<table>
<tr><th>Arduino institution</th><th>Function</th><th>Parameters</th><th>Returns</th></tr>
<tr><td>Dynamixel.begin(Baud_rate , Control_Pin)</td><td>Setup Arduino to communicate with Dynamixel servo, this instruction must be run at lest once</td><td>Baud_rate: this is the speed at which the Arduino will be set and must be the same as the Dynamixel
Control_Pin: the output pin on the Arduino used to control the full to half duplex buffer chip.</td><td>Error code (See table below)
0 = non</td></tr>
<tr><td>Dynamixel.end</td><td></td><td></td><td></td></tr>
<tr><td>Dynamixel.reset(Servo_ID)</td><td>Reset the Dynamixel servo back to factory default</td><td>Servo_ID: Servo ID of which the institution will be addressed too</td><td>Error code (See table below)
0 = non</td></tr>
<tr><td>Dynamixel.ping(Servo_ID)</td><td>Ping the Dynamixel servo and receive status packet from Dynamixel servo</td><td>Servo_ID: Servo ID of which the institution will be addressed too</td><td>Returns = Servo ID of Pinged servo

Error code (See table below)</td></tr>
<tr><td>Dynamixel.setStatusPaketReturnDelay(Servo_ID, Return_Delay)</td><td>The delay between Dynamixel servo receiving an instruction packet from the Arduino and the Dynamixel servo returning(sending) status packet.</td><td>Servo_ID: Servo ID of which the institution will be addressed too
Return_Delay: Return delay value (in uSec)
(Dynamxiel Default Value = 500 uSec)</td><td>Returns = Servo ID if setStatusPacket is set to ALL

Error code (See table below)</td></tr>
<tr><td>Dynamixel.setBaudRate(Servo_ID, Baud_Rate)</td><td>Set a new baud rate speed on the Dynamixel servo</td><td>Servo_ID: Servo ID of which the institution will be addressed too
Baud_Rate: 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></td><td>Returns = Servo ID if setStatusPacket is set to ALL

Error code (See table below)</td></tr>
<tr><td>Dynamixel.setMaxTorque(Servo_ID, Torque)</td><td>Set max torque value which when reached the Dynamixel will alarm and shut down</td><td>Servo_ID: Servo ID of which the institution will be addressed too
Torque: value of max torque (value range 0x00 to 0x3FF)
(Dynamxiel Default Value = 0x3FF)</td><td>Returns = Servo ID if setStatusPacket is set to ALL

Error code (See table below)</td></tr>
<tr><td>Dynamixel.setHoldingTorque(Servo_ID, Bool)</td><td>Set holding torque</td><td>Servo_ID: Servo ID of which the institution will be addressed too
Bool:
ON = Holding torque true,</td><td>Returns = Servo ID if setStatusPacket is set to ALL

Error code (See table below)</td></tr>
</table>

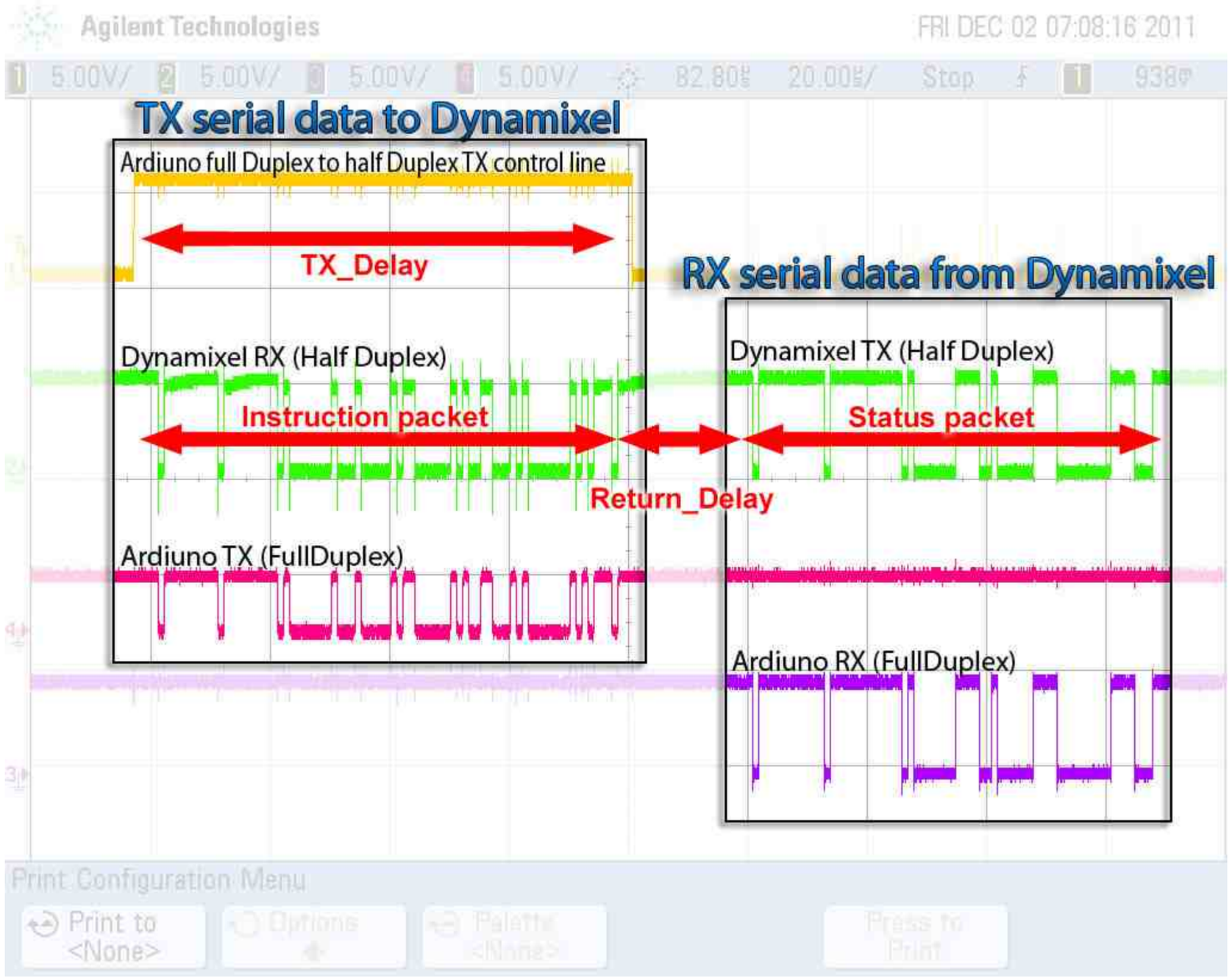
		OFF = Holding torque false	
Dynamixel.setAlarmShutdown(**Servo_ID, Alarms**)	Set which alarms for the Dynamixel to monitor	***Servo_ID:*** Servo ID of which the institution will be addressed too	Returns = Servo ID if setStatusPacket is set to ALL Error code (See table below)
		Alarms: which alarm bits will be set (see Robotis manual for bit define)	
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 Error code (See table below)
		Status: NONE = no status packet are returned from Dynamixel. READ = only read instructions return a status packet. ALL = all instructions return a status packet	
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 Error code (See table below)
		Bool: WHEEL = wheel mode. SERVO = servo mode.	
		CW_Limit: angel limit for clockwise movement (value range 0x01 to 0xFFF) one unit is about 0.088 degrees e.g 0xFFF x 0.088 is about 360 degrees	
		CCW_Limit: angel limit for anticlockwise movement (value range 0x01 to 0xFFF) one unit is about 0.088 degrees e.g 0xFFF 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 tooReturns = Servo ID if setStatusPacket is set to ALL Error code (See table below)	Returns = Servo ID if setStatusPacket is set to ALL 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 Error code (See table below)
		P_Gain: Proportional ban	
		I_Gain: Integral action	
		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 Error code (See table below)
		Temp: Temperature value in Celsius	
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 Error code (See table below)
		Lowest_Volt: Low voltage limit	
		Highest_Volt: High voltage limit	
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 Error code (See table below)
		Angle: Angle to move servo too (value range 0x00 to 0xFFF) one unit is about 0.088 degrees e.g 0xFFF x 0.088 is about 360 degrees	
		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" *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.*	***Servo_ID:*** Servo ID of which the institution will be addressed too ***Angle:*** Angle to move servo too (value range 0x00 to 0xFFF) one unit is about 0.088 degrees e.g 0xFFF x 0.088 is about 360 degrees ***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	Returns = Servo ID if setStatusPacket is set to ALL Error code (See table below)
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 ***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	Returns = Servo ID if setStatusPacket is set to ALL 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 ***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	Returns = Servo ID if setStatusPacket is set to ALL 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 0xFFF 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

Dynamixel.checkRegister(**Servo_ID)**	Check if there are any commands in register e.g Dynamixel.servoPreLoad()	***Servo_ID:*** Servo ID of which the institution will be addressed too	Error code (See table below) & 0 = Non 1 = Yes
Dynamixel.checkMovement(**Servo_ID)**	Check to see if servo is still moving to goal angle	***Servo_ID:*** Servo ID of which the institution will be addressed too	Error code (See table below) & 0 = Goal angle complete 1 = Goal angle still in progress
Dynamixel.checkLock(**Servo_ID)**	Check to see if the Dynamixel has EEPROM locked	***Servo_ID:*** Servo ID of which the institution will be addressed too	Error code (See table below) & 0 = EEPROM can be modified 1 = EEPROM can not be modified
Dynamixel.ledState(**Servo_ID, State**)	Turn LED on Dynamixel off/on	***Servo_ID:*** Servo ID of which the institution will be addressed too ***State:*** ON = turn LED on OFF = Turn LED off	Error code (See table below) 0 = non

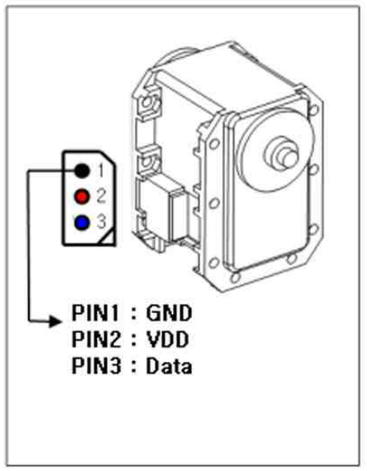
ERROR CODES															
Bits															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Bit(s) are 1 when there is and error, Bit(s) are 0 when there is no error															

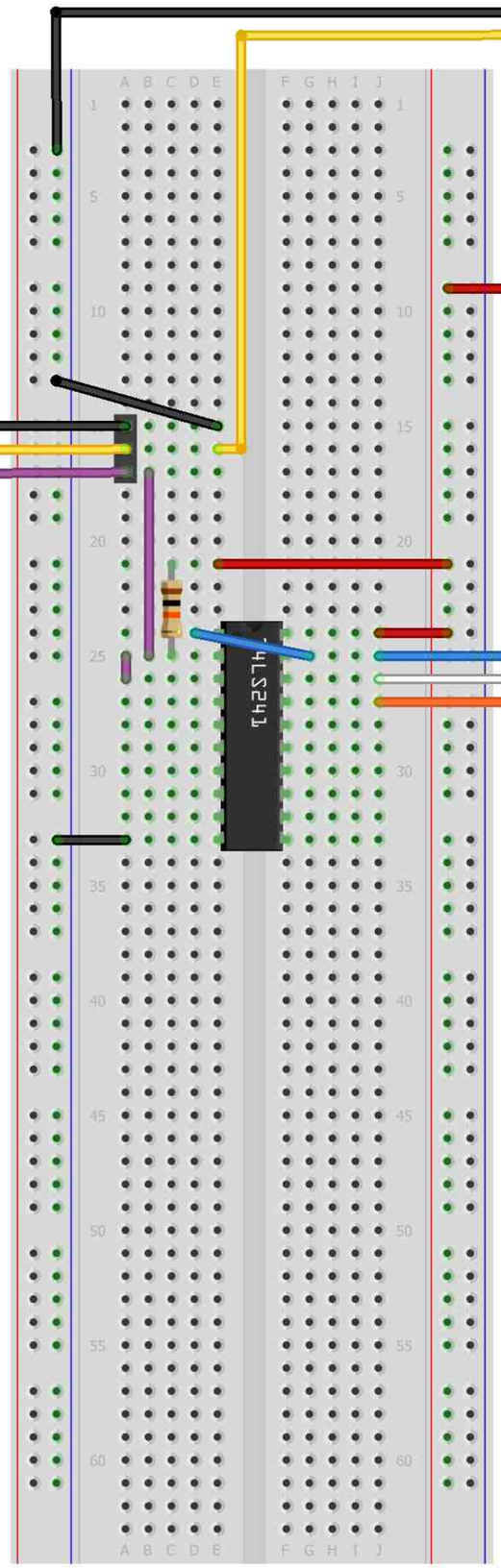
Bit 0	Dynamixel input voltage error (Voltage out side of set range)
Bit 1	Dynamixel angle limit error (Goal position is outside angle limits)
Bit 2	Dynamixel over heating error (Temperature is above set limit)
Bit 3	Dynamixel command is beyond range of use
Bit 4	Dynamixel Checksum error (packet received has invalid checksum)
Bit 5	Dynamixel Overlaod error (current load can not be controlled with the present maximum torque)
Bit 6	Dynamixel Instruction error (A undefined instruction has been send or action command is invalid)
Bit 7	Recived status packet error (there has not been any status packet received by Arduino)
Bit 8	None
Bit 9	None
Bit 10	None
Bit 11	None
Bit 12	All 4 are set 1 when there is a error on any of the above, 0 if there is no error As the Dynamixel does not use the bit 12 to 15 for any data, these four bits are set (0xF000) so if these bits are set there is an error and other bits can be check as per this table to find out error. If bit 12 to 15 are zero all other data returned is valid e.g. temp, volt e.t.c.
Bit 13	
Bit 14	
Bit 15	

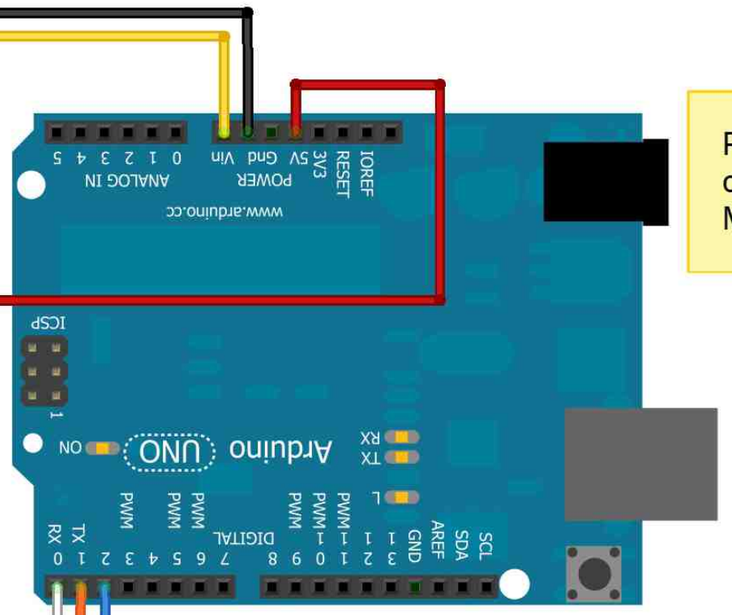

Agilent Technologies
FRI DEC 02 07:08:16 2011
TX serial data to Dynamixel
Ardiuno full Duplex to half Duplex TX control line
TX_Delay
Dynamixel RX (Half Duplex)
Instruction packet
Return_Delay
Ardiuno TX (FullDuplex)
RX serial data from Dynamixel
Dynamixel TX (Half Duplex)
Status packet
Ardiuno RX (FullDuplex)
Print Configuration Menu
Print to <None>
Options
Palette <None>
Press to Print

TO MX28

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



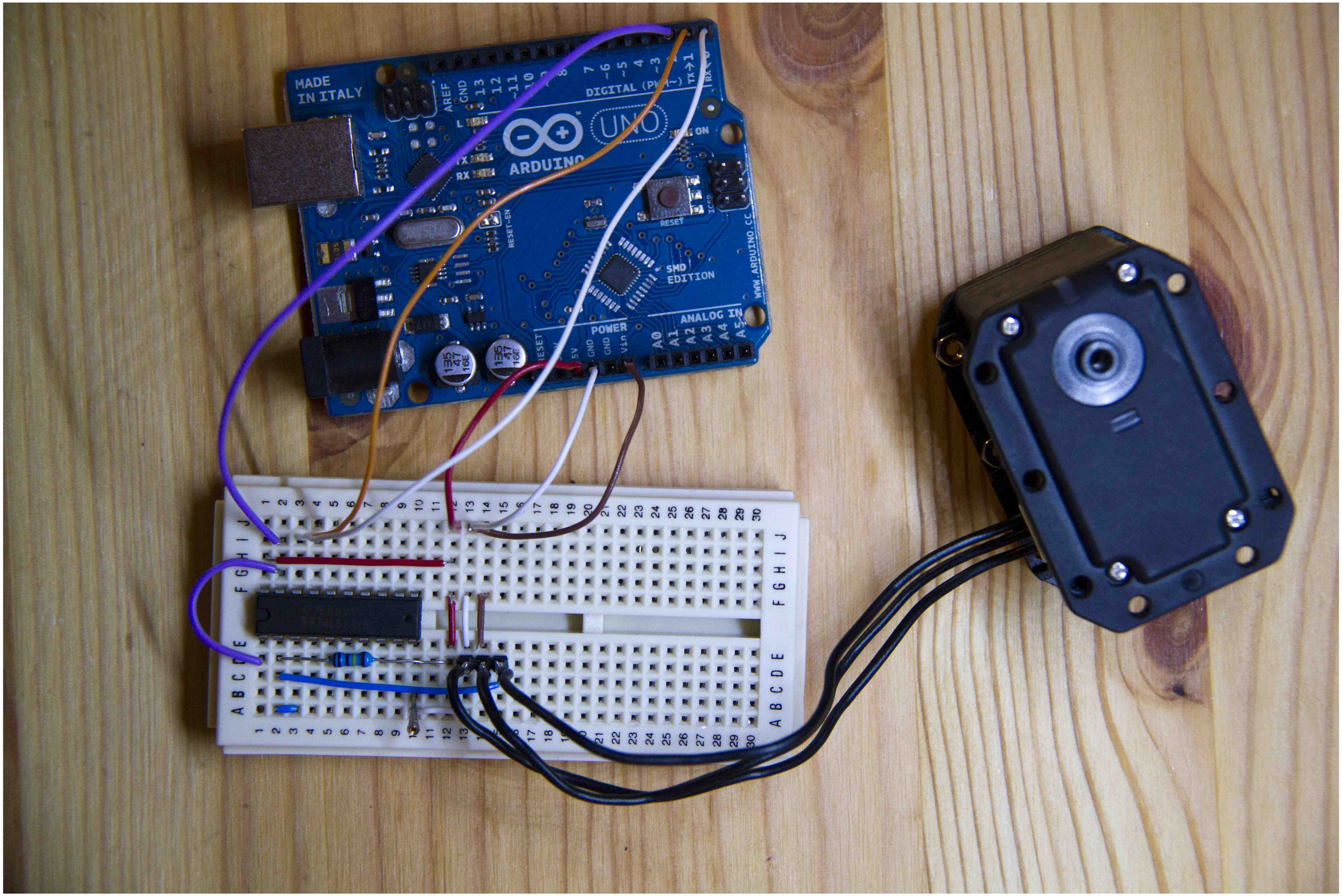




Power supply connected to Arduino MUST be 12Vdc

Hardware need:

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


MADE
IN ITALY
ARDUINO
UNO
DIGITAL (PWM~)
POWER
ANALOG IN
SMD
EDITION
WWW.ARDUINO.CC
RESET
RESET-EN