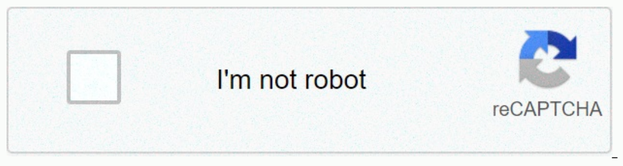

Continue

Dynamixel ax- 12a arduino library

Page 2 AX-12A Datasheet Library Installation Download zip file Zip extract Rename folder : AX-12A-Servo-Library-Master --> AX-12A-Servo-Library Place the folder in your library folder (usually documents/Arduino/Libraries) by using the library to use the library, including a header file located in the SRC folder. #include AX12A.h Library is provided with 4 Arduino examples : Blink : The most simplistic example of blinking a built-in LED move : Example shows how to set infiniteTurn goal position : Example of infinite wheel mode (wheel mode) read debugging A X-12A by printing all its registers you have 4 lines to configure : #define DirectionPin (10u) #define BaudRate (1000000ul) #define ID (1u) ... AX12A.begin(BaudRate, DirectionPin, & Tori); DirectionPin is a GPIO used to change the direction of communication. BaudRate corresponds to the baud rate used to communicate with the serb. Identifier corresponds to the Serb ID (from 0 to 253, 254 is used for transmission). & Serial refers to the serial port that you want to use. You may need to change it to &; SerialUSB or the clipboard package you are using (variable files). Change the AX-12A settings by connecting the serb to your computer using USB2Dynamixel. Thus, install the Dynamixel Wizard that is included in the RoboPlus software package. Follow these instructions to connect and access AX-12A settings. Baud rate tests we tested the library on different boards at the most common transmission rates. Some boards cannot reach the desired baud rate: address 0x04 Feather transmission rate M0 feather 32u4 nRF52 STM32L4 1 1 1000000 Certificate fail 3 500 Certificate fail certificate 4 4 4 400000 certificate fail certificate 7 250000 certificate 9 200000 certificate fail certificate 16 115200 in order of approval 34 5760 fine fail in order 103 19200 order 207 9600 fine page 2 AX-12A datasheet install library download zip file zip extract change folder name: AX-2 --servo-library-master --> AX-12A-servo-library Place the folder in your library folder (usually documents/Arduino/libraries) by using the library to use the library, Include a header file located in the src folder. #include AX12A.h Library is provided with 4 Arduino examples : Blink : The most simplistic example of blinking a built-in LED move : Example shows how to set infiniteTurn goal position : Example of infinite wheel mode (wheel mode) read debugging A X-12A by printing all its registers you have 4 lines to configure : #define DirectionPin (10u) #define BaudRate (1000000ul) #define ID (1u) ... AX12A.begin(BaudRate, DirectionPin, & Tori); DirectionPin is a GPIO used to change the direction of communication. BaudRate corresponds to the baud rate used to communicate with the serb. Identifier corresponds to the Serb ID (from 0 to 253, 254 is used for transmission). & Serial refers to the serial port that you want to use. You may need to change it to &; SerialUSB or the clipboard package you are using (variable files). Change the AX-12A settings by connecting the serb to your computer using USB2Dynamixel. So, install The wizard is included in the suite of Roboflos' software. Follow these instructions to connect and access AX-12A settings. Baud rate tests we tested the library on different boards at the most common transmission rates. Some boards cannot reach the desired baud rate: Address 0x04 Bud Feather Rate M0 Feather 32u4 nRF52 STM32L4 1 1000000 Certificate Certificate Fail 3 500000 Certificate 4 4 400000 Certificate Fail Order 7 250000 Certificate 2 9 20000 ok ok0000 OK fail order 16 115200 fine 34 57600 fine fail order order 103 19200 fine 207 9600 ok I decided to use some Dynamixel AX-12A engines for the project, and eventually there is a directory code for them, So I thought I'd share what I've learned. Despite being a little more expensive, this engine has some advantages over the more common TowerPro engines. Mostly, Dynamixel engines are more powerful, daisy-chainable, and have a robust control system with feedback that reports location, temperature, torque, etc. There are two different guides for AX-12A engines online. This 2006 guide is a little outdated: some specification values are obsolete (operating voltage range, for example), and some of the initial values reported in the control table are incorrect, but it goes into details about how to send instructions to the engine and how to read its response. This other guide includes more accurate values for initial specifications and conditions, but some of the details of the communication protocol are missing. It was useful to have both. BioloidAX-12(english).pdf resembles other servots, Dynamixel does not respond to PWM signals, but a slightly more complicated protocol of instructions for reading and writing on its memory. This communication occurs over a semi-double-sided UART port, using only one wire to send and receive. What this means is that we need to build a small circuit that converts full duplex into half duplex, whether we want to use Raspberry Pi or Arduino (or another microcont controller with a full duplex serial interface) to control these engines. The AX-12 guide from 2006 recommends this circuit:It's basically a tristate hoarding kit for bus arbitration; It makes sure that when the controller transmits, the bus is not connected to pin Rx, and that when it expects to receive, it is not driven by the Tx pin. Instead of using 74HC126 and 74HC04, I used 74LS241 (as recommended here), because it already has the built-in ability of operating half of its buffers with high signals, and the other half with a low signal. The drawing of the circle, which I eventually used and simple PCB design are top 123D.circuits.From the blog oppedijk:set configuration parameters /boot/config.txt: init_uart_clock = 16000000 secret stty -F/dev/ttyAMA0 1000000 edit /boot/cmdline.txt and remove all options reminiscent of ttyAMA0. Edit /etc/inittab and comment on all lines reminiscent of ttyAMA0, especially the getty one. RebootUPDATE for Rasabian Jesse: Follow this so thread to disable tty console and receive ttyAMA.I first checked the circuit using Arduino and the library was found here. But, because I eventually had to connect my project to the Internet, and track its situation between reboots, I decided to use raspberry pies as cattle instead. I started testing this engine control library, but then I decided to rewrite it to make it more object-oriented, and have some of the same capabilities as the Ardoino Library. The AX-12A python library created for Raspberry Pi is on github. Turns out the timing between sending an order and getting its response is pretty critical and sensitive. A certain amount of time has been spent changing lag values to minimize the number of commands dropped, but I feel there may still be problems with this aspect of the library. I'm still not sure what causes some commands to never reach the engines (this may have something to do with the timing of the Rx/Tx directional type), but for now I can decrease the number of missed commands by catching a time-out exception in the Python serial library, and re-denying the command. Go to the topic: Dynamixel Servo Library (read 53050 times) Previous topic - Next topic 08-06-2017 #1 I'm new to this forum. I followed some tutorials online to get my Dynamicsal working. I used a buffer and I got a -1 to respond from Dynamicsal. To ensure whether my repository works fine or not I also tested the circuit without buffering by combining TX and RX into an arduino data pin and then connecting it to a data pin of Dynamicsal. These are tutorials I've been following. (I apologize if linking to other sites is against the rules of the forums.) The circle and reservoir I used are just like the one in the tutorial. By the way I only used repository + arduino + dynamicsal no more module or things. this github page was also written for the MX series. It worked to transfer data but not to get data. appreciate your help. 08-06-2017 #2, may help get more information, like what do you use to drive a servo? An example of what kind of Ardoino? How exactly do you have it attached? Pictures help! Also how do you have a servo turned on? Do you have any sample code that you tried to run? Which version of Arduino IDE do you use? Note: I didn't use any of the libraries or tutorials mentioned in your publication. I used the bioloid code that Troussen used with their kits targeting their Arbotix-m board. I also made my own version of the bioloid library which I did work regardless of arbotix code. with this you can switch in the stream* to the AX init function. This option should be for one of the hardware serial ports. There are some programs around that use this, including my test program that is up to: again no Otherwise I can offer without further information. 08-07-2017 #3 checked the code on both mega2560 and uno. I used a sample code with the library called DynamixelSerial.zip (arduino ide ---> -----> serial dynamixel libraries ----> 3 sample code available) You can download it here : and on diagram circuit : In a circle above arduino's vin pin used but I used a separate 12v adapter. I get the message serially that voltage is -1 ..... (It's kind of an error code) I'm going to change the baud rate and check the ID if I do put it right in the code... Somehow diagnose the problem./ And take a look at the library you mentioned. I read cluster forums about dynamicsal. This forum is really cool. Thank you. Last edited by agha; 08-07-2017 at 12:16 p.m. 08-07-2017 #4 if you can write servo, than the Baud lesson is ok. I don't have time to debug their library. But a quick look at functions like: code: int DynamixelClass::readVoltage (unsigned character ID) { Chom check = (~(ID + AX_VOLT_LENGTH + AX_READ_DATA + AX_PRESENT_VOLTAGE + AX_BYTE_READ))&0xFF; switchCom(Direction_Pin,Tx_MODE); sendData(AX_START); sendData(AX_START); sendData(ID); sendData(AX_VOLT_LENGTH); sendData(AX_READ_DATA); send data(AX_PRESENT_VOLTAGE AX_BYTE_READ); send data (test test); delay(TX_DELAY_TIME); switchCom(Direction_Pin,Rx_MODE); ... I really don't like a code like this that tries to make timing like delayus(...) to try to get it right. You can play with the suspension time to try to make it work. It's best instead to make sure that the data you're outputing has finished being output before changing the direction pin... Assuming you have a fairly modern version of Arduino IDE > 1.5.x, then you can do so using Serial.flush(); Again assuming the Sarryle bone... And not Tory1, or Tory2... In Mega. As SendData is defined as: #define sendData(args) (Serial.write(args)) // Write Over Serial So for an experiment try changing such function and replacing the delay in the U.S. with Serial.flush() and seeing if you have better luck 08-08-2017 #5 Thanks to kurt iEck replay assumes this case is not a type of troubleshooting. Since im ok by sending data. Did you happen to use dynamicsal with arm microcont controls like stm32? 08-08-2017 #6 I mainly use tiny panels: www.pjrc.com where T3.6/T3.5 are M4f arm, T3.2 is M3 arm and T-LC is M0 M0 arm