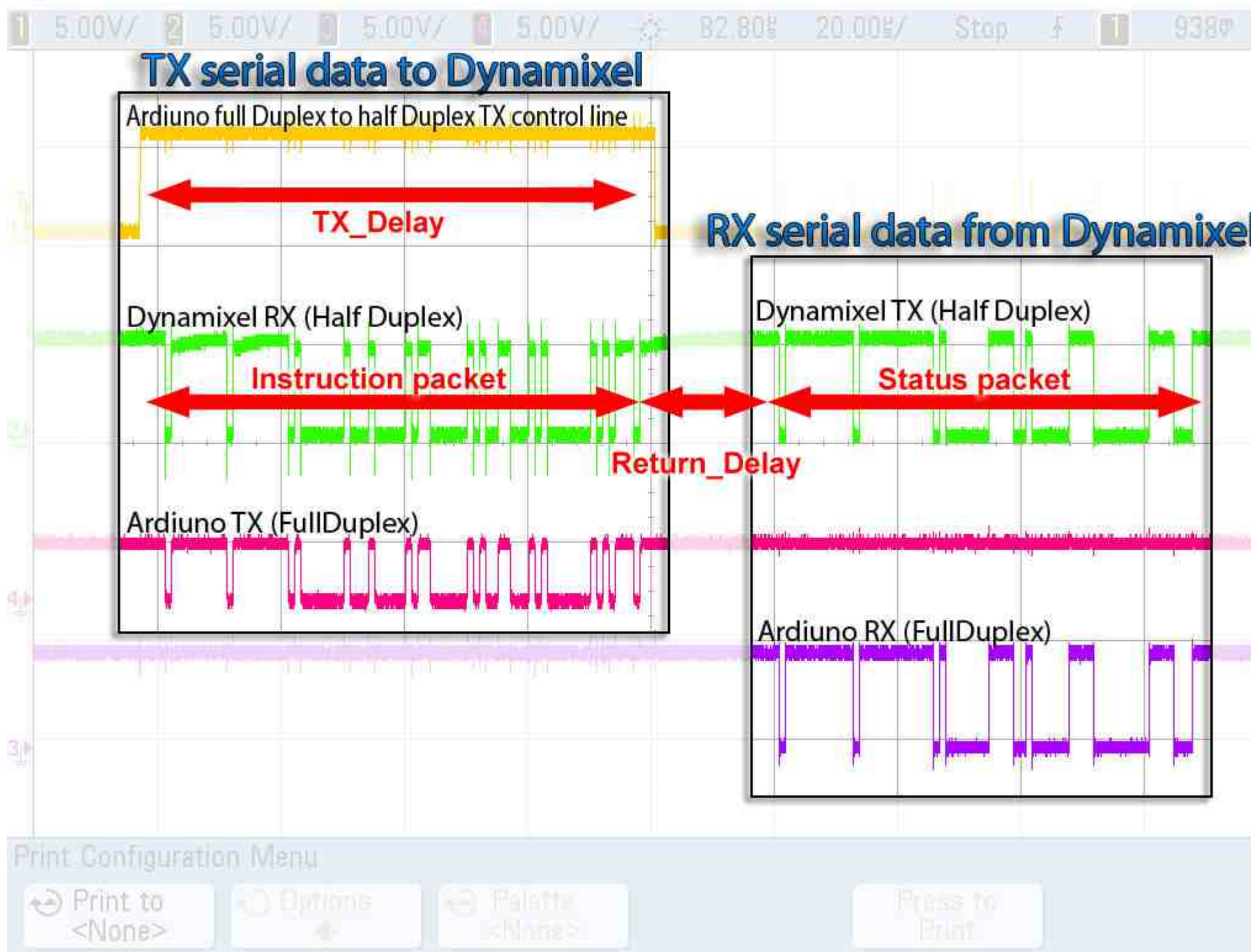


Arduino institution	Function	Parameters	Returns																										
Dynamixel.begin(Baud_rate , Control_Pin)	Setup Arduino to communicate with Dynamixel servo, this instruction must be run at lest once	<i>Baud_rate:</i> this is the speed at which the Arduino will be set and must be the same as the Dynamixel <i>Control_Pin:</i> the output pin on the Arduino used to control the full to half duplex buffer chip.	Error code (See table below) 0 = non																										
Dynamixel.end																													
Dynamixel.reset(Servo_ID)	Reset the Dynamixel servo back to factory default	<i>Servo_ID:</i> Servo ID of which the institution will be addressed too	Error code (See table below) 0 = non																										
Dynamixel.ping(Servo_ID)	Ping the Dynamixel servo and receive status packet from Dynamixel servo	<i>Servo_ID:</i> Servo ID of which the institution will be addressed too	Returns = Servo ID of Pinged servo Error code (See table below)																										
Dynamixel.setStatusPaketReturnDelay(Servo_ID , Return_Delay)	The delay between Dynamixel servo receiving an instruction packet from the Arduino and the Dynamixel servo returning(sending) status packet.	<i>Servo_ID:</i> Servo ID of which the institution will be addressed too <i>Return_Delay:</i> Return delay value (in uSec) (Dynamxiel Default Value = 500 uSec)	Returns = Servo ID if setStatusPacket is set to ALL Error code (See table below)																										
Dynamixel.setBaudRate(Servo_ID , Baud_Rate)	Set a new baud rate speed on the Dynamixel servo	<i>Servo_ID:</i> Servo ID of which the institution will be addressed too <i>Baud_Rate:</i> New Baud rate to which Dynamixel will be set <table><tr><th>Target BPS</th><th>Error Between Dynamixel and Arduino</th></tr><tr><td>3000000</td><td>0.000 %</td></tr><tr><td>2500000</td><td>0.000 %</td></tr><tr><td>2250000</td><td>0.000 %</td></tr><tr><td>1000000</td><td>0.000 %</td></tr><tr><td>500000</td><td>0.000 %</td></tr><tr><td>400000</td><td>0.000 %</td></tr><tr><td>250000</td><td>0.000 %</td></tr><tr><td>200000</td><td>0.000 %</td></tr><tr><td>115200</td><td>-2.124 %</td></tr><tr><td>57600 (Dynamixel Default)</td><td>0.794 %</td></tr><tr><td>19200</td><td>-0.160 %</td></tr><tr><td>9600</td><td>-0.160 %</td></tr></table>	Target BPS	Error Between Dynamixel and Arduino	3000000	0.000 %	2500000	0.000 %	2250000	0.000 %	1000000	0.000 %	500000	0.000 %	400000	0.000 %	250000	0.000 %	200000	0.000 %	115200	-2.124 %	57600 (Dynamixel Default)	0.794 %	19200	-0.160 %	9600	-0.160 %	Returns = Servo ID if setStatusPacket is set to ALL Error code (See table below)
Target BPS	Error Between Dynamixel and Arduino																												
3000000	0.000 %																												
2500000	0.000 %																												
2250000	0.000 %																												
1000000	0.000 %																												
500000	0.000 %																												
400000	0.000 %																												
250000	0.000 %																												
200000	0.000 %																												
115200	-2.124 %																												
57600 (Dynamixel Default)	0.794 %																												
19200	-0.160 %																												
9600	-0.160 %																												
Dynamixel.setMaxTorque(Servo_ID , Torque)	Set max torque value which when reached the Dynamixel will alarm and shut down	<i>Servo_ID:</i> Servo ID of which the institution will be addressed too <i>Torque:</i> value of max torque (value range 0x00 to 0x3FF) (Dynamxiel Default Value = 0x3FF)	Returns = Servo ID if setStatusPacket is set to ALL Error code (See table below)																										
Dynamixel.setHoldingTorque(Servo_ID , Bool)	Set holding torque	<i>Servo_ID:</i> Servo ID of which the institution will be addressed too <i>Bool:</i> ON = Holding torque true,	Returns = Servo ID if setStatusPacket is set to ALL Error code (See table below)																										

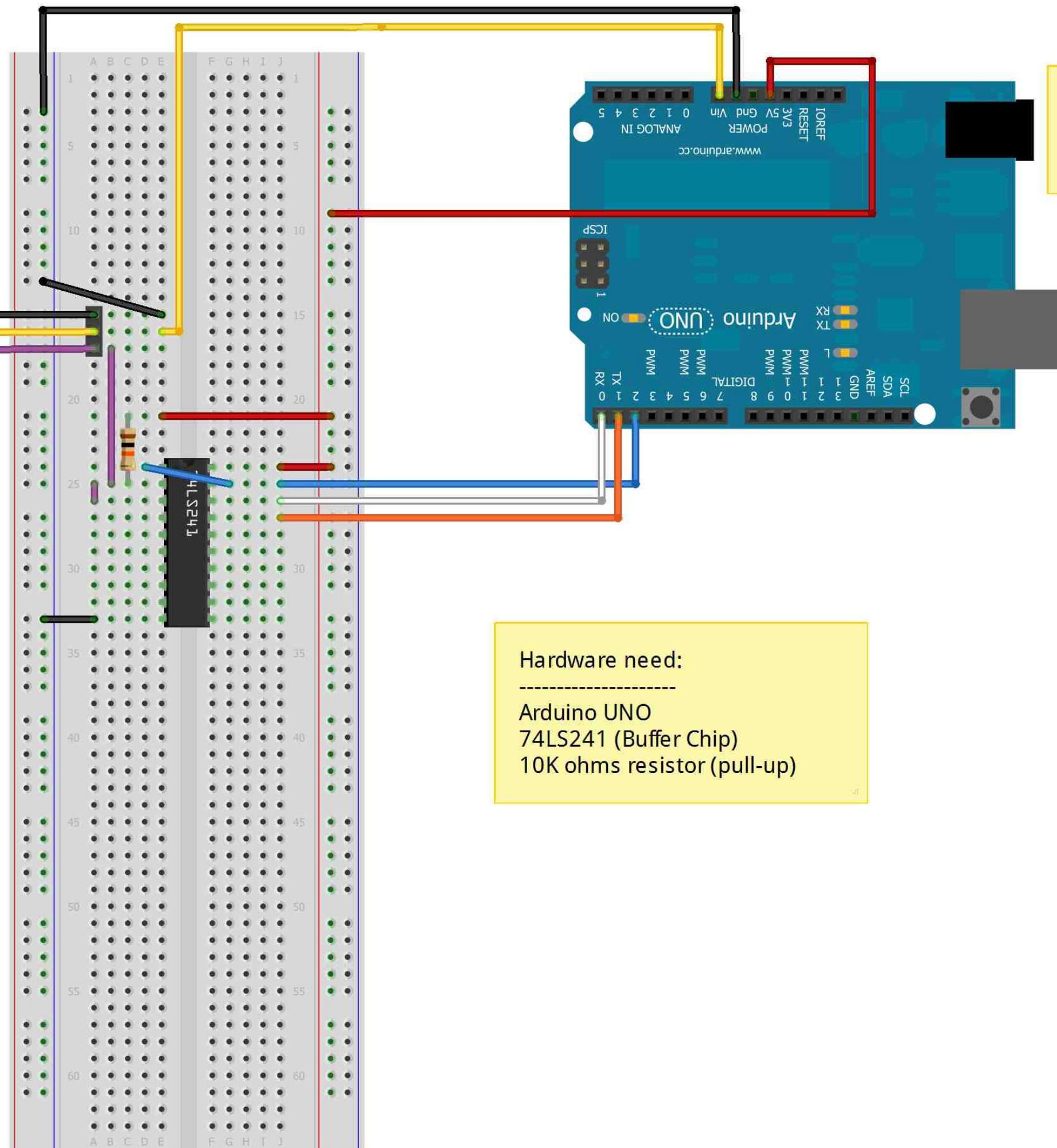
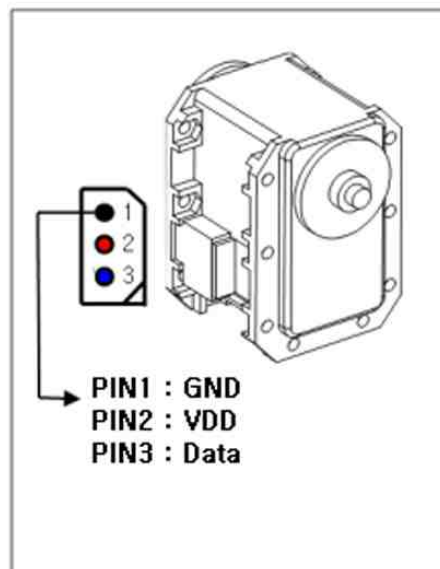
		OFF = Holding torque false	
Dynamixel.setAlarmShutdown(Servo_ID, Alarms)	Set which alarms for the Dynamixel to monitor	Servo_ID: Servo ID of which the institution will be addressed too	Returns = Servo ID if setStatusPacket is set to ALL
		Alarms: which alarm bits will be set (see Robotis manual for bit define)	Error code (See table below)
Dynamixel.setStatusPaket(Servo_ID, Status)	Set when the Dynamixel will or will not return a status packet	Servo_ID: Servo ID of which the institution will be addressed too	Returns = Servo ID if setStatusPacket is set to ALL
		Status: NONE = no status packet are returned from Dynamixel. READ = only read instructions return a status packet. ALL = all instructions return a status packet	Error code (See table below)
Dynamixel.setMode(Servo_ID,bool,CW_Limit,CCW_Limit)	Set servo to “wheel” mode or “servo” mode and set angel limits for servo mode	Servo_ID: Servo ID of which the institution will be addressed too	Returns = Servo ID if setStatusPacket is set to ALL
		Bool: WHEEL = wheel mode. SERVO = servo mode.	Error code (See table below)
		CW_Limit: angel limit for clockwise movement (value range 0x01 to 0xFFFF) one unit is about 0.088 degrees e.g 0xFFFF x 0.088 is about 360 degrees	
		CCW_Limit: angel limit for anticlockwise movement (value range 0x01 to 0xFFFF) one unit is about 0.088 degrees e.g 0xFFFF x 0.088 is about 360 degrees	
Dynamixel.setPunch(Servo_ID, Punch)	Set Punch value	Servo_ID: Servo ID of which the institution will be addressed too Returns = Servo ID if setStatusPacket is set to ALL	Returns = Servo ID if setStatusPacket is set to ALL
		Error code (See table below)	Error code (See table below)
		Punch: Punch value	
Dynamixel.setPID(Servo_ID, P_Gain, I_Gain, D_Gain)	Set PID settings	Servo_ID: Servo ID of which the institution will be addressed too	Returns = Servo ID if setStatusPacket is set to ALL
		P_Gain: Proportional ban	
		I_Gain: Integral action	Error code (See table below)
		D_Gain: Derivative action	
Dynamixel.set.Temp(Servo_ID,Temp)	Set highest temperature alarm limit	Servo_ID: Servo ID of which the institution will be addressed too	Returns = Servo ID if setStatusPacket is set to ALL
		Temp: Temperature value in Celsius	Error code (See table below)
Dynamixel.setVoltage(Servo_ID, Lowest_Volt, Highest_Volt)	Set lowest and highest voltage alarm limits	Servo_ID: Servo ID of which the institution will be addressed too	Returns = Servo ID if setStatusPacket is set to ALL
		Lowest_Volt: Low voltage limit	
		Highest_Volt: High voltage limit	Error code (See table below)
Dynamixel.servo(Servo_ID, Angle, Speed)	Move in servo mode to a specific angle at speed “x”	Servo_ID: Servo ID of which the institution will be addressed too	Returns = Servo ID if setStatusPacket is set to ALL
		Angle: Angle to move servo too (value range 0x00 to 0xFFFF) one unit is about 0.088 degrees e.g 0xFFFF x 0.088 is about 360 degrees	Error code (See table below)
		Speed: Speed to move at (value range 0x00 to 0x3FF) one unit is about 0.114rpm	

		e.g 0x3FF x 0.114 is about 117.07rpm	
Dynamixel.servoPreload(Servo_ID, Angle, Speed)	Write to Dynamixel register the servo mode to a specific angle at speed “x” <i>NOTE: this conman only writes(stores) the instruction on the Dynamixel, you must then use “Dynamixel.action” to tell it to action the information in the Dynamixel register.</i>	Servo_ID: Servo ID of which the institution will be addressed too	Returns = Servo ID if setStatusPacket is set to ALL
		Angle: Angle to move servo too (value range 0x00 to 0xFF) one unit is about 0.088 degrees e.g 0xFF x 0.088 is about 360 degrees	Error code (See table below)
		Speed: Speed to move at (value range 0x00 to 0x3FF) one unit is about 0.114rpm e.g 0x3FF x 0.114 is about 117.07rpm	
Dynamixel.wheel(Servo_ID, Direction, Speed)	Move in wheel mode left/right at speed “x”	Servo_ID: Servo ID of which the institution will be addressed too	Returns = Servo ID if setStatusPacket is set to ALL
		Direction: RIGHT = clockwise, LEFT = Anti-clockwise	
		Speed: Speed to move at (value range 0x00 to 0x3FF) one unit is about 0.114rpm e.g 0x3FF x 0.114 is about 117.07rpm	Error code (See table below)
Dynamixel.wheelPreload(Servo_ID, Direction, Speed)	Move in wheel mode left/right at speed “x”	Servo_ID: Servo ID of which the institution will be addressed too	Returns = Servo ID if setStatusPacket is set to ALL
		Direction: RIGHT = clockwise, LEFT = Anti-clockwise	
		Speed: Speed to move at (value range 0x00 to 0x3FF) one unit is about 0.114rpm e.g 0x3FF x 0.114 is about 117.07rpm	Error code (See table below)
Dynamixel.action(Servo_ID)	Tell Dynamixel to action any instruction stored in its register	Servo_ID: Servo ID of which the institution will be addressed too	Returns = Servo ID if setStatusPacket is set to ALL Error code (See table below)
Dynamixel.readTemperature(Servo_ID)	Get Dynamixel Temperature in C	Servo_ID: Servo ID of which the institution will be addressed too	Error code (See table below) & Temperature in Celsius
Dynamixel.readVoltage(Servo_ID)	Get Dynamixel Voltage	Servo_ID: Servo ID of which the institution will be addressed too	The HEX value returned is ten time the real value. e.g return value of 109 is 10.9 Volts Error code (See table below) & Voltage
Dynamixel.readPosition(Servo_ID)	Get Dynamixel Position (angle)	Servo_ID: Servo ID of which the institution will be addressed too	one unit is about 0.088 degrees e.g 0xFF x 0.088 is about 360 degrees Error code (See table below) & Posistion
Dynamixel.readLoad(Servo_ID)	Get Dynamixel load and direction of load force on the Dynamixel	Servo_ID: Servo ID of which the institution will be addressed too	Error code (See table below) & Load
Dynamixel.readSpeed(Servo_ID)	Get speed at which the servo is turning	Servo_ID: Servo ID of which the institution will be addressed too	one unit is about 0.114rpm e.g 0x3FF x 0.114 is about 117.07rpm Error code (See table below) & Speed



TO MX28

Pin 1 : GND
Pin 2 : VDD
Pin 3 : Data



Power supply
connected to Arduino
MUST be 12Vdc

Hardware need:

Arduino UNO
74LS241 (Buffer Chip)
10K ohms resistor (pull-up)

