside the servo, for the details of the instruction packet that the servo returns to host computer, please refer to the description of Table 4 below.

20. Command name: **SERVO_POS_READ** Command value: 28 Length: 3

Read the current angle value of the servo, for the details of the instruction packet that the servo returns to host computer, please refer to the description of Table 4 below.

21. Command name: **SERVO_OR_MOTOR_MODE_WRITE** Command value: 29 Length: 7

Parameter 1: **Servo mode, range 0 or 1, 0 for position control mode, 1 for motor control mode, default 0,**

Parameter 2: null value

Parameter 3: lower 8 bits of rotation speed value

Parameter 4: **higher 8 bits of rotation speed value. range -1000~1000**, **Only in the motor control mode is valid**, **control the motor speed**, **the value of the negative value represents the reverse, positive value represents the forward rotation.** Write mode and speed do not support power-down save.

Note: **Since the rotation speed is the "signed short int" type of data, it is forced to convert the data to "unsigned short int "type of data before sending the command packet.**

22. Command name: **SERVO_OR_MOTOR_MODE_READ** Command value: 30 Length: 3

Read the relative values of the servo, for the details of the command package that the servo returns to host computer, please refer to the description of Table 4 below.

23. Command name: **SERVO_LOAD_OR_UNLOAD_WRITE** Command value: 31 Length: 4

Parameter 1: **Whether the internal motor of the servo is unloaded power-down or not,** the range 0 or 1, **0 represents the unloading power down,** and **the servo has no torque output.** **1 represents the loaded motor**, then **the servo has a torque output**, the default value is 0.

24. Command name: **SERVO_LOAD_OR_UNLOAD_READ** Command value: 32 Length: 3

Read the state of the internal motor of the servo. for the details of the command package that the servo returns to host computer, please refer to the description of Table 4 below.

25. Command name: **SERVO_LED_CTRL_WRITE** Command value: 33 Length: 4

Parameter 1: **LED light/off state, the range 0 or 1, 0 represents that the LED is always on. 1 represents the LED off, the default 0, and support power-down save**

26. Command name: **SERVO_LED_CTRL_READ** Command value: 34 Length3

Read the state of the LED light. For the details of the command packet that the servo returns to host computer, please refer to the description of Table 4 below.

27. Command name: **SERVO_LED_ERROR_WRITE** Command value: 35 Length: 4

Parameter 1: **what faults will cause LED flashing alarm value**, **range 0~7** There are three types of faults that cause the LED to flash and alarm, regardless of whether the LED is in or off. The first fault is that internal temperature of the servo exceeds the maximum temperature limit (this value is set at point 16). The second fault is that the servo input voltage exceeds the limit value (this value is set at 14 points). The third one is when locked-rotor occurred.

This value corresponds to the fault alarm relationship as shown below:

0	No alarm
1	**Over temperature**
2	**Over voltage**
3	Over temperature and over voltage
4	**Locked-rotor(stalled)**
5	Over temperature and stalled
6	Over voltage and stalled
7	Over temperature , over voltage and stalled

28. Command name: **SERVO_LED_ERROR_READ** command value: 36 Length: 3

Read the servo fault alarm value. For the details of the command packet that the servo returns to host computer, please refer to the description of Table 4 below.

Table4

Command name	Command value	length
SERVO_MOVE_TIME_READ	2	7
SERVO_MOVE_TIME_WAIT_READ	8	7
SERVO_ID_READ	14	4
SERVO_ANGLE_OFFSET_READ	19	4
SERVO_ANGLE_LIMIT_READ	21	7
SERVO_VIN_LIMIT_READ	23	7
SERVO_TEMP_MAX_LIMIT_READ	25	4
SERVO_TEMP_READ	26	4
SERVO_VIN_READ	27	5
SERVO_POS_READ	28	5
SERVO_OR_MOTOR_MODE_READ	30	7
SERVO_LOAD_OR_UNLOAD_READ	32	4
SERVO_LED_CTRL_READ	34	4
SERVO_LED_ERROR_READ	36	4

Table 4 is **the command that servo return to the host computer**, **these commands will only return when host computer send a read command to servo,** what's more, the returned command value is consistent with the read command that the host computer sent to servo. the difference is that the returned value has parameters. The format of the returned data command packet is the same as the command package that the host computer sent to servo, as in Table 1.

1.Command name: **SERVO_MOVE_TIME_READ** Command value: 2 Length: 7

Parameter 1: lower 8 bits of angle value

Parameter 2: **higher 8 bits of angle, range 0~1000**

Parameter 3: lower 8 bits of time value

Parameter 4: **higher 8 bits of time value, range 0~30000ms**

2. Command name: **SERVO_MOVE_TIME_WAIT_READ** Command value: 8 Length: 7

Parameter 1: lower 8 bits of preset angle value

Parameter 2: **higher 8 bits of preset angle, range 0~1000**

Parameter 3: lower 8 bits of preset time value

Parameter 4: **higher 8 bits of preset time value, range 0~30000ms**

3. Command name: **SERVO_ID_READ** Command value: 14 Length: 4

Parameter 1: servo ID value, default value is 1

Description: ID read is a little bit special compared with other read commands , if the command packet ID is broadcast ID:254 (0xFE), the servo will return the response information, and other read commands will not return the response message when ID is broadcast ID. The purpose of this design is to inquiry the servo ID number via broadcast ID without knowing the ID number of the servo, but the limit is that the bus can only access a servo, or it will return data caused bus conflict.

4. Command name: **SERVO_ANGLE_OFFSET_READ** Command value: 19 Length: 4

Parameter 1: **The deviation set by the servo, range-125~125, default value is 0**

5. Command name: **SERVO_ANGLE_LIMIT_READ** Command value: 21 Length: 7

Parameter 1: lower 8 bits of minimum angle value

Parameter 2: **higher 8 bits of minimum angle, range 0~1000**

Parameter 3: lower 8 bits of maximum angle value

Parameter 4: **higher 8 bits of maximum angle value, range 0~1000,The default value is 0, the maximum angle is 1000**

6. Command name: **SERVO_VIN_LIMIT_READ** Command value: 23 Length: 7

Parameter 1: lower 8 bits of input voltage value

Parameter 2: **higher 8 bits of input voltage value ,range 6500~10000mv**

Parameter 3: lower 8 bits of maximum input voltage value

Parameter 4: **higher 8 bits of maximum input voltage value,range 6500~12000mv,The default value is 6500v, the maximum voltage is 12000**

7. Command name: **SERVO_TEMP_MAX_LIMIT_READ** Command value: 25Length: 4

Parameter 1: **The maximum temperature limit inside the servo, range 50~100°C, default value is 85°C**

8. Command name: **SERVO_TEMP_READ** Command value: 26 Length: 4

Parameter 1: **The current temperature inside the servo, no default value**

9. Command name: **SERVO_VIN_READ** Command value: 27 Length: 5

Parameter 1: lower 8 bits of current input voltage value

Parameter 2: higher 8 bits of current input voltage value, no default

10. Command name: **SERVO_POS_READ** Command value: 28 Length: 5

Parameter 1: lower 8 bits of current servo position value

Parameter 2: higher 8 bits of current servo position value, no default

Description: **Returned the angular position value need to be converted to the "signed short int" type of data, because the read angle may be negative**

11. Command name: **SERVO_OR_MOTOR_MODE_READ** Command value: 30 Length: 7

Parameter 1: **The current mode of the servo, 0 for the position control mode, 1 for the motor control mode, the default 0**

Parameter 2: Null, set to 0

Parameter 3: lower 8 bits of rotation speed value

Parameter 4: **higher 8 bits of rotation speed value ,range-1000~1000,Only valid in the motor control mode** , control the motor speed, the negative value represents the reverse, positive value represents the forward.

12. Command name: **SERVO_LOAD_OR_UNLOAD_READ** Command value: 32 Length: 4

Parameter 1: **Whether the internal motor of the servo is unloaded power-down or not, range 0 or 1, 0 represents the unloading power down, and now servo has no torque output. 1 represents the loaded motor, then the servo has a torque output, the default value of 0.**

13. Command name: **SERVO_LED_CTRL_READ** Command value: 34 Length: 4

Parameter 1: **LED light /off state, the range 0 or 1, 0 represents that the LED is always on. 1 represents LED is always off, the default 0.**

14. Command name: **SERVO_LED_ERROR_READ** Command value: 36 Length: 4

Parameter 1: **What faults in the servo cause LED flash and alarm, range 0~7. The corresponding relation between the numerical value and the fault is shown in table 3.**