Quick Start for Dynamixel Pro



ROBOTIS

CONTENTS

CC	ONTENT	TS	2				
1	Tutori	ial	5				
	1.1	Roboplus					
	1.1.1	Preparation	5				
		i. RoboPlus installation (Windows only)	5				
		ii. Wiring of Dynamixel PRO 54 series	5				
		iii. Wiring of Dynamixel PRO 42 series	6				
		iv. Wiring of Dynamixel PRO 54 & 42 series	6				
		v. USB-to-Dynamixel dongle (RS485 setting)	7				
		vi. COM Port Latency Time setting	7				
1.1.	1.1.2	Dynamixel Wizard	9				
		i. Operating a Dynamixel PRO	9				
		ii. Operating Dynamixel PRO in Wheel Mode					
		iii. LED Control of Dynamixel PRO					
		iv. ID configuration					
		v. Modifying the baud rate.					
		vi. Accelerating Dynamixel PRO in Joint Mode	21				
		vii. Limiting the range of motion of Dynamixel PRO	23				
		viii. Extending the range of motion of Dynamixel PRO	25				
		ix. Accelerating Dynamixel PRO in Wheel Mode	27				
		x. Refer to '1.1.2. ii 'Operating Dynamixel PRO in Wheel Mode'					
		xi. Torque Mode of Dynamixel PRO					
		xii. Update Dynamixel PRO's firmware					
		xiii. Dynamixel PRO Firmware Recovery with Dynamixel Wizard					
	1.2	Visual Studio 2010					
	1.2.1	Preparation					
		i. Setting the development envrionment.					
		ii. Inialize and terminate the connnection with USB-to-Dynamixel dongle	41				
	1.2.2	Basic functions of Dynamixel PRO	43				
		i. Turning the torque On/Off of Dynamixel PRO.					
		ii. Operating Dynamixel PRO using C programming language					

	iii.	Modifying Dynamixel PRO's ID using C programming language	47
	iv.	Modifying the baud rate of Dynamixel PRO.	49
	v.	LED control of Dynamixel PRO	53
	vi.	Modifying the P gain value of Dynamixel PRO	56
	vii.	Operating Dynamixel PRO in various speeds	60
	viii.	Internal temperature feedback of Dynamixel PRO	64
	X.	Changing velocity of Dynamixel PRO in Wheel Mode	69
	xi.	Checking the current position and current velocity of Dynamixel PRO	72
1.2.3	C pr	ogramming language funtions	76
	i.	Modifying the zero value of Dynamixel PRO	76
	ii.	Limiting the operating range of Dynamixel PRO	79
	iii.	Extending the operating range of Dynamixel PRO.	83
1.2.4	Indi	rect Addressing function of Dynamixel PRO	87
	i.	Change the position, velocity, and acceleration using Indirect Address fucntion	87
	ii.	Reading the temperature and the current position using Indirect Address	93
1.2.5	Usir	ng Multiple Dynamixel PROs	97
	i.	Controlling the LEDs of 3 Dynamixel PROs	97
	ii.	Controlling the Goal Position of 3 Dynamixel PRO	
	iii.	Reading the the Current Position of 3 Dynamixel PROs	103
	iv.	Read temperature of the first, position of the second, and present current of the third Dyna	mixel PRO 106
1.3	Linu	IX	
1.3.1	Set-	up	
i.	Che	cking USB-to-Dynamixel connection in Linux	
2 Refere	nce		
21	Def	ault values by model	129
2.1.1	U.C.		120
2.1.1	н 56		
	1. 	H54-200-S500-R	
	11. 	H54-100-S500-R	
	111.	H42-20-S300-R	
2.1.2	M S	eries	
	i.	M54-60-S250-R	

	ii. M54-40-S250-R	130
2.1.3	L Series	
	i. L54-50-S290-R	130
	ii. L54-30-S400-R	130
	iii. L42-10-S300-R	130
2.2	Control Table of Dynamixel Pro	
2.3	Features (by width size)	
2.3.1	54-series (H54, M54, L54)	
2.3.2	42-series (H42, L42)	
2.4	Dimensions	
2.4.1	H Series	
	i. H54-200-S500-R	134
	ii. H54-100-S500-R	134
	iii. H42-20-S300-R	135
2.4.2	M Series	
	i. M54-60-S250-R	135
	ii. M54-40-S250-R	135
2.4.3	L Series	
	i. L54-50-S290-R	137
	ii. L54-30-S400-R	137
	iii. L42-10-S300-R	138
2.5	Model notation	

1 Tutorial

This guide is written for first-time Dynamixel PRO users, but, familiar with C and C++ programming languages.

- 1.1 Roboplus
 - 1.1.1 Preparation
 - i. RoboPlus installation (Windows only)
 - RoboPlus is a software package that allows you to easily control and program all ROBOTIS products.
 - This guide contains information regarding the use of RoboPlus's Dynamixel Wizard to test Dynamixel PRO.
 - You may download the most current version of RoboPlus at <u>http://www.robotis.com/xe/download</u>.
 - ii. Wiring of Dynamixel PRO 54 series
 - To operate a Dynamixel PRO, at least, a USB-to-Dynamixel dongle and 24V power supply (for high-power operations) or ROBOTIS' conventional 12V SMPS (for low-power operations) are required.
 - Dynamixel PRO 54 series can be powered via external power cable or 4-pin cable; however, it is recommended to use an external power supply (via power cable) for better stability.
 - As illustrated below, use a 4-pin cable to connect the USB-to-Dynamixel dongle and Dynamixel PRO.
 - Connect one end of the power cable to Dynamixel PRO; the other end to the power supply.



- Please note the schematic below.



- Dynamixel PROs can be connected in series (daisy chain) with the 4-pin cables for communications (and low-power operations). However, power (high-power operations) must be supplied in parallel (individually).
- iii. Wiring of Dynamixel PRO 42 series
 - Similar to the 54 series, the 42 series requires a 24V power supply.
 - Dynamixel PRO 42 series can be powered by the 4-pin cable (low-power operations). Connecting via SMPS-to-Dynamixel is also allowed.
 - A USB-to-Dynamixel dongle connects to SMPS-to-Dynamixel; then SMPS-to-Dynamixel to Dynamixel PRO(s).



- iv. Wiring of Dynamixel PRO 54 & 42 series
 - First, connect an USB-to-Dynamixel dongle and Dynamixel PRO via 4-pin cable.
 - Second, connect a dedicated power supply to Dynamixel PRO via power cable.

For multiple Dynamixel PROs they can be connected in series, for communications (daisy chain) and low-power operations, and parallel (individually), for high-power operations, as illustrated below.

_

- Please note the schematic below.
- v. USB-to-Dynamixel dongle (RS485 setting)
 - Dynamixel PRO communicates via RS485.
 - Set the switch on the left hand side of the USB-to-Dynamixel dongle to RS485 (#2).



- vi. COM Port Latency Time setting
 - To control Dynamixel PRO using a USB-to-Dynamixel dongle, it is recommended to modify the Latency Time of the Port. Please refer to the images below to adjust the Latency Time.
 - Under Windows Device Manager → Port → USB Serial Port (right mouse click)→ properties → Port Setting -> Advanced → Latency Timer (msec) → set to 1msec.

Device Manager	X
9 📴 🙀 🕫	
ces Jllers ng devices	
Update Driver Software Disable Uninstall Scan for hardware changes Properties	
	Device Manager

	USD SI	ental Po	sit (CO	wiza) Proj	Jertiles
General	Port Settings	Driver	Details	Events	
		<u>B</u> its p	er second	9600	~
			<u>D</u> ata bits	8	*
			<u>P</u> arity	None	¥
			<u>S</u> top bits	1	*
		Ele	w control	None	v
			A	lvanced	Restore Defaults
				C)K Cance

Advanced Settin	ngs for COM24	? ×
COM Port Number: COM24 USB Transfer Sizes Select lower settings to correct performance problems at low Select higher settings for faster performance. Receive (Bytes): 4096 V Transmit (Bytes): 4096	v baud rates.	OK Cancel Defaults
BM Options Select lower settings to correct response problems. Latency Timer (msec): 1 v	Miscellaneous Options Serial Enumerator Serial Printer	2
Timeouts Minimum Read Timeout (msec): 0 V Minimum Write Timeout (msec): 0 V	Cancel If Power Off Event On Surprise Removal Set RTS On Close Disable Modem Ctrl At Startup	

- Click "Ok" to confirm.
- 1.1.2 Dynamixel Wizard
 - i. Operating a Dynamixel PRO
 - Supply 24V power to Dynamixel PRO once the wiring is complete. For safety, finish connecting the wires before supplying the power.
 - Open RoboPlus to run Dynamixel Wizard.
 - Select a COM Port that refers to the correction of USB-to-Dynamixel dongle to your PC and click subtron to connect to USB-to-Dynamixel.



- Once the port is connected, select 'DXL 2.0' and '57600,' then click 'Start Searching.' Dynamixel PRO's Default ID setting is 1 and baud rate of 57600 bps.

_

8	Dyn	amixel Wizard	i		×
СОМ24 🔹 🖉 🌋	m 🕈 G 🕀	• 🔞 🗸			
·····? Not found.					
		bos	Search	O DXL 1.0	
	▶ 0	2400			
	1	57600			
	2	115200			
	3	1000000			
	4	2000000		Start searching	
	5	3000000			
				Stop searching	
Ready					

Select the Dynamixel PRO that has been found on the left hand side.

**	Dyr	namixel Wizard	ł	- 🗆 🚺 🗙
COM24 👻 🖉 🌌 🖉	n 🗲 🖟 🖲) 🕜 🔹		
Baud_57600				
		bps	Search	🔿 DXL 1.0 💿 DXL 2.0
	0	2400		
	▶ 1	57600	✓	
	2	115200		
	3	1000000		
	4	2000000		Start exampling
	5	3000000		otal searching
			4// 3	Stop searching
	-			
	Search Cor	mplete		
otal of 1 Dynamixel(s) found.				

🍺 Baud_57600	Addr	Description	Value	
DXL Pro	7	ID	1	
C [D.001] Her	8	Baud Rate	1	
	9	Return Delay Time	250	
	11	Operating mode	3	
	13	Homing offset	0	
	17	Moving threshold	50	
	21	Temperature limit	80	1
	22	Max Voltage Limit	400	
	24	Min Voltage Limit	150	
	26	Accelation Limit	255	
	30	Torque limit	465	
	32	Velocity Limit	10300	
	36	+ Position Limit	900000	
	40	- Position Limit	-900000	
	44	External Port Mode 1	0	Y

- Unlike other Dynamixel models, Dynamixel PRO is only drivable when torque is enabled. Therefore, torque must be enabled before driving Dynamixel PRO.
- Scroll down the middle table on Dynamixel PRO Wizard and locate Torque Enable (#562).

- 65	Dynamixel Wizard		- 🗆 🗙
ЕСОМ24 🗸 🖌 🎢	n 🗲 G 🕕 😥 -		1
	Addr Description	Value 🔨	
[ID:001] H42	36 + Position Limit	900000	
	40 - Position Limit	-900000	
	44 External Port Mode 1	0	
	45 External Port Mode 2	0	
	46 External Port Mode 3	0	
	47 External Port Mode 4	0	
l lr	18 Shutdewn	•	
	562 Torque Enable	0	
	563 LED RED	0	Value
	564 LED GREEN	0	
	565 LED BLUE	0	MAX: I MIN: 0
	586 Velocity I Gain	40	
	588 Velocity P Gain	440	U 🖶
	590 Position D Gain	0	Apply
	592 Poeition Gain	n Y	
	ERROR 6 5 4 3	2 1 0	
Total of 1 Dynamixel(s) found.			

8		Dynamixel Wizard			×
COM24 🗸 🖌	on 3	▶ ଓ ⊕ 🔞 -			
	Addr	Description	Value	^	
[ID:001] H42	36	+ Position Limit	900000		
	40	- Position Limit	-900000		
	44	External Port Mode 1	0		
	45	External Port Mode 2	0		
	46	External Port Mode 3	0		
	47	External Port Mode 4	0		
	48	Shutdown	0		
	562				
	563	LED RED	0		161-2
	564	LED GREEN	0	Value	value
	565	LED BLUE	0		MAX: I MIN: 0
	586	Velocity I Gain	40		
	588	Velocity P Gain	440		
	590	Position D Gain	0		Apply
	592	Position Gain	0	~	
	SIAIL	ERROR 6 5 4 3	210		
Total of 1 Dynamixel(s) found.					

- Change the value on the lower right hand corner to '1' and click 'Apply.'

- Confirm that the 'Torque Enable' value is 1 and click Goal Position (#596).

8	Dynamixel Wizard – 🗆 🗙							
COM24 - 2 🖉 🖉 😰 🖗 🕼 -								
Baud_57600	Addr Description	Value 🔨						
[ID:001] H42	47 External Port Mode 4	0						
	48 Shutdown	0						
	562 Torque Enable	1						
	563 LED RED	0						
	564 LED GREEN	0						
	565 LED BLUE	0						
	586 Velocity I Gain	40						
	588 Velocity P Gain	440						
	590 Position D Gain	0	Mahar					
	592 Position I Gain	0	Value					
	594 Position P Gain	32	MAX: 214/48364/ MIN: -2147483648					
	596 Goal position	-150088	150000					
	600 Goal velocity	0	-130061-					
	604 Goal Torque	0	Apply					
L L	606 Goal accelation	0 4						
	ERROR 6 5 4 3	2 1 0						
Total of 1 Dynamixel(s) found.								

- Change the value of Goal Position on the lower right hand corner with an appropriate value and click 'Apply'. Check each Dynamixel PRO's Min and Max Position Limits (lower and upper "soft" limits).
 by default Dynamixel PRO 54 series: -251000 ~ 251000 and Dynamixel PRO 42 series: -151875 ~ 151875.
- Visually verify position of Dynamixel PRO after clicking on 'Apply.'

- If Dynamixel PRO does not move, check to see if the Torque Enable value is 1. If the value is not set to '1' change the value and try again.
- Change the Goal Position value and click 'Apply.' Visually check Dynamixel PRO's position.
- The relationship between the Goal Position value and degree of rotation is shown below.





- ii. Operating Dynamixel PRO in Wheel Mode
 - There are 3 Modes in with Dynamixel PRO (Joint Mode, Wheel Mode, and Torque Mode).
 - On Dynamixel Wizard, locate and click 'Operating Mode' (#11).
 - When Torque Enable (#562) is turned 'On', the Operating Mode cannot be modified (left image below). The Operating Modes can be modified once the Torque Enable is turned 'Off' by setting it to 0. (right image below)

-

Dynamixel Wizard			Dynamixel Wizard		
E COM17 ■ Waud.57600 ■ ● Walk Pro ■ ● E CON10 H42	Image: system Image: system Image: system	22 • 42 • 0 • 16 • 1 • 250 • 3 • 0 • 50 • 80 • 400 • 150 • 255 • 150 • 1000 •	E COM17 Baud_57600 Baud_57600 BAUD Pro DXL Pro CID:0013 H42	m m m T T T 0 Model Number 2 Model Information 6 Version of Firmware 7 ID 8 Baud Rate 9 Return Delay Time 11 Operating mode 13 Homing offset 17 Moving threshold 21 Temperature limit 22 Max Voltage Limit 24 Min Voltage Limit 25 Accelation Limit 30 Torque limit 30 Lorque Limit	22 42 0 16 1 1 250 3 0 50 50 3 00 150 255 3 00 3 00 50 255 3 0 3 0 3 0 3
총 1개의 다이나믹셀을 찾았습니다	ERROR 6 5 4 3	2 1 0	총 1개의 다이나믹셀을 찾았습니		3 2 1 0

Operating Mode (#11) settings are described below.

Operating Mode	Value	Description				
		Controls the velocity and position of the servo.				
Joint Mode	3	1. Can move to the desired position at a desired velocity.				
		2. Can move using self-assigned trapezoidal Velocity Profile.				
Controls the velocity of the servo.						
Wheel Mede	1	1. Can rotate the servo at a desired velocity.				
wheel Mode	1	2. Can move using self-assigned trapezoidal Velocity Profile.				
		3. Cannot control the position of the servo.				
Controls the torque of the servo.						
		1. Cannot control velocity and position.				
Torque Mode	0	2. Only controls the output torque.				
		3. Since position and velocity control are not possible, the				
		performance is similar to Wheel Mode.				

- Note: use the reference above to set Dynamixel PRO to an appropriate mode.
- Set Toque Enable value as 0 and Operating Mode as 1, then click 'Apply.'

E->> Baud_5/600	주소	: 설명	값	*	
EID:001] H42	0	Model Number	42		
	2	Model Information	0		
	6	Version of Firmware	16	-	
	7	ID	1		
	8	Baud Rate	1		
	9	Return Delay Time	250		
	(1))	Operating mode	1 1		
	13	Homing offset	0		
	17	Moving threshold	50		
	21	Temperature limit	80		武
	22	Max Voltage Limit	400		MAX:
	24	Min Voltage Limit	150		IVIIIN.
	26	Accelation Limit	255		1
	30	30 Torque limit 465		적용	
	32	Velocity Limit	10300	<u>.</u>	<u></u>

- Next, set Torque Enable (#562) as 1 and click 'Apply.' **X Torque Enable value on Dynamixel PRO series MUST be 1 to be activated.**
 - On the Table, set Goal Velocity (#600) as 1000 and click 'Apply.'

📸 Dynamixel Wizard			
: COM17 🔹 🖉	m 🕈 🕃 🛞 🔞 -		1
Baud_57600	주소 설명	값 ^	
[ID:001] H42	48 Shutdown	0	
	562 Torque Enable	1	
	563 LED RED	0	
	564 LED GREEN	0	
	565 LED BLUE	0	
	586 Velocity I Gain	40	
	588 Velocity P Gain	440	
	590 Position D Gain	0	
	592 Position I Gain	0 =	71
	594 Position P Gain	32	ы. Малик 0147400047
	596 Goal position	-442	MAX: 2147483647 MIN: -2147483648
	600 Goal velocity	0	1000
	604 Goal Torque	0	
	606 Goal accelation	0	적용
	610 Moving	1 -	
L	ERROR 6 5 4 3	2 1 0	J
초 17비아 티아티네의 차아스니티	L		
응 패릭 대한대학설을 젖었습니다	-[

- Visually check for Dynamixel PRO rotate slowly in counter-clockwise direction.
- If Dynamixel PRO does not rotate, check if Torque Enable is set to 1.
- Set Goal Velocity (#600) to -1000 then click on 'Apply.' Visually check for Dynamixel PRO rotate slowly in clockwise direction.
- Set Goal Velocity as 0 and Dynamixel PRO stops moving.
- Try changing the Goal Velocity value as 3000, 10000, 15000, -3000, -10000, -15000, 0, etc... and observe the changes in rotation.
- The relationship between Goal Velocity (#600) values and rpm is as follows:

Sign of Value	Rotating Direction	RPM (@ 20V minimum)
+	CCW	M agntude of Value
-	CW	Gear Redución Ratio

***** Notice 1

- If Torque Enable(#562) on the Table is set as 1, it means Dynamixel PRO is ready to operate. This is called the **Torque On** state.
- If Torque Enable(#562) on the Table is set as 0, it means Dynamixel PRO is unable to operate. This is called the **Torque Off** state.

***** Notice 2

- The Table in the middle of the Dynamixel Wizard is called the Control Table.
- Control Table includes the list of functions that is used to control a Dynamixel

- Dynamixel PRO movements and settings can be adjusted by inputting the appropriate values on the address.

- For example, on Control Table, the values on Torque Enable (#562) can be changed to 0 or 1 to turn torque. Also, appropriate values can be entered into Goal Velocity (#600) to rotate in desired velocity.

***** Notice 3

- The values that are 'locked' on the Control Table when the Torque Mode is on, is referred as the EEPROM domain (non-volatile).
- The values that are written on the EEPROM area are preserved even if the Dynamixel PRO is turned on/off. The value cannot be modified when the torque is on, and the torque needs to be off to modify the EEPROM area values.
- EEPROM area is indicated as pink on Dynamixel Wizard.
- The section that is indicated as blue on the Control Table, such as Goal Velocity, is called the RAM domain (volatile).
- The values on the RAM domain are lost when Dynamixel PRO is turned off. The values can be changed regardless of the torque on/off setting.

- iii. LED Control of Dynamixel PRO
 - Unlike other Dynamixels, Dynamixel PRO Dynamixel PRO has 3 color LEDs.
 - Therefore, LEDs can be set to emit multiple colors.
 - Try clicking the LED RED.

Baud_1000000	주소	설명	값	•	
March 10 (ID:003] H42	36	+ Position Limit	303750		
	40	- Position Limit	-303750		
	44	External Port Mode 1	0		
	45	External Port Mode 2	0		
	46	External Port Mode 3	0		
	47	External Port Mode 4	0		
	48	Shutdown	0	=	
	562	Torque Enable	0		
	583				
	564	LED GREEN	0		11.
	565	LED BLUE	0		MAX: 25
	586	Velocity I Gain	40		1400 V.
	588	Velocity P Gain	440		U 8
	590	Position D Gain	0		적용
	592	Position L Gain	n	Ŧ	<u> </u>

- Input 255 under LED RED value and click 'Apply'. The red LED should turn on.
- The intensity of the LED emission can be adjusted by changing the values between 0-255.
- Try changing the values of the LED RED, LED GREEN, and LED BLUE to observe the change in colors.



iv. ID configuration

- Individual ID is required for communication when when multiple Dynamixel PROs are linked together.
- For example, a single Dynamixel PRO can be driven via its ID even when 3 Dynamixel PROs are linked together.
- A problem may occur if more than one Dynamixel with the same ID are linked together.
- #7 on the Control Table of the Dynamixel Wizard is the value for ID.
- ID is a part of the EEPROM area on the Control Table. Be sure to check if Dynamixel PRO torque is Off before changing the ID.

d_57600	소?	설명	값	
[ID:001] H42	0	Model Number	42	
	2	Model Information	0	
	6	Version of Firmware	16	1
	7	ID	1	
	8	Baud Rate	1	-
	9	Return Delay Time	250	
	11	Operating mode	3	
	13	Homing offset	0	
1	17	Moving threshold	50	
	21	Temperature limit	80	
	22	Max Voltage Limit	400	
	24	Min Voltage Limit	150	
	26	Accelation Limit	255	
3	30	Torque limit	465	
	32	Velocity Limit	10300	्

- Once changes are made to ID (#7) value to 3; click on 'Apply.' The change of value can be observed on the left table.

	주소	설명	값	*		
E-@ [ID:003] H42	0	Model Number	42			
- 222	2	Model Information	0			
	6	Version of Firmware	16	1		
	7					
	8	Baud Rate	1			
	9 Return Delay Time	Return Delay Time	250			
	11	Operating mode	3			
	13	Homing offset	0			
	17	Moving threshold	50		-	
	21 Temperature limit 80 22 Max Voltage Limit 400		-11. 			
			MAX: MIND	25		
	24	24 Min Voltage Limit 1	150		1911131	
	26	Accelation Limit	255		3	1
	30 Torque limit	Torque limit	465		적용	
	32	Velocity Limit	10300	· •	<u> </u>	_

- If the Dynamixel PRO ID cannot be changed, make sure the Torque Enable is turned off.

- v. Modifying the baud rate.
 - #8 on the Control Table of the Dynamixel Wizard is the baud rate.
 - <u>Baud rate is a part of the EEPROM area on the Control Table. Be sure to check if Dynamixel PRO torque is off before changing the baud rate.</u>
 - Set the baud rate value as 3 then click 'Apply.'

Baud_1000000	주소	: 설명	값	*	
EID:003] H42	0	Model Number	42		
- T 191	2	Model Information	0		
	6	Version of Firmware	16	=	
	7	ID	3		
	8	Baud Rate	3		
	9	Return Delay Time	250		
	11	Operating mode	3		
	13	Homing offset	0		
	17	Moving threshold	50		
	21	Temperature limit	80		- 武
	22	Max Voltage Limit	400		MAX:
	24	Min Voltage Limit	150		IVIIIN.
	26	Accelation Limit	255		3
	30	Torque limit	465		적용
	32	Velocity Limit	10300	Ŧ	<u></u>

- Notice the change in baud rate to 1Mbps.
- If the baud rate cannot be modified check if Dynamixel PRO Torque Enable is turned on.
- Dynamixel PRO's baud rate and Control Table values are as follows:

Value	of	Control	Baud Rate(bps: bit for seconds)
Table			
0			2400 bps
1			57600 bps
2			115200 bps
3			1 Mbps
4			2Mbps
5			3Mbps
6			4Mbps
7			4.5Mbps
8			10.5Mbps

- ****** When using a USB-to-Dynamixwl dongle, Dynamixel PRO's baud rate needs to be less than the USB-to-Dynamixwl's supported baud rate in current use.

- vi. Accelerating Dynamixel PRO in Joint Mode
 - Verify if Dynamixel PRO is set as torque off status. If not, click on the Torque Enable (#562) on Control Table and set the value to 0. Click the 'Apply' to change it to torque off status.
 - Modify the Operating Mode (#11) value as 3 then click 'Apply' to set Dynamixel PRO as Joint Mode.

COM17 🔹 🖉 🎽	60	7600				
Baud_1000000	주소	설명	값			
EID:003] H42	590	Position D Gain	0			
	592	Position I Gain	0			
	594	Position P Gain	32			
	596	Goal position	0			
	600	Goal velocity	0			
	604	Goal Torque	0			
	606		4			
	610	Moving	0			
	611	Present position	-2	-	71	
	615	Present velocity	0		武	
	621	Present Current	65518	E	MAX: 214748	3647 C
	623	Present input voltage	121		IVIII N.	100
	625	Present temperature	44086		4	-
	627	External Port Data 1	11269		적용	
	629	External Port Data 2	4868	Ŧ	<u> </u>	_
	STAT	ERROR 6 5 4	321	0		

Set Goal Acceleration (#606) value as 4, then click 'Apply.'

- Set Torque Enable (#562) value as 1 and click 'Apply' to change Dynamixel PRO to 'Torque On' status.
- Set the Goal Position (#596) value to 100,000 or -100,000, then click 'Apply.' Note the difference in the movement of Dynamixel PRO from the time the Goal Acceleration was set to 0.

Baud_100000	주소	설명	값	*	
E [ID:003] H42	563	LED RED	0	11	
	564	LED GREEN	0		
	565	LED BLUE	0		
	586	Velocity I Gain	40		
	588	Velocity P Gain	440		
	590	Position D Gain	0		
	592	Position I Gain	0		
	594	Position P Gain	32		
	596	Goal position	100000		
	600	Goal velocity	0	н	값
	604	Goal Torque	0		MAX: 214748364
	606	Goal accelation	4		MIN: -214740304
	610	Moving	0		10000
	611	Present position	100003		적용
	615	Present velocity	n	-	<u></u>
	STAT	US ERROR 6 5 4 3	210)	

- If Dynamixel PRO cannot be driven check if Torque Enable is turned on.
- Also, if the Goal Acceleration movement does not differ from the input value, check if Goal Acceleration value is set to 4.
- If the Goal Acceleration value is not zero Dynamixel PRO will create a 'Trapezoidal Velocity Profile' (2.2 reference).
- Input random variables into Goal Acceleration and Goal Position to observe the different responses of Dynamixel PRO.

- vii. Limiting the range of motion of Dynamixel PRO
 - Set Dynamixel PRO to torque off status.
 - Set +Position Limit (#36) value as 100,000 then click 'Apply.'
 - Set –Position Limit (#40) value as -100,000 then click 'Apply.'

Baud_1000000	주소 설명		
T ♥ [ID:003] H42	26 Accelation Limit 30 Torque limit 32 Velocity Limit 36 • Position Limit 40 - Position Limit 40 - Position Limit 44 External Port Mode 1 45 External Port Mode 2 46 External Port Mode 3 47 External Port Mode 4 48 Shutdown 562 Torque Enable 563 LED RED 564 LED GREEN 565 LED BLUE 585 Velocity L Gain	2k 255 465 10300 -100000 -100000 0 0	▲
L		3 2 1 0	

- <u>On the Control Table, +Position Limit (#36) and –Position Limit (#40) are a</u> part of the EEPROM area.
- <u>Therefore, if the values cannot be changed check if the Torque Enable is</u> <u>turned On.</u>
- If +Position Limitd and –Position Limit values can be modified it means Torque Enable is on.

→☆ →☆ →☆ ↓☆				36	1	1
Image: Set Section 1 Gain 40 588 Velocity P Gain 440 590 Position D Gain 0 592 Position P Gain 32 596 Goal position 0 600 Goal velocity 0 604 Goal accelation 4 610 Moving 0 611 Present position 2 615 Present velocity 0 621 Present Current 65524 623 Present input veltage 121	Buddel Hococo	수소	설명	값		
588 Velocity P Gain 440 590 Position D Gain 0 592 Position I Gain 0 594 Position P Gain 32 595 Goal position 0 600 Goal velocity 0 604 Goal accelation 4 610 Moving 0 611 Present position 2 615 Present velocity 0 621 Present Current 65524 623 Present input veltage 121	T 👸 [ID:003] H42	586	Velocity I Gain	40		
590 Position D Gain 0 592 Position I Gain 0 594 Position P Gain 32 596 Goal position 0 600 Goal velocity 0 604 Goal accelation 4 610 Moving 0 611 Present position 2 615 Present velocity 0 621 Present Current 65524 633 Present input veltage 121		588	Velocity P Gain	440		
592 Position I Gain 0 594 Position P Gain 32 595 Goal position 0 600 Goal velocity 0 604 Goal accelation 4 610 Moving 0 611 Present position 2 615 Present velocity 0 621 Present Current 65524 623 Present input veltage 121		590	Position D Gain	0		
594 Position P Gain 32 996 Goal position 0 600 Goal velocity 0 604 Goal Torque 0 606 Goal accelation 4 610 Moving 0 611 Present position 2 615 Present velocity 0 621 Present Current 65524 623 Present input velocate 121		592	Position I Gain	0		
596 Goal position 0 600 Goal velocity 0 604 Goal Torque 0 606 Goal accelation 4 610 Moving 0 611 Present position 2 615 Present velocity 0 621 Present Current 65524 623 Present input velocite 121		594	Position P Gain	32		
600 Goal velocity 0 604 Goal Torque 0 606 Goal accelation 4 610 Moving 0 611 Present position 2 615 Present velocity 0 621 Present Current 65524 623 Present input veloce 121		596				
604 Goal Torque 0 606 Goal accelation 4 610 Moving 0 611 Present position 2 615 Present velocity 0 621 Present Current 65524 623 Present input voltage 121		600	Goal velocity	0		
606 Goal accelation 4 610 Moving 0 611 Present position 2 615 Present velocity 0 621 Present Current 65524 623 Present input veloce 121		604	Goal Torque	0		
610 Moving 0 611 Present position 2 615 Present velocity 0 621 Present Current 65524 623 Present input voltage 121		606	Goal accelation	4		
611 Present position 2 615 Present velocity 0 621 Present Current 65524 623 Present input veltage 121		610	Moving	0		武
615 Present velocity 0 621 Present Current 65524 623 Present input voltage 121		611	Present position	2	ш	MAX: 214748364
621 Present Current 65524 150000		615	Present velocity	0		IVIIIN: -214740304(
623 Present input voltage 121 31 8		621	Present Current	65524		150000
deb i resent input voltage i ei ge		623	Present input voltage	121		적용
625 Present temperature 62526 -		625	Present temperature	62526		<u> </u>

- Once Torque Enable is on set Goal Position (#596) value as 150,000.

- Even though the Dynamixel PRO is in a torque on state, there is no response because the Position Limit value exceeds the set limit.
- If Dynamixel PRO is operational, check the +Position Limit value, and if the value is not 100,000, turn torque off. Input the +Position Limit value as 100,000 and click 'Apply;' then turn torqueoOn.
- This time, set the Goal Position value as 100,000. Observe movement to position value '100,000.'
- Input the Goal Position value as -150,000 (there should be no movement). Try inputting -100,000 instead (Dynamixel PRO will move).
- Input random Goal Position values and click on 'Apply.' Position values will move within the "soft" limits.

- viii. Extending the range of motion of Dynamixel PRO
 - +Position Limit and –Position Limit restrict or extend Dynamixel PRO's operation range.
 - Turn off Torque Enable by changing the value to 0.
 - On the Control Table, set +Position Limit (#36) value as 900,000 then click 'Apply.'
 - On the Control Table, set –Position Limit (#40) value as -900,000 then click 'Apply.'



- <u>+Position Limit(#36) and –Position Limit(#40) on the Control Table are a</u> <u>part of EEPROM area.</u>
- If the values cannot be changed, check if Torque Enable is on.
- Change the +Position and –Position Limit values, and then turn Torque Enable on.

COM17 🔹 🖉 🎽	00	2600			
∃~>> Baud_1000000	주소	설명	값	*	
[ID:003] H42	590	Position D Gain	0		
- T- 1	592	Position I Gain	0		
	594	Position P Gain	32		
	596	Goal position	900000		
	600	Goal velocity	0		
	604	Goal Torque	0		
	606	Goal accelation	4		
	610	Moving	0		
	611	Present position	899998		
	615	Present velocity	0		- 武
	621	Present Current	15		MAX: 214748364
	623	Present input voltage	121	=	14114, -2147403040
	625	Present temperature	52800		90000
	627	External Port Data 1	23300		적용
	629	External Port Nata 2	8195	Ŧ	<u> </u>
	STAT	US ERROR 654	3210)	

Set the value of Goal Position (#596) as 900,000 while Enable Torque is on.

- Dynamixel PRO rotates more than 360 degrees to reach its goal position.
- If Dynamixel PRO is not responsive check the +Position Limit value. If the value is not 900,000 turn Torque Enable off. Change the +Position Limit value to 900,000 and click on 'Apply.' Turn Torque Enable back on.
- Set Goal Position as -900,000. Dynamixel PRO will rotate a couple of times in a couter-clockwise direction and stop at the designated position.
- Input random values into the Goal Position and click 'Apply.' Verify that Dynamixel PRO reaches the Goal Position.
- Please refer to 1.1.2.i. 'Relationship between angle (degrees) and position value.'

- ix. Accelerating Dynamixel PRO in Wheel Mode
- x. Refer to '1.1.2. ii 'Operating Dynamixel PRO in Wheel Mode'
 - Set Goal Acceleration (#606) value as 4, and then click 'Apply.'

				_	
Badd 100000	주소	설명	값	-	
T. 👸 [ID:003] H42	590	Position D Gain	0		
	592	Position I Gain	0		
	594	Position P Gain	32		
	596	Goal position	0		
	600	Goal velocity	0		
	604	Goal Torque	0		
	606	Goal accelation	4		
	610	Moving	0		
	611	Present position	-2		71
	615	Present velocity	0		截
	621	Present Current	65518		MAX; 214/48364
	623	Present input voltage	121		iviin,
	625	Present temperature	44086		4
	627	External Port Data 1	11269		적용
	629	External Port Data 2	4868		L
	STAT	US	321	D	

- Set Torque Enable (#562) as 1, and click 'Apply' to set Dynamixel PRO to torque on status.
- Set the Goal Velocity (#600) value to 5,000 or -5,000 and click 'Apply.' Compare the movement to the time when the Goal Acceleration value was set to 0.

563 LEC 564 LEC 565 LEC 586 Vel 588 Vel 590 Pos 592 Pos 594 Pos 594 Pos 594 Pos 595 Go	D RED D GREEN D BLUE locity I Gain locity P Gain sition D Gain sition I Gain	0 0 40 440 0 0 32		
564 LEC 565 LEC 586 Vel 588 Vel 590 Pos 592 Pos 594 Pos 595 Go	D GREEN D BLUE locity I Gain locity P Gain sition D Gain sition I Gain	0 0 40 440 0 0 32		
565 LEC 586 Vel 588 Vel 590 Pos 592 Pos 594 Pos 596 Go	D BLUE locity I Gain sition D Gain sition I Gain sition P Gain	0 40 440 0 0 32		
586 Vel 588 Vel 590 Pos 592 Pos 594 Pos 596 Go	ocity I Gain Iocity P Gain sition D Gain sition I Gain sition P Gain	40 440 0 0 32		
588 Vel 590 Pos 592 Pos 594 Pos 596 Go	locity P Gain sition D Gain sition I Gain sition P Gain	440 0 0 32	-	
590 Pos 592 Pos 594 Pos 596 Go 590 Go	sition D Gain sition I Gain sition P Gain	000000000000000000000000000000000000000	-	
592 Pos 594 Pos 595 Go 500 Go	sition I Gain sition P Gain	0		
594 Pos 596 Go 600 Go	sition P Gain	32		
596 Go				
600 Go		100000		
	al velocity	0	E	- 값
604 Go:	al Torque	0		MAX: 214748
606 Go:	al accelation	4		100000
610. Mo	ving	0		
611 Pre	esent position	100003		적용
615 Pre	sent velocitu	n	-	

- If Dynamixel PRO does not move check it the Enable Torque value is 1.
- If the movement is no different than when the Goal Acceleration value was 0 check if the Goal Acceleration value is set to 4.

- If the Goal Acceleration value is not zero Dynamixel PRO will create a 'Trapezoidal Velocity Profile' (2.2 reference).
- Input random values into Goal Acceleration and Goal Position to observe the different responses of Dynamixel PRO.

- xi. Torque Mode of Dynamixel PRO
 - Set Dynamixel PRO to torque off condition and input the Operating Mode value as 0. Click on 'Apply.'

🖃 🌝 Baud_1000000	주:	소 설명	값	*	
E 🗃 DXL Pro	0	Model Number	42		
	2	Model Information	0	-	
	6	Version of Firmware	16	E	
	7	ID	3		
	8	Baud Rate	3		
	9	Return Delay Time	250		
			0		
	13	Homing offset	0		
	17	Moving threshold	50		71
	21	Temperature limit	80		EL .
	22	Max Voltage Limit	400		MINE
	24	Min Voltage Limit	150		
	26	Accelation Limit	255		
	30	Torque limit	465		적용
	32	Velocity Limit	10300	-	<u></u>

- Enable the torque by turning Torque Enable on.
- Set Goal Torque (#604) value as 100, and click 'Apply'.

📸 Dynamixel Wizard	-			-		×
COM17 🔹 🖉	ano	<u>≯ 6 ⊕ @ -</u>			1	
	주소	설명	값	^		
[ID:003] H42	48	Shutdown	Û			
	562	Torque Enable	1			
	563	LED RED	Û			
	564	LED GREEN	0			
	565	LED BLUE	0			
	586	Velocity I Gain	40			
	588	Velocity P Gain	440			
	590	Position D Gain	0			
	592	Position I Gain	0	Ξ	71	
	594	Position P Gain	32		ы́.	00707
	596	Goal position	-11334		MAX: MINE	32768 -32768
	600	Goal velocity	0	1	100	-32100
	604				100	÷
	606	Goal accelation	0		적용	
	610	Moving	1	Ŧ		
L L	SIAI	ERROR 6 5 4 C	21	0	1	
초 17박이 다이나 비세운 차안스니 [-L					
중 표계의 다이나라설을 젖었습니다	-1			_		

- Note the rotation of the Dynamixel PRO accelerating in the counter-clockwise direction.
- If Dynamixel PRO is not moving check if Dynamixel PRO is in torque off condition.

- Set the Goal Torque (#604) as -100 and click on 'Apply.'
- Note the rotation of Dynamixel PRO accelerating in the clockwise direction.
- Set the Goal Velocity as 0 to stop rotation.
- Try inputing 30, 100, 400, -400, -100, -30, 0, etc... values into Goal Velocity and click on 'Apply.' Observe how the Dynamixel PRO responds.
- Goal Torque (#604) value controls the flow of current into Dynamixel PRO.
- The relationship between the Goal Torque value and the current is shown below. Please refer to Dynamixel PRO datasheet to see how current relates to torque.

Model	Relationship between goal torque and current
H54	Current $(mA) = goal$ torque value $\times \frac{33000}{2048}$
H42	Current $(mA) = goal$ torque value $\times \frac{8250}{2048}$

- xii. Update Dynamixel PRO's firmware
 - Firmware refers to code installed into Dynamixel PRO. This code is responsibe for controlling Dynamixel PRO.
 - Dynamixel Wizard connects to the internet and checks for the latest firmware
 - When a new firmware for Dynamixel PRO is detected a red checkmark appears on the Dynamixel PRO icon (see illustrations below).



- Only one Dynamixel PRO must be connected at a time during firmware update.
- For firmware update select the correct Dynamixel PRO device and click the Firmware update button.
- Begin firmware update by following the instructions.
- Be sure to maintain all connections and power supply until update is finished.

	Dynamixel Fir	mware Update	
Start Dynamixel ma	nagement.		
Precautions :			
- Do not unplug US	B2Dynamixel from	PC.	
- Do not cut off pov	ver of Dynamixel.		
- Do not unplug Dy	namixel from USB2	Dynamixel.	
	c Rack	Next >	Capaal
	< DBCK	INEXT	Cancer

- The model number and firmware of the connected Dynamixel PRO can be verified.

H42] Baud:1000000, ID:1, Version: 16 -> 17 === Total Count : 1		
=== Total Count : 1		[°]
	=== Total Count : 1	
		Ų
	< Pack Next >	Cancel

- Click on 'Next' to begin firmware update. Be sure to maintain all connections and power supply until the update is complete.

	Dynamixel Firmware Update
Installing Dynami	xel firmware!
100% (1 / 1)	
	< Back Next > Cancel

- Check Dynamixel PRO firmware update results.

Upa	ate Result *****************	^
[H42] Baud:100000	0, ID:1 : Success	
=== Total Count : 1		
		~

- xiii. Dynamixel PRO Firmware Recovery with Dynamixel Wizard.
 - Dynamixel Wizard can be used to recover the Dynamixel PRO's firmware if there is an issue.
 - ****** When firmware is recovered, all the settings are set to default; therefore, the ID and baud rate must be checked. Be sure to have the USB-to-Dynamixel dongle properly set to the appropriate connector (RS-485).
 - ****** When recoverying the firmware, Dynamixel PRO must be connected one at a time. Dynamixel PRO may malfunction if two or more Dynamixel PROs are daisy chained when recoverying the firmware.
 - Begin Dynamixel PRO firmware recovery by clicking on the firmware recovery icon from the toolbar.
 - Select thet COM port number of the dongle to begin firmware recovery.

<u>10</u>			Dynamixe			mixe		
COM24	-	1	2	0%	1	8	Ð	0.
		1	27		1	U	0	U

Please read the instructions and begin Dynamixel PRO firmware recovery.

Dynamixel Firmware Hecovery
It will start Dynamixel firmware management,
Please, do not action below during update,
- Do not unplug USB2Dynamixel from PC,
- Do not cut off power of Dynamixel,
- Do not unplug Dynamixel from USB2Dynamixel,
< Back Next > Cancel

- Only one Dynamixel PRO must be linked to before initiating the Dynamixel firmware recovery.



- Dynamixel PRO cannot be searched automatically because the firmware is not recognized. Therefore, you must set the Dynamixel PRO connected port manually. Since Dynamixel PRO cannot be recognized if the port is in use, finish other programs, and then continue the procedure.
- Select USB-to-Dynamixel-connected port and press "Search" button. Select the port the USB-to-Dynamixel dongle is connected to and click 'Search.'

Select the p	on connected to Dynamixel(s) and cli	ck search.
f unable to f	ind any Dynamixel(s)	
then the por	t was set improperly or	
he port is cu	mently used by another application.	
Port :	COM24 ↔	Coarob
Status :	Not connected	Search

- Turn off Dynamixel PRO then turn it back on to let the program detect Dynamixel PRO.

Disconnect t	he connected Dynamixel, connect it a	again.
Only one dyn	namixel should be connected before re	ecovery.
Port :	COM24 v	Saamh
Port : Status :	COM24 v	Search
Port : Status :	COM24 v Not connected	Search

- If the Dynamixel PRO is not detected turn Dynamixel PRO off and on again.

_

The screen will look like the following once Dynamixel PRO is detected.

	Dynamix	kel Firmw	are Recover	У
)ynamixel(s)	detected.			
Port :	COM24	~		County

- Once Dynamixel PRO is detected downloadable information will appear. In Dynamixel PRO firmware recovery mode select the correct Dynamixel PRO version. Dynamixel PRO may malfunction if an incorrect firmware is downloaded.

~	Version :	17
using <mark>d</mark> ifferent mo	odel's firmware.	
	our o minimaro.	
	using different m	Version : using different model's firmware.

- Click on 'Next' to start firmware recovery. Be sure to maintain all connections and power supply until the recovery is complete.
| lling Dynamixel fim | ware ! | | |
|------------------------------|---------------------|----------------------|------------------------|
| ot tum the power of | off or unplug the o | cable before the ins | tallation is complete. |
| | | | 47 |
| Model :
6 (76265/76265 by | H4Z
(tes) | Version : | 17 |

- Dynamixel PRO firmware recovery is complete.

	Dyr	namixel Fi	rmware	Recover	У	
Congrate	lations!					
Dynamix	el <mark>firmware re</mark> c	covery succes	ssful.			
	T	e Baole		Nort 5	De	iab
		< pack		Mext >	FIN	ISI

1.2 Visual Studio 2010

1.2.1 Preparation

- i. Setting the development envrionment.
 - For the following tutorial, prepare one Dynamixel PRO with an ID 1 and baud rate of 57600 bps.
 - To familiarize the terminology used in the tutorial please go over the previous chapters.
 - Find the Dynamixel SDK 2.0 for Windows with the included USB memory dongle.
 - Extract the compressed file.



- Dynamixel SDK for Windows contains the following:

0	C
/bin	: DLL for Windows.
/import	: header and library files are located. (lib, h)
/src	: Contains DLL source code.
/example	: examples of various examples commonly used to
Dynamixel PRO.	

- Properly set settings to use Dynamixel SDK in Visual Studio 2010.
- Create 'New Project' on Visual Studio 2010.
- In the 'Project Folder' you made, copy the following files are found inside the extracted SDK file: bin/dynamixel.dll, /import/dynamixel.h, /import/dynamixel.lib

control



- Go to project properties (see illustration below).

- On 'Properties', click 'Linker -> Input.
- Click on the 'Additional Dependancies', then select 'Edit.'

DXL_Pro_Example 속성 페이지		8 2
구성(C): 활성(Debug)	▼ 플랫폼(P): 활성(Win32)	▼ 구성 관리자(0)
 > 공용 속성 ▲ 구성 속성 일반 디버깅 > C/C++ 디렉터리 > C/C++ 3커 일반 입력 매니페스트 파일 디버깅 시스템 최적화 포급 명령줄 > 매니페스트 도구 > XML 문서 생성기 > 찾아보기 정보 > 별드 이벤트 > 사용자 지정 빌드 단계 	추가 증속성 모든 기본 라이브러리 무시 특정 기본 라이브러리 무시 모듈 정의 파일 어셈블리에 모듈 추가 관리되는 리소스 파일 포함 경제 기호 참조 지연 로드된 DLL 어셈블리와 리소스 링크	<mark>(전집></mark> <편집> <푸모 또는 프로젝트 기본값에서 상속> I정합니다.
		확인 취소 적용(A)



- Enter 'dynamixel.lib' and click on 'confirm' to save the changes.

- Dynamixel SDK programming setup is complete.

ii. Inialize and terminate the connnection with USB-to-Dynamixel dongle

- First, open a .cpp file and copy the code below. The **COM_PORT_NUM** and **BAUD_RATE_NUM** values need to match the proper USB-to-Dynamixel dongle in use.

#include	"dynamixel.h"			
#define	COM_PORT_NUM	17	//Comport Number of USB2DXL	
#define	BAUD_RATE_NUM	1	//Baudrate Number of Dynamixel PRO	

- The relationship between the baud rate number and the baud rate is indicated below.

Baudrate	Bps(= bit per seconds)
Number	
0	2400 bps
1	57600 bps
2	115200 bps
3	1 Mbps
4	2 Mbps
5	3 Mbps
6	4 Mbps
7	4.5 Mbps
8	10.5 Mbps

- Declare the 'SerialPort' type variable and then initialize all the values to zero.
- Declare the 'SerialPort' pointer type variable, then initialize it with addresss of the predefined variable. Please refer to the source code shown below.

SerialPort sp = $\{0,0,0,0,0\};$	
SerialPort *Port = &sp	

- Next, implement functions, **dxl_initialize** and **dxl_terminate**, to connect and disconnect to USB-to-Dynamixel dongle.
- The program's entire source is shown below.

main.cpp

······································		
<pre>#include <stdio.h></stdio.h></pre>		
<pre>#include <conio.h></conio.h></pre>		
<pre>#include "dynamixel.h"</pre>		
-		
#define COM_PORT_NUM	17	//Comport Number of USB2DXL
#define BAUD_RATE_NUM	1	//Baudrate Number of Dynamixel PRO. 1 is 57600 bps
int main(void)		
{		
SerialPort sp = $\{0,0,0,0,0\}$;		
SerialPort *Port = &sp		
//Open the port of USB2DX	KL	
if(dxl_initialize(Port, COM	_PORT_N	UM, BAUD_RATE_NUM) == COMM_RXSUCCESS)

```
printf("Succeed to open USB2Dynamixel!\n");
else
{
    printf( "Failed to open USB2Dynamixel!\n" );
    printf( "Press any key to terminate...\n" );
    __getch();
    return 0;
}
printf( "Press any key to terminate...\n" );
getch();
//Close the port of USB2DXL
dxl_terminate(Port);
return 0;
```

- **COMM_RXSUCCESS** is returned when USB-to-Dynamixel dongle is successfully communicates with Dynamixel PRO.
- When using the Dynamixel SDK to communicate with USB-to-Dynamixel you can check if communication is successful.

1.2.2 Basic functions of Dynamixel PRO.

- i. Turning the torque On/Off of Dynamixel PRO.
 - As explained in 1.1, Dynamixel PRO's Control Table contains Torque Enable function that controls the torque to be on/off. The address for Torque Enable function is 562. Torque Enable is has 1 byte assigned.
 - Therefore, implement dxl_write_byte function (See 2.4.3)
 - The program's entire source is shown below.

```
main.cpp
#include <stdio.h>
#include <conio.h>
#include "dynamixel.h"
#define COM PORT NUM
                                17
                                           //Comport Number of USB2DXL
        BAUD RATE NUM
                                            //Baudrate Number of Dynamixel PRO. 1 is 57600 bps
#define
                                 1
#define P TORQUE ENABLE
                                562
                                           //Address of Torque Enable in Control Table
#define ID
                                      //ID of Dynamixel PRO you use
                              1
int main(void)
{
       SerialPort sp = \{0,0,0,0,0\};
       SerialPort *Port = &sp;
       //Open the port of USB2DXL
       if(dxl initialize(Port, COM PORT NUM, BAUD RATE NUM) == COMM RXSUCCESS)
               printf("Succeed to open USB2Dynamixel!\n");
       else
        ł
               printf( "Failed to open USB2Dynamixel!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
       }
       int ErrorStatus;
       int Result;
       //Torque ON
       printf( "Press any key to turn on the torque...\n" );
        getch();
       Result = dxl write byte(Port, ID, P TORQUE ENABLE, 1, &ErrorStatus);
       //Torque OFF
       printf( "Press any key to turn off the torque...\n" );
        getch():
       Result = dxl write byte(Port, ID, P TORQUE ENABLE, 0, &ErrorStatus);
       //Close the port of USB2DXL
```

```
printf( "Press any key to terminate...\n" );
_getch();
dxl_terminate(Port);
return 0;
```

- The functions, such as dxl_initialize, dxl_write_byte, etc..., are used to communicate with Dynamixel PRO. These functions return communication results.
- **COMM_RXSUCCESS** is returned for successful communications.

```
ii. Operating Dynamixel PRO using C programming language.
                    As explained in 1.1, Dynamixel PRO's Control Table's Goal Position address is
                    562. Also, Goal Position is assigned 4 bytes of memory.
                    Therefore, a function called dxl write dword function controls Goal position.
                    (See 2.4.3)
                    The program's entire source is shown below.
main.cpp
#include <stdio.h>
#include <conio.h>
#include "dynamixel.h"
#define COM PORT NUM
                                17
                                            //Comport Number of USB2DXL
#define
        BAUD RATE NUM
                                 1
                                            //Baudrate Number of Dynamixel PRO. 1 is 57600 bps
#define P TORQUE ENABLE
                                           //Address of Torque Enable in Control Table
                                 562
#define P GOAL POSITION
                               596
                                         //Address of Goal Position in Control Table
#define ID
                               1
                                      //ID of Dynamixel PRO you use
int main(void)
{
       SerialPort sp = \{0, 0, 0, 0, 0\};
       SerialPort *Port = &sp;
       //Open the port of USB2DXL
       if(dxl initialize(Port, COM PORT NUM, BAUD RATE NUM) == COMM RXSUCCESS)
               printf("Succeed to open USB2Dynamixel!\n");
       else
       ł
               printf( "Failed to open USB2Dynamixel!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
       }
       int Result, ErrorStatus;
       //Torque ON
       printf( "Torque On...\n" );
       Result = dxl write byte(Port, ID, P TORQUE ENABLE, 1, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
       {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               _getch();
               return 0;
       }
       int Goal Pos1 = 100000, Goal Pos2 = -100000;
```

```
//Change the vlaue of goal position
printf("Press any key to roatate the Dynamixel PRO to position 1\n");
getch();
Result = dx1 write dword(Port, ID, P GOAL POSITION, Goal Pos1, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( " Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        getch();
        return 0;
}
//Change the vlaue of goal position
printf("Press any key to roatate the Dynamixel PRO to position 2\n");
_getch():
Result = dx1 write dword(Port, ID, P GOAL POSITION, Goal Pos2, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( " Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        _getch();
        return 0;
}
//Torque OFF
printf( "Press any key to turn off the torque...\n" );
getch():
Result = dxl_write_dword(Port, ID, P_TORQUE_ENABLE, 0, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( " Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        getch();
        return 0;
}
//Close the port of USB2DXL
printf( "Press any key to terminate...\n" );
getch();
dx1 terminate(Port);
return 0;
```

iii. Modifying Dynamixel PRO's ID using C programming language.

- As explained in 1.1, the address for the ID is 7 on the Control Table. ID can be modified with the function called **dxl_write_byte**.
- However, ID cannot be changed while the Torque Enable is on, so the torque must be turned off in order to change its value.
- The program's entire source is shown below.

main.cpp					
<pre>#include <stdio.h></stdio.h></pre>					
<pre>#include <conio.h></conio.h></pre>					
<pre>#include "dynamixel.h"</pre>					
#define COM PORT NUM	17	//Comport Number of USB2DXL			
#define BAUD RATE NUM	1	//Baudrate Number of Dynamixel PRO			
		2			
#define P ID 7	// <i>I</i>	Address of ID in Contorl Table			
#define P_TOROUE_ENABLE	562	//Address of Torque Enable in Control Table			
	002				
#define ID 1	//T	D of Dynamixel PRO you use			
	// 1				
// Print error bit of status packet					
void PrintErrorCode(int ErrorCode))				
)				
if(ErrorCode & ERRBIT VOI	(TAGE)				
printf("Input voltage error	r!\n").				
printi(input voltage error	I:∖II),				
if Error Code & ERRRIT AND	JE)				
n(EnorCode & EKKBIT_ANC					
printi(Angle filmt erfort)	\II),				
f(ErrorCode & EDDDIT OVI					
II(EITOICOUE & EKKBII_UVEKHEAI)					
printi("Overneat error!\n");				
Company Code & EDDDIT DAN					
II(ErrorCode & ERKBII_KANGE)					
print("Out of range error!\n");					
if(ErrorCode & ERRBII_CHE		1)			
printf("Checksum error!\r	n");				
if(ErrorCode & ERRBIT_OVI	ERLOAD)			
printf("Overload error!\n'	');				
if(ErrorCode & ERRBIT_INS'	TRUCTI	ON)			
printf("Instruction code en	rror!\n");				
}					
int main(void)					
{					
SerialPort sp = $\{0,0,0,0,0\}$;					
SerialPort *Port = &sp					

```
//Open the port of USB2DXL
if(dxl initialize(Port, COM PORT NUM, BAUD RATE NUM) == COMM RXSUCCESS)
       printf("Succeed to open USB2Dynamixel!\n");
else
{
       printf( "Failed to open USB2Dynamixel!\n" );
       printf( "Press any key to terminate...\n" );
       _getch();
       return 0;
}
int Result, ErrorStatus;
//Torque Off
printf( "Torque Off...\n" );
Result = dxl write_byte(Port, ID, P_TORQUE_ENABLE, 0, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
       printf( "Failed to write!\n" );
       printf( "Press any key to terminate...\n" ):
       _getch();
       return 0;
}
int desired ID = 3;
//Change the ID of Dynamixel PRO you use
printf("Press any key to change the ID of Dynamixel PRO you use\n");
getch();
Result = dxl write byte(Port, ID, P ID, desired ID, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
       printf( "Failed to write!\n" );
       printf( "Press any key to terminate...\n" );
        getch();
       return 0;
}
else
ł
       if (ErrorStatus != 0)
                PrintErrorCode(ErrorStatus);
       else
               printf("Succeed to chagne the ID of Dynamixel PRO you use!\n");
}
//Close the port of USB2DXL
printf( "Press any key to terminate...\n" );
getch();
dxl terminate(Port);
```

return 0;		

- The return pointer of the read/write functions returns the error/status code sent from Dynamixel PRO.
- Please refer to 2.4.2 for detailed information regarding error values.
- Check the ID that is modified using Dynamixel Wizard.

8	Dyn	amixel Wizard	ł	- 🗉 🗙
COM24 🔹 🖌 🖉	n 🔰 🕃 🕀	0.		
[ID:003] H42		bps	Search	O DXL 1.0
	▶ 0	2400		
	1	57600	•	
	2	1152 <mark>0</mark> 0		
	3	1000000		
	4	2000000		Start searching
	5	3000000		otar ocaranny
				Stop searching
	Search Con	nplete		

- iv. Modifying the baud rate of Dynamixel PRO.
 - Please use Dynamixel Wizard to change Dynamixel PRO ID to 1.
 - As explained in 1.1, the baud rate of Dynamixel PRO on the Control Table is in address #8. Also, the baud rate is assigned with 1 byte so it can be modified by implementing dxl_write_byte function.
 - The baud rate cannot be changed while Dynamixel PRO is in torque on status. Torque must be turned off in order to change its value.
 - The program's entire source is shown below.

main.cpp)		
#include	<stdio.h></stdio.h>		
#include	<conio.h></conio.h>		
#include	"dynamixel.h"		
	-		
#define	COM_PORT_NUM	17	//Comport Number of USB2DXL
#define	BAUD_RATE_NUM	1	//Baudrate Number of Dynamixel PRO
#define	P_BAUD_RATE	8	//Address of ID in Contorl Table
#define	P_TORQUE_ENABLE	562	//Address of Torque Enable in Control Table
#define I	D 1		ID of Dynamixel PRO you use

```
// Print error bit of status packet
void PrintErrorCode(int ErrorCode)
{
    if(ErrorCode & ERRBIT_VOLTAGE)
         printf("Input voltage error!\n");
    if(ErrorCode & ERRBIT ANGLE)
         printf("Angle limit error!\n");
    if(ErrorCode & ERRBIT OVERHEAT)
         printf("Overheat error!\n");
    if(ErrorCode & ERRBIT RANGE)
         printf("Out of range error!\n");
    if(ErrorCode & ERRBIT CHECKSUM)
         printf("Checksum error!\n");
    if(ErrorCode & ERRBIT OVERLOAD)
         printf("Overload error!\n");
    if(ErrorCode & ERRBIT INSTRUCTION)
         printf("Instruction code error!\n");
}
int main(void)
{
       SerialPort sp = \{0,0,0,0,0\};
       SerialPort *Port = &sp;
       //Open the port of USB2DXL
       if(dxl initialize(Port, COM PORT NUM, BAUD RATE NUM) == COMM RXSUCCESS)
               printf("Succeed to open USB2Dynamixel!\n");
       else
       ł
               printf( "Failed to open USB2Dynamixel!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
       }
       int Result, ErrorStatus;
       //Torque Off
       printf( "Torque Off...\n" );
       Result = dxl write byte(Port, ID, P TORQUE ENABLE, 0, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
       {
               printf( "Failed to write!\n" );
```

```
printf( "Press any key to terminate...\n" );
        _getch();
        return 0;
}
int desired Baudrate = 3; //1 Mbps
//Change the ID of Dynamixel PRO you use
printf("Press any key to change the baudrate of Dynamixel PRO you use\n");
_getch();
Result = dxl write byte(Port, ID, P BAUD RATE, 3, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        _getch();
        return 0;
}
else
ł
        if(ErrorStatus != 0)
                PrintErrorCode(ErrorStatus);
        else
                printf("Succeed to chage the baudrate of Dynamixel PRO you use!\n");
}
//Close the port of USB2DXL
printf( "Press any key to terminate...\n" );
_getch():
dx1 terminate(Port);
return 0;
```

_

<u>68</u>	Dyr	amixel Wizard	ł	- 🗆 🗙
COM24 🔹 🖉 🌋	m 🗲 🕃 🕀) 🕜 🔸		
COM24	0 1 2 3 4 5	bps 2400 57600 115200 2000000 2000000 3000000	Search	O DXL 1.0 (DXL 2.0) Start searching Stop searching
	Search Cor	nplete		

Use Dynamixel Wizard to check if the baud rate has been changed to 1 Mbps.

- If the baud rate has not been changed check if Dynamixel PRO is in torque off status.
- The following content assumes Dynamixel PRO's baud rate set to 1 Mbps.

v. LED control of Dynamixel PRO							
- The baud rate is now 1 Mbps therefore the code must be implemented as follows:							
#define COM_PORT_NUM 17 //Comport Number of USB2DXL							
#define BAUD_RATE_NUM 3 //Baudrate Number of Dynamixel PRO							
 As explained in 1.1, Dynamixel PRO Control Table contrains 3 addresses for the LEDs, each assigned with 1 byte of memory: LED_RED (#563), LED_GREEN (#564), and LED_BLUE (#565). Implement dxl_write_byte function to control its values. Since LED is assigned with 1 byte of memory select values between 0~255 to control the brightness of the LED. The program's entire source is shown below. 							
main.cpp							
#include <stdio.h></stdio.h>							
#include <conio.h></conio.h>							
#define COM PORT NUM 17 //Comport Number of USB2DXL							
#define BAUD_RATE_NUM 3 //Baudrate Number of Dynamixel PRO							
#defineP_LED_RED563//Address of LED_RED in Contorl Table#defineP_LED_GREEN564//Address of LED_RED in Contorl Table#defineP_LED_BLUE565//Address of LED_RED in Contorl Table							
#define ID 1 //ID of Dynamixel PRO you use							
// Print error bit of status packet void PrintErrorCode(int ErrorCode)							
printf("Input voltage error!\n");							
if(ErrorCode & ERRBIT_ANGLE)							
printf("Angle limit error!\n");							
if(ErrorCode & EBRBIT OVERHEAT)							
printf("Overheat error!\n"):							
printi (Overheat error: ur),							
if(ErrorCode & ERRBIT_RANGE)							
printf("Out of range error!\n");							
if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n");							
if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n");							
if(ErrorCode & ERRBIT_INSTRUCTION)							

```
printf("Instruction code error!\n");
}
int main(void)
{
       SerialPort sp = \{0, 0, 0, 0, 0\};
       SerialPort *Port = &sp;
       //Open the port of USB2DXL
       if(dxl initialize(Port, COM PORT NUM, BAUD RATE NUM) == COMM RXSUCCESS)
               printf("Succeed to open USB2Dynamixel!\n");
       else
        {
               printf( "Failed to open USB2Dynamixel!\n" );
               printf( "Press any key to terminate...\n" );
               _getch():
               return 0;
        }
       int Result, ErrorStatus;
       //Turn off the LED
       Result = dxl write byte(Port, ID, P LED RED,
                                                         0, &ErrorStatus);
       Result = dx1 write byte(Port, ID, P LED GREEN, 0, &ErrorStatus);
       Result = dxl write byte(Port, ID, P LED BLUE, 0, &ErrorStatus);
       //Turn on and change the color of LED in Dynamixel PRO
       printf("Press any key to change the color of LED in Dynamixel PRO\n");
        getch();
       Result = dxl write byte(Port, ID, P LED RED, 255, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
        ł
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
        }
       else
        {
               if (ErrorStatus != 0)
                       PrintErrorCode(ErrorStatus);
               else
                       printf("Succeed to change the color of LED in Dynamixel PRO!\n");
        }
       printf("Press any key to change the color of LED in Dynamixel PRO\n");
        getch();
       Result = dxl write byte(Port, ID, P LED RED,
                                                           0, &ErrorStatus);
       Result = dxl write byte(Port, ID, P LED GREEN, 255, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
```

```
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        getch();
        return 0;
}
else
ł
        if(ErrorStatus != 0)
                PrintErrorCode(ErrorStatus);
        else
                printf("Succeed to change the color of LED in Dynamixel PRO!\n");
}
printf("Press any key to change the color of LED in Dynamixel PRO\n");
_getch();
Result = dxl write byte(Port, ID, P LED GREEN, 0, &ErrorStatus);
Result = dxl_write_byte(Port, ID, P_LED_BLUE, 255, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        getch();
        return 0;
}
else
ł
        if (ErrorStatus != 0)
                PrintErrorCode(ErrorStatus);
        else
                printf("Succeed to change the color of LED in Dynamixel PRO!\n");
}
//Close the port of USB2DXL
printf( "Press any key to terminate...\n" );
getch();
dx1 terminate(Port);
return 0;
```

- Try controlling the intensity of the LED light.

- vi. Modifying the P gain value of Dynamixel PRO
 - Let's control the P gain value of Dynamixel PRO's position.
 - Dynamixel PRO implements PID control to maniputate its motor; therefore, the movement changes cooresponding to the gain values.
 - On Dynamixel PRO's Control Table, the Position_P_Gain address is set as 594 and it utilizes 2 bytes (1 word) of memory.
 - Thus, dxl_write_word function, which modifies 2 bytes of memory, changes the P gain value.
 - The program's entire source is shown below.

main.cpp								
#include	#include <stdio.h></stdio.h>							
#include	<pre>#include <conio.h></conio.h></pre>							
#include	"dynamixel.h"							
	-							
#define	COM_PORT_NUM	17	//Comport Number of USB2DXL					
#define	efine BAUD RATE NUM 3 //Baudrate Number of Dynamixel PRO							
#define	efine P TORQUE ENABLE 562 //Address of Torque Enable in Control Table							
#define	fine P POSITION P GAIN 594 //Address of Position P Gain in Contorl Table							
#define	P GOAL POSITION	596	//Address of Goal Position in Contorl Table					
#define I	D 1	//ID o	f Dynamixel PRO you use					
// Print en	rror bit of status packet							
void Prin	tErrorCode(int ErrorCode)							
{								
if(E	rrorCode & ERRBIT_VOLT	AGE)						
	printf("Input voltage error!\	n");						
if(E	rrorCode & ERRBIT_ANGL	E)						
	printf("Angle limit error!\n");						
if(E	if(ErrorCode & ERRBIT_OVERHEAT)							
	printf("Overheat error!\n");							
	• • • • • • • • • • • • • • • • • • • •							
if(E	if(ErrorCode & ERRBIT_RANGE)							
	printf("Out of range error!\n");							
if(E	rrorCode & ERRBIT_CHEC	KSUM)						
	printf("Checksum error!\n");							
if(E	if(ErrorCode & ERRBIT_OVERLOAD)							
	printf("Overload error!\n");							
if(E	rrorCode & ERRBIT_INSTR	UCTION)					
	printf("Instruction code error	or!∖n");						
}								
int main(void)							

{

```
SerialPort sp = \{0, 0, 0, 0, 0\};
SerialPort *Port = &sp;
//Open the port of USB2DXL
if(dxl initialize(Port, COM PORT NUM, BAUD RATE NUM) == COMM RXSUCCESS)
       printf("Succeed to open USB2Dynamixel!\n");
else
{
       printf( "Failed to open USB2Dynamixel!\n" );
       printf( "Press any key to terminate...\n" );
        getch();
       return 0;
}
int Result, ErrorStatus;
//Torque on
printf( "Torque on...\n" );
Result = dxl write byte(Port, ID, P TORQUE ENABLE, 1, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
       printf( "Failed to write!\n" );
       printf( "Press any key to terminate...\n" );
        getch();
       return 0;
}
//change the position p gain
printf("Press any key to change the position p gain\n");
getch();
Result = dxl write word(Port, ID, P POSITION P GAIN, 8, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
       printf( "Failed to write!\n" );
       printf( "Press any key to terminate...\n" );
        _getch();
       return 0;
}
else
ł
       if (ErrorStatus != 0)
                PrintErrorCode(ErrorStatus);
}
//change the goal position value
printf("Press any key to change the goal position\n");
getch();
Result = dxl_write_dword(Port, ID, P GOAL POSITION, 100000, &ErrorStatus);
if( Result != COMM RXSUCCESS )
```

```
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        getch();
        return 0;
}
else
{
        if (ErrorStatus != 0)
                PrintErrorCode(ErrorStatus);
}
//change the position p gain
printf("Press any key to change the position p gain\n");
getch();
Result = dxl write word(Port, ID, P POSITION P GAIN, 256, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        _getch();
        return 0;
}
else
{
        if (ErrorStatus != 0)
                PrintErrorCode(ErrorStatus);
}
//change the goal position value
printf("Press any key to change the goal position\n");
getch();
Result = dxl write dword(Port, ID, P GOAL POSITION, -100000, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        getch();
        return 0;
}
else
{
        if (ErrorStatus != 0)
                PrintErrorCode(ErrorStatus);
}
//Close the port of USB2DXL
printf( "Press any key to terminate...\n" );
getch();
dxl terminate(Port);
```

return 0;

1

- Observe the difference in Dynamixel PRO's movement speed relevant to the change in P Gain values.

- vii. Operating Dynamixel PRO in various speeds
 - To control the movement speed in Joint Mode change the Goal Velocity.
 - There are two methods in changing the Goal Position and Goal Velocity of Dynamixel PRO.
 - First, implement dxl_write_dword function twice to change the values. Second, implement dxl_write function to simultaneously change the Goal Position and Goal Velocity (see 2.44).
 - Since we have gone over dxl_write_byte, dxl_write_word, and dxl_write_dword, please try the second method to change the Goal Position and Goal Velocity of Dynamixel PRO.
 - Goal Position and Goal Velocity both use 2 bytes each (or 1 word each).
 - Thus, in order to simultaneously change the Goal Position and Goal Velocity, 8 bytes of fata needs to be transferred. You need to modify the data accordingly.
 - The data can be written like the example below.

int position, velocity;
position = 100000;
velocity = $10000;$
//Make a tx data
unsigned char data[8];
data[0] = DXL_LOBYTE(DXL_LOWORD(position));
data[1] = DXL_HIBYTE(DXL_LOWORD(position));
data[2] = DXL_LOBYTE(DXL_HIWORD(position));
data[3] = DXL_HIBYTE(DXL_HIWORD(position));
data[4] = DXL LOBYTE(DXL LOWORD(velocity));
data[5] = DXL_HIBYTE(DXL_LOWORD(velocity));
data[6] = DXL_LOBYTE(DXL_HIWORD(velocity));
data[7] = DXL_HIBYTE(DXL_HIWORD(velocity));

- The goal position and goal velocity values are entered into the unsigned char type array implementing DXL_LOWORD, DXL_HIWORD, DXL_LOBYTE, and DXL_HIBYTE functions.
- Next, use **dxl_write** function to send the configured data to Dynamixel PRO.

dxi write(Port, ID, P GOAL POSITION, 8, data, & ErrorStatus);	d	xl write(Port,	ID, P	GOAL	POSITION,	8, data	, &ErrorStatus);
---	---	----------------	-------	------	-----------	---------	------------------

- To simultaneously change the Goal Position and Goal Velocity implement **dxl_write** function to modify 8 bytes of memory.
- The program's entire source is shown below.

main.cpp						
#include	<stdio.h></stdio.h>					
#include	<conio.h></conio.h>					
#include	#include "dynamixel.h"					
#define	COM_PORT_NUM	17	//Comport Number of USB2DXL			
#define	BAUD_RATE_NUM	3	//Baudrate Number of Dynamixel PRO			

#define #define	P_TORQUE_ENABLE P_GOAL_POSITION	562 596	//Address of Torque Enable in Control Table //Address of Goal Position in Contorl Table
#define I	D 1	//ID of	Dynamixel PRO you use
// Print en void Prin	rror bit of status packet tErrorCode(int ErrorCode)		
if(E	rrorCode & ERRBIT_VOL printf("Input voltage error	LTAGE) !\n");	
if(E	rrorCode & ERRBIT_ANC printf("Angle limit error!\	GLE) n");	
if(E	rrorCode & ERRBIT_OVE printf("Overheat error!\n"	ERHEAT));	
if(E	rrorCode & ERRBIT_RAN printf("Out of range error!	₩GE) !\n");	
if(E	rrorCode & ERRBIT_CHE printf("Checksum error!\n	CCKSUM) ");	
if(E	rrorCode & ERRBIT_OVE printf("Overload error!\n"	ERLOAD));	
if(E	rrorCode & ERRBIT_INST printf("Instruction code er	TRUCTION) ror!\n");	
int main(void)		
{	SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp		
/ i	/Open the port of USB2DX f(dxl_initialize(Port, COM printf("Succeed to o	KL _PORT_NUM open USB2D	M, BAUD_RATE_NUM) == COMM_RXSUCCESS) ynamixel!\n");
e {			
	printf("Failed to op printf("Press any k _getch(); return 0:	ey to termina	namixel!\n"); ite\n");
}			
i	nt Result, ErrorStatus;		
/ I	/Torque on printf("Torque on\n");		

```
Result = dxl write byte(Port, ID, P TORQUE ENABLE, 1, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
       printf( "Failed to write!\n" ):
       printf( "Press any key to terminate...\n" );
       getch();
       return 0;
}
int position, velocity;
position = 100000;
velocity = 10000;
//Make a tx data
unsigned char data[8];
data[0] = DXL LOBYTE(DXL LOWORD(position));
data[1] = DXL HIBYTE(DXL LOWORD(position));
data[2] = DXL LOBYTE(DXL HIWORD(position));
data[3] = DXL HIBYTE(DXL HIWORD(position));
data[4] = DXL LOBYTE(DXL LOWORD(velocity));
data[5] = DXL HIBYTE(DXL LOWORD(velocity));
data[6] = DXL LOBYTE(DXL HIWORD(velocity));
data[7] = DXL HIBYTE(DXL HIWORD(velocity));
//change the position value and moving speed
printf( "Press any key to change the position value and moving speed...\n" );
getch();
Result = dxl write(Port, ID, P GOAL POSITION, 8, data, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
       printf( "Failed to write!\n" );
       printf( "Press any key to terminate...\n" );
       _getch();
       return 0;
}
else
{
       if(ErrorStatus != 0)
              PrintErrorCode(ErrorStatus);
}
position = -100000;
velocity =
             2000;
//Make a tx data
data[0] = DXL LOBYTE(DXL LOWORD(position));
data[1] = DXL HIBYTE(DXL LOWORD(position));
data[2] = DXL LOBYTE(DXL HIWORD(position));
data[3] = DXL HIBYTE(DXL HIWORD(position));
data[4] = DXL LOBYTE(DXL LOWORD(velocity));
```

```
data[5] = DXL HIBYTE(DXL LOWORD(velocity));
data[6] = DXL LOBYTE(DXL HIWORD(velocity));
data[7] = DXL_HIBYTE(DXL_HIWORD(velocity));
//change the position value and moving speed
printf("Press any key to change the position value and moving speed...\n");
_getch():
Result = dxl write(Port, ID, P GOAL POSITION, 8, data, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
       printf( "Failed to write!\n" );
       printf( "Press any key to terminate...\n" );
       getch();
       return 0;
}
else
{
       if (ErrorStatus != 0)
               PrintErrorCode(ErrorStatus);
}
//Close the port of USB2DXL
printf( "Press any key to terminate...\n" );
_getch();
dx1 terminate(Port);
return 0;
```

- The velocity should be fast when rotating towards the first position and slower when rotating towards the second position.

viii. Internal temperature feedback of Dynamixel PRO

- So far, we have covered how to use the Write command to send commands to Dynamixel PRO.
- Next, we will go over how to use the Read command to receive data from Dynamixel PRO.
- The address of Present Temperature on the Control Table is 625 and is assigned 1 byte of memory.
- Thus, dxl_read_byte function can be used to obtain the Current Temperature of Dynamixel PRO.
- The program's entire source is shown below.

nain.cpp					
<pre>#include <stdio.h></stdio.h></pre>					
<pre>#include <conio.h></conio.h></pre>					
<pre>#include "dynamixel.h"</pre>					
#defineCOM_PORT_NUM17//Comport Number of USB2DXL#defineBAUD_RATE_NUM3//Baudrate Number of Dynamixel PRO#defineP_PRESENT_TEMPERATURE625//Address of Present Temperature in ContorlFableFableFableFable					
define ID 1 //ID of Dynamixel PRO you use					
/ Print error bit of status packet void PrintErrorCode(int ErrorCode)					
if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n");					
if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n");					
if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n");					
if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n");					
if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n");					
if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n");					
if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n");					
nt main(void)					

```
{
       SerialPort sp = \{0, 0, 0, 0, 0\};
       SerialPort *Port = &sp;
       //Open the port of USB2DXL
       if(dxl initialize(Port, COM PORT NUM, BAUD RATE NUM) == COMM RXSUCCESS)
               printf("Succeed to open USB2Dynamixel!\n");
       else
        {
               printf( "Failed to open USB2Dynamixel!\n" );
               printf( "Press any key to terminate...\n" );
                getch();
               return 0;
        }
       int Result, ErrorStatus;
       printf("Press any key to terminate...\n");
       printf("\n");
       int temp;
       while(true)
        {
               if( kbhit())
                       break;
               //Read the present temperature
               Result = dxl read byte(Port, ID,
                                P_PRESENT_TEMPERATURE, &temp, &ErrorStatus);
               if( Result != COMM RXSUCCESS )
                {
                       printf( "Failed to write!\n" );
                       printf( "Press any key to terminate...\n" );
                        getch();
                       return 0;
                }
               else
                {
                       PrintErrorCode(ErrorStatus);
                }
               printf("\r");
               printf("current temperature : %d", temp);
        }
       //Close the port of USB2DXL
       dx1 terminate(Port);
       return 0;
```

- The present temperature is constantly read.

ix.	Read	the	Present	Position	of Dy	namixel	PRO
					5		

- Present Position indicates the present position of Dynamixel PRO Its address is 611 and is assigned 4 bytes of memory.
- Therefore, implement **dxl_read_dword** function present position of Dynamixel PRO.
- The program's entire source is shown below.

main.cpp)						
#include	<stdio.h></stdio.h>						
#include	<conio.h></conio.h>						
#include	"dynamixel.h"						
#define #define #define	COM_PORT_NUM BAUD_RATE_NUM P_PRESENT_POSITION	17 3 611	//Comport Number of USB2DXL//Baudrate Number of Dynamixel PRO//Address of Present Position in Contorl Table				
#define I	D	1	//ID of Dynamixel PRO you use				
// Print en void Print	rror bit of status packet tErrorCode(int ErrorCode)						
{ if(E	rrorCode & ERRBIT_VOLTA printf("Input voltage error!\n	AGE) ");					
if(E	if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n");						
if(E	if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n");						
if(E	if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n");						
if(E	if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n");						
if(E	if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n");						
<pre>if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); }</pre>							
int main((void)						
	{ SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp						
/ i	/Open the port of USB2DXL f(dxl_initialize(Port, COM_P	ORT_NU	M, BAUD_RATE_NUM) == COMM_RXSUCCESS)				

```
printf("Succeed to open USB2Dynamixel!\n");
        else
        {
                printf( "Failed to open USB2Dynamixel!\n" );
                printf( "Press any key to terminate...\n" );
                _getch():
                return 0;
        ł
        int Result, ErrorStatus;
        printf("Press any key to terminate...\n" );
        printf("\n");
        int temp;
        while(true)
        {
                if(_kbhit())
                        break;
                //Read the present position
                Result = dxl read dword(Port, ID, P PRESENT POSITION, (unsigned*)&temp,
&ErrorStatus);
                if( Result != COMM RXSUCCESS )
                {
                        printf( "Failed to write!\n" );
                        printf( "Press any key to terminate...\n" );
                         _getch();
                        return 0;
                }
                else
                 ł
                        PrintErrorCode(ErrorStatus);
                 }
                printf("\r");
                printf("present position : %d", temp);
        }
        //Close the port of USB2DXL
        dxl terminate(Port);
        return 0;
```

The Present Position is shown when the program is initiated.

- x. Changing velocity of Dynamixel PRO in Wheel Mode
 - On Dynamixel PRO's Control Table the address for Operating Mode is 11. It requires 1 byte of memory. By changing the value of the Operating Mode Dynamixel PRO can be set to Joint Mode, Wheel Mode, or Torque Mode,
 - Implement dxl_write_byte to change Operating Mode. It is a part of the EEPROM area so the Torque Enable must be off to modify its value.
 - Also, as shown in 1.1, Torque Enable needs to be on to activate Dynamixel PRO in Wheel Mode. The speed of Dynamixel PRO can be controlled by modifying the Goal_Velocity values.
 - The program's entire source code is shown below.

main.cpp	ain.cpp							
#include	#include <windows.h></windows.h>							
#include	<stdio.h></stdio.h>							
#include <conio.h></conio.h>								
#include	"dvnamixel.h"							
#define	COM PORT NUM	17	//Comport Number of USB2DXL					
#define	BAUD RATE NUM 3 //Baudrate Number of Dynamixel PRO							
" define		5	"Duddide Humber of D ynumker File					
#define	P OPERATING MODE	11	//Address of Operation Mode in Control Table					
#define	P TOPOLIE ENABLE	562	//Address of Torque Enable in Control Table					
#define	P COAL VELOCITY	502 600	//Address of Cool Velecity in Control Table					
#define	P_GOAL_VELOCITY	000	//Address of Goal velocity in Control Table					
#define I	D	1	//ID of Dynamixel PRO you use					
// Print ei	rror bit of status packet							
void Prin	tErrorCode(int ErrorCode)							
{								
i	f(ErrorCode & ERRBIT_VOI	LTAGE)						
	<pre>printf("Input voltage error!\n");</pre>							
if(ErrorCode & ERRBIT_ANGLE)								
	printf("Angle limit error!\n");							
	r //							
i	f(ErrorCode & ERRBIT OVI	ERHEAT)						
	printf("Overheat error!\n"):							
	printi (Overheat error: ur),							
i	f(ErrorCode & ERRBIT RAN	NGE)						
	printf("Out of range er	ror!(n")						
	princi (Out of range error! III),							
i	if(ErrorCode & ERRBIT CHECKSUM)							
1	nrintf("Checksum erro	rl\n"))					
	printi("Cnecksum error!\n");							
;	f(ErrorCode & EPPRIT OVI							
1	nrintf("Overlagd armer		,					
	princi Overload effor	: ш <i>)</i> ,						
:	(ErrorCode & EDDDIT INS'	TDUCTIC						
1	11(ErrorCode & ERRBIT_INSTRUCTION)							
	printi("instruction cod	e error!\n),					

```
}
int main(void)
{
       SerialPort sp = \{0,0,0,0,0\};
       SerialPort *Port = &sp;
       //Open the port of USB2DXL
       if(dxl initialize(Port, COM PORT NUM, BAUD RATE NUM) == COMM RXSUCCESS)
               printf("Succeed to open USB2Dynamixel!\n");
       else
        {
               printf( "Failed to open USB2Dynamixel!\n" );
               printf( "Press any key to terminate...\n" );
               _getch();
               return 0;
        }
       int Result, ErrorStatus;
       //Torque Off
       printf( "Torque Off...\n" );
       Result = dxl write byte(Port, ID, P TORQUE ENABLE, 0, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
        {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               _getch();
               return 0;
        }
       printf("Press any key to change the operating mode...\n");
       _getch();
       Result = dx1 write byte(Port, ID, P OPERATING MODE, 1, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
        {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
        }
       else
        {
               if (ErrorStatus != 0)
                       PrintErrorCode(ErrorStatus);
               else
                       printf("Succeed to chage the operationg mode!\n");
        }
       //Torque On
```

```
printf( "Torque On...\n" );
Result = dxl write byte(Port, ID, P TORQUE ENABLE, 1, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        _getch():
        return 0;
}
//Change the Goal Velocity
printf("Press any key to change goal velocity...\n");
getch():
Result = dxl write dword(Port, ID, P GOAL VELOCITY, 5000, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        _getch():
        return 0;
}
else
ł
        if (ErrorStatus != 0)
                PrintErrorCode(ErrorStatus);
        else
                printf("Succeed to chage the operationg mode!\n");
}
//Close the port of USB2DXL
printf( "Press any key to terminate...\n" );
getch();
//Torque Off
printf( "Torque Off...\n" );
Result = dxl write byte(Port, ID, P TORQUE ENABLE, 0, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        _getch();
        return 0;
}
dxl terminate(Port);
return 0;
```

- xi. Checking the current position and current velocity of Dynamixel PRO
 - There are two methods to check the present position of Dynamixel PRO.
 - First, is to implement dxl_read_dword function twice to change the values.
 Second, is to implement dxl_read function to simultaneously read the Present Position and Present Velocity.
 - You have gone over dxl_read_byte, dxl_read_word, and dxl_tead_dword, This time implement the second method to change the Goal Position and Goal Velocity of Dynamixel PRO.
 - Both Goal Position and Goal Velocity require 4 bytes.
 - Thus, to simultaneously read the Present Position and Present Velocity you need to modify the data length accordingly.
 - Data can be read by entering the data type as shown on the example below.

unsigned char data[8]; dxl_read(Port, ID, P_PRESENT_POSITION, 8, data, &ErrorStatus);

int present_position, present_velocity;

present_position=DXL_MAKEDWORD(DXL_MAKEWORD(data[0],data[1]),DXL_MAKEWORD(data[2],data[3])); present_velocity=DXL_MAKEDWORD(DXL_MAKEWORD(data[4],data[5]),DXL_MAKEWORD(data[5]),DXL_MAKEWORD(data[5]

ta[6],data[7]));

- The unsigned char type array data obtained is converted into the present position and velocity with DXL_MAKEWORD, DXL_MAKEDWORD (refer to the source code above).
 - The program's entire source code is shown below.

<pre>#include <stdio.h> #include <conio.h> #include "dynamixel.h" #define COM_PORT_NUM 17 //Comport Number of USB2DXL #define BAUD_RATE_NUM 3 //Baudrate Number of Dynamixel PRO #define P_OPERATING_MODE 11 //Address of Operation Mode in Control Table #define P_TORQUE_ENABLE 562 //Address of Torque Enable in Control Table #define P_GOAL_POSITION 596 //Address of Goal Position in Control Table #define P_GOAL_VELOCITY 600 //Address of Goal Velocity in Control Table #define P_PRESENT_POSITION 611 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define ID 1 //ID of Dynamixel PRO you use // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERBBIT_VOLTAGE) } </conio.h></stdio.h></pre>	main.cpp							
<pre>#include <conio.h> #include "dynamixel.h" #define COM_PORT_NUM 17 //Comport Number of USB2DXL #define BAUD_RATE_NUM 3 //Baudrate Number of Dynamixel PRO #define P_OPERATING_MODE 11 //Address of Operation Mode in Control Table #define P_TORQUE_ENABLE 562 //Address of Torque Enable in Control Table #define P_GOAL_POSITION 596 //Address of Goal Position in Control Table #define P_GOAL_VELOCITY 600 //Address of Goal Velocity in Control Table #define P_PRESENT_POSITION 611 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define ID 1 //ID of Dynamixel PRO you use // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERBBIT_VOLTAGE) } </conio.h></pre>	#include	<stdio.h></stdio.h>						
<pre>#include "dynamixel.h" #define COM_PORT_NUM 17 //Comport Number of USB2DXL #define BAUD_RATE_NUM 3 //Baudrate Number of Dynamixel PRO #define P_OPERATING_MODE 11 //Address of Operation Mode in Control Table #define P_TORQUE_ENABLE 562 //Address of Torque Enable in Control Table #define P_GOAL_POSITION 596 //Address of Goal Position in Control Table #define P_GOAL_VELOCITY 600 //Address of Goal Velocity in Control Table #define P_PRESENT_POSITION 611 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define ID 1 //ID of Dynamixel PRO you use // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERBBIT_VOLTAGE) } </pre>	#include	#include <conio.h></conio.h>						
#define COM_PORT_NUM 17 //Comport Number of USB2DXL #define BAUD_RATE_NUM 3 //Baudrate Number of Dynamixel PRO #define P_OPERATING_MODE 11 //Address of Operation Mode in Control Table #define P_TORQUE_ENABLE 562 //Address of Torque Enable in Control Table #define P_GOAL_POSITION 596 //Address of Goal Position in Control Table #define P_GOAL_VELOCITY 600 //Address of Goal Velocity in Control Table #define P_PRESENT_POSITION 611 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define D_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define ID 1 //ID of Dynamixel PRO you use // Print error bit of status packet void PrintErrorCode(int ErrorCode) [iff(ErrorCode & ERRBIT_VOLTAGE) iff(ErrorCode & ERRBIT_VOLTAGE) [#include "dynamixel.h"							
#define COM_PORT_NUM 17 //Comport Number of USB2DXL #define BAUD_RATE_NUM 3 //Baudrate Number of Dynamixel PRO #define P_OPERATING_MODE 11 //Address of Operation Mode in Control Table #define P_TORQUE_ENABLE 562 //Address of Torque Enable in Control Table #define P_GOAL_POSITION 596 //Address of Goal Position in Control Table #define P_GOAL_VELOCITY 600 //Address of Goal Velocity in Control Table #define P_PRESENT_POSITION 611 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define D_PRESENT_VELOCITY 615 //Iddress of Present Velocity in Control Table #define ID 1 //ID of Dynamixel PRO you use // Print error bit of status packet void PrintErrorCode(int ErrorCode) [iffErrorCode & ERBBIT_VOLTAGE) [iffErrorCode & ERBBIT_VOLTAGE)								
#define BAUD_RATE_NUM 3 //Baudrate Number of Dynamixel PRO #define P_OPERATING_MODE 11 //Address of Operation Mode in Control Table #define P_TORQUE_ENABLE 562 //Address of Torque Enable in Control Table #define P_GOAL_POSITION 596 //Address of Goal Position in Control Table #define P_GOAL_VELOCITY 600 //Address of Goal Velocity in Control Table #define P_PRESENT_POSITION 611 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define I //ID of Dynamixel PRO you use //ID of Dynamixel PRO you use // Print error bit of status packet yoid PrintErrorCode(int ErrorCode) //ID of Dynamixel PRO you use	#define	COM_PORT_NUM	17	//Comport Number of USB2DXL				
#define P_OPERATING_MODE 11 //Address of Operation Mode in Control Table #define P_TORQUE_ENABLE 562 //Address of Torque Enable in Control Table #define P_GOAL_POSITION 596 //Address of Goal Position in Control Table #define P_GOAL_VELOCITY 600 //Address of Goal Velocity in Control Table #define P_PRESENT_POSITION 611 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define ID 1 //ID of Dynamixel PRO you use // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERRBIT_VOLTAGE)	#define	BAUD_RATE_NUM	3	//Baudrate Number of Dynamixel PRO				
#define P_OPERATING_MODE 11 //Address of Operation Mode in Control Table #define P_TORQUE_ENABLE 562 //Address of Torque Enable in Control Table #define P_GOAL_POSITION 596 //Address of Goal Position in Control Table #define P_GOAL_VELOCITY 600 //Address of Goal Velocity in Control Table #define P_GOAL_VELOCITY 600 //Address of Goal Velocity in Control Table #define P_PRESENT_POSITION 611 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define I //ID of Dynamixel PRO you use //ID of Dynamixel PRO you use // Print error bit of status packet void PrintErrorCode(int ErrorCode) //IE of Dynamixel PRO you use { if(ErrorCode & ERRBIT_VOLTAGE) //IE of Dynamixel PRO you use //IE of Dynamixel PRO you use								
#define P_TORQUE_ENABLE 562 //Address of Torque Enable in Control Table #define P_GOAL_POSITION 596 //Address of Goal Position in Control Table #define P_GOAL_VELOCITY 600 //Address of Goal Velocity in Control Table #define P_PRESENT_POSITION 611 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define ID 1 //ID of Dynamixel PRO you use // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERBBIT_VOLTAGE)	#define	P_OPERATING_MODE	11	//Address of Operation Mode in Control Table				
#define P_GOAL_POSITION 596 //Address of Goal Position in Control Table #define P_GOAL_VELOCITY 600 //Address of Goal Velocity in Control Table #define P_PRESENT_POSITION 611 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define ID 1 //ID of Dynamixel PRO you use // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERBBIT_VOLTAGE)	#define	P_TORQUE_ENABLE	562	//Address of Torque Enable in Control Table				
#define P_GOAL_VELOCITY 600 //Address of Goal Velocity in Control Table #define P_PRESENT_POSITION 611 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define ID 1 //ID of Dynamixel PRO you use // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERBBIT_VOLTAGE)	#define	P_GOAL_POSITION	596	//Address of Goal Position in Control Table				
#define P_PRESENT_POSITION 611 //Address of Present Position in Control Table #define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define ID 1 //ID of Dynamixel PRO you use // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERBBIT_VOLTAGE)	#define	P GOAL VELOCITY	600	//Address of Goal Velocity in Control Table				
<pre>#define P_PRESENT_VELOCITY 615 //Address of Present Velocity in Control Table #define ID 1 //ID of Dynamixel PRO you use // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERBBIT_VOLTAGE)</pre>	#define	P PRESENT POSITION	611	//Address of Present Position in Control Table				
#define ID 1 //ID of Dynamixel PRO you use // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERBBIT_VOLTAGE)	#define	P PRESENT VELOCITY	615	//Address of Present Velocity in Control Table				
#define ID 1 //ID of Dynamixel PRO you use // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERBBIT_VOLTAGE)								
// Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERBBIT_VOLTAGE)	#define I	D	1	//ID of Dynamixel PRO you use				
// Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERBBIT_VOLTAGE)								
<pre>void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERBBIT_VOLTAGE)</pre>	// Print en	rror bit of status packet						
{ if(ErrorCode & ERRBIT_VOLTAGE)	void PrintErrorCode(int ErrorCode)							
if(ErrorCode & ERRBIT_VOLTAGE)	{	{						
	i	f(ErrorCode & ERRBIT_VOI	TAGE)					
printf("Input voltage error!\n");		printf("Input voltage e	rror!\n");					
```
if(ErrorCode & ERRBIT ANGLE)
               printf("Angle limit error!\n");
       if(ErrorCode & ERRBIT OVERHEAT)
               printf("Overheat error!\n");
       if(ErrorCode & ERRBIT RANGE)
               printf("Out of range error!\n");
       if(ErrorCode & ERRBIT CHECKSUM)
               printf("Checksum error!\n");
       if(ErrorCode & ERRBIT OVERLOAD)
               printf("Overload error!\n");
       if(ErrorCode & ERRBIT INSTRUCTION)
               printf("Instruction code error!\n");
}
int main(void)
{
       SerialPort sp = \{0, 0, 0, 0, 0\};
       SerialPort *Port = &sp;
       //Open the port of USB2DXL
       if(dxl initialize(Port, COM PORT NUM, BAUD RATE NUM) == COMM RXSUCCESS)
               printf("Succeed to open USB2Dynamixel!\n");
       else
        {
               printf( "Failed to open USB2Dynamixel!\n" );
               printf( "Press any key to terminate...\n" );
               _getch();
               return 0;
        }
       int Result, ErrorStatus;
       //Torque Off
       printf( "Torque Off...\n" );
       Result = dxl write byte(Port, ID, P TORQUE ENABLE, 0, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
        {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               _getch();
               return 0;
        }
       printf("Press any key to change the operating mode...\n");
```

```
getch();
Result = dx1 write byte(Port, ID, P OPERATING MODE, 1, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        _getch():
        return 0;
}
else
ł
        if(ErrorStatus != 0)
                PrintErrorCode(ErrorStatus);
        else
                printf("Succeed to chage the operationg mode!\n");
}
//Torque On
printf( "Torque On...\n" );
Result = dx1 write byte(Port, ID, P TORQUE ENABLE, 1, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        getch();
        return 0;
}
//Change the Goal Velocity
printf("Press any key to change goal velocity...\n");
getch():
Result = dxl write dword(Port, ID, P GOAL VELOCITY, 5000, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        _getch();
        return 0;
}
else
{
        if (ErrorStatus != 0)
                PrintErrorCode(ErrorStatus);
        else
                printf("Succeed to chage the operationg mode!\n");
}
printf("Press any key to terminate...\n");
printf("\n");
```

```
while(true)
        Ş
               if( kbhit())
                       break:
               unsigned char data[8];
               dx1 read(Port, ID, P PRESENT POSITION, 8, data, &ErrorStatus);
               int present position, present velocity;
               present position = DXL MAKEDWORD( DXL MAKEWORD(data[0], data[1]),
DXL MAKEWORD(data[2], data[3]) );
               present velocity = DXL MAKEDWORD( DXL MAKEWORD(data[4], data[5]),
DXL MAKEWORD(data[6], data[7]));
               printf("\r");
               printf("present position : %d, presen velocity : %d", present position,
present velocity);
        ł
       printf("\n");
       //Close the port of USB2DXL
       printf( "Press any key to terminate...\n" );
       getch();
       //Torque Off
       printf( "Torque Off...\n" );
       Result = dxl_write_byte(Port, ID, P_TORQUE_ENABLE, 0, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
        {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               _getch():
               return 0;
        }
       dx1 terminate(Port);
       return 0;
```

1.2.3 C programming language functions.

- i. Modifying the zero value of Dynamixel PRO
 - You can modify the zero-position of Dynamixel PRO.
 - Change the Control table's Homing Offset to modify Dynamixel PRO's zero-position.
 - Homing Offset address on the Control Tableis 13 and requires 4 bytes of memory. Therefore, implement dxl_write_dword to modify the Homing Offset value.
 - Homing Offset is a part of the EEPROM area, which requires the Torque Enable to be turned off be for changes.
 - When modifying the Homing Offset value, designate the zero position by inverting the desired zero position with a negative sign (-).
 - For instance, to set the Position Valu 50000 as the zero position, -50000 needs to be entered as the Homing Offset value.
 - The program's entire source code is shown below.
 - Check if the Operation Mode is set as 'Joint Mode' before running the program below.

main.cpp					
#include	#include <stdio.h></stdio.h>				
#include	<conio.h></conio.h>				
#include	"dynamixel.h"				
	-				
#define	COM PORT NUM	17	//Comport Number of USB2DXL		
#define	BAUD RATE NUM	3	//Baudrate Number of Dynamixel PRO		
			5		
#define	P HOMING OFFSET	13	//Address of Homing Offset in Control Table		
#define	P TOROUE ENABLE	562	//Address of Torque Enable in Control Table		
#define	P GOAL POSITION	596	//Address of Goal Position in Control Table		
		0 / 0			
#define I	D	1	//ID of Dynamixel PRO you use		
i define i		1	The of D finalities into you use		
// Print er	rror bit of status packet				
void Prin	tErrorCode(int ErrorCode)				
۲۰۱۵ ۲ ۲ ۱۱۱۱ ۲	f				
્યું	if/ErrorCode & EDDDIT VOLTACE)				
1	printf("Input voltage e	rrorl\n").			
	printi(input voltage error!\n`);				
:	Contractor & EDDDIT ANG				
1	I(EIIOICOde & EKKBII_ANC	JLE)			
	printi(Angle limit erro	Or!(n);			
	CERTER C. J. & EDDDIT OVI				
1	I(ErrorCode & ERRBII_OVE	KHEAI)			
	printf("Overheat error!\n");				
II(ErrorCode & EKKBI1_KANGE)					
	printf("Out of range er	ror! n'');			
1	t(ErrorCode & ERRBIT_CHE	CKSUM)		

```
printf("Checksum error!\n");
       if(ErrorCode & ERRBIT OVERLOAD)
               printf("Overload error!\n");
       if(ErrorCode & ERRBIT INSTRUCTION)
               printf("Instruction code error!\n");
}
int main(void)
{
       SerialPort sp = \{0,0,0,0,0\};
       SerialPort *Port = &sp;
       //Open the port of USB2DXL
       if(dxl initialize(Port, COM_PORT_NUM, BAUD_RATE_NUM) == COMM_RXSUCCESS)
               printf("Succeed to open USB2Dynamixel!\n");
       else
        ł
               printf( "Failed to open USB2Dynamixel!\n" );
               printf( "Press any key to terminate...\n" );
               _getch();
               return 0;
        }
       int Result, ErrorStatus;
       //Torque Off
       printf( "Torque Off...\n" );
       Result = dxl write byte(Port, ID, P TORQUE ENABLE, 0, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
        {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
        }
       //Change the zero point
       printf("Press any key to change the zero point...\n");
        getch();
       Result = dxl write dword(Port, ID, P HOMING OFFSET, -50000, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
        {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
        }
       else
```

```
{
        if (ErrorStatus != 0)
                PrintErrorCode(ErrorStatus);
        else
                printf("Succeed to chage the homing offset!\n");
}
//Torque On
printf( "Torque On...\n" );
Result = dxl write byte(Port, ID, P TORQUE ENABLE, 1, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        getch();
        return 0;
}
//Change the Goal position
printf("Press any key to change the Goal Position...\n");
getch();
Result = dxl write dword(Port, ID, P GOAL POSITION, 0, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        _getch();
        return 0;
}
else
        PrintErrorCode(ErrorStatus);
//Close the port of USB2DXL
printf( "Press any key to terminate...\n" );
_getch();
dxl terminate(Port);
return 0;
```

- If the program does not run properly check if the Operating Mode is set to Joint Mode.

ii. Limiting the operating range of Dynamixel PRO

- Please refer to 1.1 for information regarding +, Position Limit.
- The program's entire source code is shown below.

main.cpp	main.cpp					
#include	<stdio.h></stdio.h>					
#include	<conio.h></conio.h>					
#include	"dynamixel.h"					
	-					
#define	COM PORT NUM	17	//Comport Number of USB2DXL			
#define	BAUD RATE NUM	3	//Baudrate Number of Dynamixel PRO			
	21102_10112_10011	U	,, <i>2</i>			
#define	P PLUS POSITION LIMIT	36	//Address of Plus Position Limit in Control Table			
#dofino	D MINUS DOSITION LIMIT	40	//Address of Minus Desition Limit in Control			
Tabla		40	//Address of Winds I Osition Limit in Control			
	D TODOLIE ENADLE	5()	//A damage of Tengene Englishin Construct Tells			
#define	P_IORQUE_ENABLE	562	//Address of Torque Enable in Control Table			
#define	P_GOAL_POSITION 5	96	//Address of Goal Position in Control Table			
#define I	ID 1	//IC	of Dynamixel PRO you use			
// Print en	error bit of status packet					
void Prin	ntErrorCode(int ErrorCode)					
{						
i	if(ErrorCode & ERRBIT_VOLTA)	GE)				
1	printf("Input voltage error!	(n'')				
	princi(input voltage error:	, <u>,</u>				
:						
1	II(EITOFCOGE & EKKBII_ANGLE)					
	printf("Angle limit error!\n");					
1	if(ErrorCode & ERRBIT_OVERHEAT)					
	printf("Overheat error!\n");					
i	if(ErrorCode & ERRBIT RANGE))				
	printf("Out of range error!	(n"):				
	r (thirthe official					
i	if(ErrorCode & ERRBIT_CHECKS	SUM)				
1	printf("Checksum errorl\n"	י).				
	printi(Checksum ciror: (ii),				
1	II(ErrorCode & ERRBII_OVERLO	JAD)				
	printf("Overload error!\n");					
i	<pre>it(ErrorCode & ERRBIT_INSTRU</pre>	CTION)				
	printf("Instruction code err	or!\n");				
}						
int main((void)					
{	(· - · · · · · · · · · · · · · · · · ·					
((SerialPort sn = $\{0, 0, 0, 0, 0\}$.					
L	SerialPort *Port - kon.					
L.	$scharon = \alpha sp,$					

```
//Open the port of USB2DXL
if(dxl initialize(Port, COM PORT NUM, BAUD_RATE_NUM) == COMM_RXSUCCESS)
       printf("Succeed to open USB2Dynamixel!\n");
else
ł
       printf( "Failed to open USB2Dynamixel!\n" );
       printf( "Press any key to terminate...\n" );
        getch();
       return 0;
}
int Result, ErrorStatus;
//Torque Off
printf( "Torque Off...\n" );
Result = dx1 write byte(Port, ID, P_TORQUE_ENABLE, 0, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
       printf( "Failed to write!\n" );
       printf( "Press any key to terminate...\n" );
        _getch();
       return 0;
}
//Change the Plus Position Limit
printf("Press any key to change the plus position limit...\n");
getch():
Result = dxl write dword(Port, ID, P PLUS POSITION LIMIT, 100000, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
       printf( "Failed to write!\n" );
       printf( "Press any key to terminate...\n" );
        getch();
       return 0;
}
else
ł
       if(ErrorStatus != 0)
               PrintErrorCode(ErrorStatus);
       else
               printf("Succeed to chage the plus position limit!\n");
}
//Change the Minus Position Limit
printf("Press any key to change the minus position limit...\n");
getch();
Result = dxl write dword(Port, ID, P MINUS POSITION LIMIT, -100000, &ErrorStatus);
if( Result != COMM RXSUCCESS )
```

```
printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        _getch();
        return 0;
}
else
ł
        if (ErrorStatus != 0)
                PrintErrorCode(ErrorStatus);
        else
                printf("Succeed to chage the minus position limit!\n");
}
//Torque On
printf( "Torque On...\n" );
Result = dxl write byte(Port, ID, P TORQUE ENABLE, 1, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        _getch();
        return 0;
}
//Change the Goal position
printf("Press any key to change the Goal Position to 120000...\n");
getch():
Result = dxl_write_dword(Port, ID, P_GOAL_POSITION, 120000, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        getch();
        return 0;
}
else
        PrintErrorCode(ErrorStatus);
//Change the Goal position
printf("Press any key to change the Goal Position to 100000\n");
getch():
Result = dxl write dword(Port, ID, P GOAL POSITION, 100000, &ErrorStatus);
if( Result != COMM RXSUCCESS )
ł
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        getch();
        return 0;
```

```
else
        PrintErrorCode(ErrorStatus);
//Change the Goal position
printf("Press any key to change the Goal Position to -120000\n");
getch();
Result = dxl_write_dword(Port, ID, P_GOAL_POSITION, -120000, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        getch();
        return 0;
}
else
        PrintErrorCode(ErrorStatus);
//Change the Goal position
printf("Press any key to change the Goal Position to -100000\n");
getch();
Result = dxl write dword(Port, ID, P GOAL POSITION, -100000, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        getch();
        return 0;
}
else
        PrintErrorCode(ErrorStatus);
//Close the port of USB2DXL
printf( "Press any key to terminate...\n" );
getch();
dx1 terminate(Port);
return 0;
```

- If the program does not run properly check if the Operating Mode is set to Joint Mode.

iii. Extending the operating range of Dynamixel PRO.

- Please refer to 1.1 for information regarding +/- Position Limits.
- The program's entire source code is shown below.

main.cpp					
#include	<stdio.h></stdio.h>				
#include	<conio.h></conio.h>				
#include	"dynamixel.h"				
#define	COM_PORT_NUM	17	//Comport Number of USB2DXL		
#define	BAUD_RATE_NUM	3	//Baudrate Number of Dynamixel PRO		
#define #define Table	P_PLUS_POSITION_LIMIT P_MINUS_POSITION_LIMIT	36 40	//Address of Plus Position Limit in Control Table //Address of Minus Position Limit in Control		
#define	P TOROUE ENABLE	562	//Address of Torque Enable in Control Table		
#define	P GOAL POSITION	596	//Address of Goal Position in Control Table		
" define		570			
#define I	D	l //I	D of Dynamixel PRO you use		
// Print en	rror bit of status packet atErrorCode(int ErrorCode)				
1	f Error Code & EPPRIT VOLT	AGE)			
1	n(EnorCode & EKKBI1_VOL1	AUE)			
	printi(input voltage end)I!\II),			
i	f(ErrorCode & ERRBIT_ANGL printf("Angle limit error	E) !\n");			
i	if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n");				
i	f(ErrorCode & ERRBIT_RANC printf("Out of range erro	iE) r!\n");			
i	f(ErrorCode & ERRBIT_CHEC printf("Checksum error!"	KSUM) n");			
i	if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n");				
i	<pre>if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n");</pre>				
}					
int main({	(void)				
(SerialPort sp = $\{0.0.0.0.0\}$;				
í.	SerialPort *Port = &sp				
	1 /				

```
//Open the port of USB2DXL
if(dxl initialize(Port, COM PORT NUM, BAUD RATE NUM) == COMM RXSUCCESS)
       printf("Succeed to open USB2Dynamixel!\n");
else
{
       printf( "Failed to open USB2Dynamixel!\n" );
       printf( "Press any key to terminate...\n" );
       _getch();
       return 0;
int Result, ErrorStatus;
//Torque Off
printf( "Torque Off...\n" );
Result = dx1 write byte(Port, ID, P_TORQUE_ENABLE, 0, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
       printf( "Failed to write!\n" );
       printf( "Press any key to terminate...\n" );
        _getch();
       return 0;
}
//Change the Plus Position Limit
printf("Press any key to change the plus position limit...\n");
getch();
Result = dxl write dword(Port, ID, P PLUS POSITION LIMIT, 5000000, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
       printf( "Failed to write!\n" ):
       printf( "Press any key to terminate...\n" );
       _getch();
       return 0;
}
else
ł
       if (ErrorStatus != 0)
               PrintErrorCode(ErrorStatus);
       else
                printf("Succeed to chage the plus position limit!\n");
}
//Change the Minus Position Limit
printf("Press any key to change the minus position limit...\n");
_getch();
Result = dx1 write dword(Port, ID, P MINUS POSITION LIMIT, -500000, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
       printf( "Failed to write!\n" );
```

```
printf( "Press any key to terminate...\n" );
        getch();
        return 0;
}
else
ł
        if(ErrorStatus != 0)
                PrintErrorCode(ErrorStatus);
        else
                printf("Succeed to chage the minus position limit!\n");
}
//Torque On
printf( "Torque On...\n" );
Result = dxl_write_byte(Port, ID, P_TORQUE_ENABLE, 1, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        _getch():
        return 0;
}
//Change the Goal position
printf("Press any key to change the Goal Position to 500000...\n");
getch();
Result = dxl write dword(Port, ID, P GOAL POSITION, 500000, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        _getch():
        return 0;
}
else
        PrintErrorCode(ErrorStatus);
//Change the Goal position
printf("Press any key to change the Goal Position to -500000\n");
getch();
Result = dxl write dword(Port, ID, P GOAL POSITION, -500000, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
        printf( "Failed to write!\n" );
        printf( "Press any key to terminate...\n" );
        _getch();
        return 0;
}
else
        PrintErrorCode(ErrorStatus);
```

//Close the port of USB2DXL
printf("Press any key to terminate...\n");
_getch();
dxl_terminate(Port);
return 0;

- If the program does not run properly check if the Operating Mode is set to Joint Mode.

- 1.2.4 Indirect Addressing function of Dynamixel PRO
 - i. Change the position, velocity, and acceleration using Indirect Address fucntion.
 - On Dynamixel PRO's Control Table-Goal Position, Goal Velocity, and Goal Acceleration addresses-are not arranged in consecutive order. Thus, dxl_write and dxl_write_dword need to be implemented 3 times to change all 3 addresses.
 - To resolve this issue, Dynamixel PRO has Indirect Address function.
 - Indirect Address assignes another address to the default one.
 - For example, Goal Position (#596) address can be assigned another address (#634).
 - Start by writing the value of the desired address in the EEPROM area of the Indirect Address.
 - The written value is the default address. Please refer to the source code below

dxl write word(Port, ID, P INDIRECT ADDRESS 0, P GOAL POSITION, &ErrorStatus); dxl write word(Port, ID, P INDIRECT ADDRESS 1, P GOAL POSITION+1, &ErrorStatus); dxl write word(Port, ID, P INDIRECT ADDRESS 2, P GOAL POSITION+2, &ErrorStatus); dx1 write word(Port, ID, P INDIRECT ADDRESS 3, P GOAL POSITION+3, &ErrorStatus); dxl_write_word(Port, ID, P_INDIRECT ADDRESS 4, P GOAL VELOCITY, &ErrorStatus): dxl_write_word(Port, ID, P_INDIRECT_ADDRESS_5, P_GOAL_VELOCITY+1, &ErrorStatus); dxl write word(Port, ID, P INDIRECT ADDRESS 6, P GOAL VELOCITY+2, &ErrorStatus); dxl_write_word(Port, ID, P_INDIRECT_ADDRESS_7, P_GOAL_VELOCITY+3, &ErrorStatus); dxl write word(Port, ID, P INDIRECT ADDRESS 8, P GOAL ACCELERATION, &ErrorStatus); dxl write word(Port, ID, P INDIRECT ADDRESS 9, P GOAL ACCELERATION+1, &ErrorStatus); dxl write word(Port, ID, P INDIRECT ADDRESS 10, P GOAL ACCELERATION+2, &ErrorStatus); dxl write word(Port, ID, P INDIRECT ADDRESS 11, P GOAL ACCELERATION+3, &ErrorStatus);

- The program's entire source code is shown below.

main.cpp			
#include	<stdio.h></stdio.h>		
#include	<conio.h></conio.h>		
#include	"dynamixel.h"		
#define	COM_PORT_NUM	17	//Comport Number of USB2DXL
#define	BAUD_RATE_NUM	3	//Baudrate Number of Dynamixel PRO
#define	P_INDIRECT_ADDRESS_0	49	//Address of 1st Indirect Address in Control
Table			
#define	P_INDIRECT_ADDRESS_1	51	//Address of 2nd Indirect Address in Control
Table			
#define	P INDIRECT ADDRESS 2	53	//Address of 3rd Indirect Address in Control
Table			
#define	P INDIRECT ADDRESS 3	55	//Address of 4th Indirect Address in Control
Table			
#define	P INDIRECT ADDRESS 4	57	//Address of 5th Indirect Address in Control

Table #define	P_INDIRECT_ADDRESS_5	59	//Address of 6th Indirect Address in Control			
Table #define	P_INDIRECT_ADDRESS_6	61	//Address of 7th Indirect Address in Control			
Table #define	P_INDIRECT_ADDRESS_7	63	//Address of 8th Indirect Address in Control			
Table #define	P_INDIRECT_ADDRESS_8	65	//Address of 9th Indirect Address in Control			
Table #define	P_INDIRECT_ADDRESS_9	67	//Address of 10th Indirect Address in Control			
Table #define	P_INDIRECT_ADDRESS_10	69	//Address of 11th Indirect Address in Control			
Table #define Table	P_INDIRECT_ADDRESS_11	71	//Address of 12th Indirect Address in Control			
#define #define #define #define	P_TORQUE_ENABLE P_GOAL_POSITION P_GOAL_VELOCITY P_GOAL_ACCELERATION	562 596 600 606	 //Address of Torque Enable in Control Table //Address of Goal Position in Control Table //Address of Goal Velocity in Control Table //Address of Goal Acceleration in Control 			
Table #define	P_INDIRECT_DATA_0	634	//Address of Goal Indirect Data in Control Table			
#define I	D	1 //	/ID of Dynamixel PRO you use			
// Print exvoid Print	rror bit of status packet ntErrorCode(int ErrorCode)					
i i	if(ErrorCode & ERRBIT_VOLT printf("Input voltage erro	AGE) or!\n");				
i	f(ErrorCode & ERRBIT_ANGL printf("Angle limit error	Æ) !\n");				
i	if(ErrorCode & ERRBIT_OVER printf("Overheat error!\n	HEAT) ");				
i	<pre>if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n");</pre>					
i	if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n");					
i	if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n");					
i }	if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n");					
,						

```
int main(void)
{
       SerialPort sp = \{0, 0, 0, 0, 0\};
       SerialPort *Port = &sp;
       //Open the port of USB2DXL
       if(dxl initialize(Port, COM PORT NUM, BAUD RATE NUM) == COMM RXSUCCESS)
               printf("Succeed to open USB2Dynamixel!\n");
       else
        ł
               printf( "Failed to open USB2Dynamixel!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
        }
       int Result, ErrorStatus;
       //Torque Off
       printf( "Torque Off...\n" );
       Result = dxl write byte(Port, ID, P TORQUE ENABLE, 0, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
        {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
        }
       //Set the Indirect Address for Goal Position
       printf("Press any key to set the indirect address for goal position...\n");
        getch();
       Result = dxl write word(Port, ID, P INDIRECT ADDRESS 0, P GOAL POSITION,
&ErrorStatus);
       Result = dx1 write word(Port, ID, P INDIRECT ADDRESS 1, P GOAL POSITION+1,
&ErrorStatus);
       Result = dxl write word(Port, ID, P INDIRECT ADDRESS 2, P GOAL POSITION+2,
&ErrorStatus):
       Result = dxl write word(Port, ID, P INDIRECT ADDRESS 3, P GOAL POSITION+3,
&ErrorStatus);
       if( Result != COMM RXSUCCESS )
        {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
       else
        {
               if (ErrorStatus != 0)
```

```
PrintErrorCode(ErrorStatus);
               else
                      printf("Succeed to set the indirect address for goal position!\n");
       }
       //Set the Indirect Address for Goal Velocity
       printf("Press any key to set the indirect address for goal velocity...\n");
        getch():
       Result = dxl write word(Port, ID, P INDIRECT ADDRESS 4, P GOAL VELOCITY,
&ErrorStatus);
       Result = dxl write word(Port, ID, P INDIRECT ADDRESS 5, P GOAL VELOCITY+1,
&ErrorStatus);
       Result = dxl write word(Port, ID, P INDIRECT ADDRESS 6, P GOAL VELOCITY+2,
&ErrorStatus);
       Result = dxl write word(Port, ID, P INDIRECT ADDRESS 7, P GOAL VELOCITY+3,
&ErrorStatus);
       if( Result != COMM RXSUCCESS )
       {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
        }
       else
        Ş
               if(ErrorStatus != 0)
                      PrintErrorCode(ErrorStatus);
               else
                      printf("Succeed to set the indirect address for goal velocity!\n");
       }
       //Set the Indirect Address for Goal Acceleration
       printf("Press any key to set the indirect address for goal acceleration...\n");
        getch();
       Result = dxl write word(Port, ID, P INDIRECT ADDRESS 8,
P GOAL ACCELERATION,
                               &ErrorStatus);
       Result = dxl write word(Port, ID, P INDIRECT ADDRESS 9,
P GOAL ACCELERATION+1, & ErrorStatus);
       Result = dxl write word(Port, ID, P INDIRECT ADDRESS 10,
P_GOAL_ACCELERATION+2, & ErrorStatus):
       Result = dxl write word(Port, ID, P INDIRECT ADDRESS 11,
P GOAL ACCELERATION+3, & ErrorStatus);
       if( Result != COMM RXSUCCESS )
       {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
       else
```

```
{
       if(ErrorStatus != 0)
              PrintErrorCode(ErrorStatus);
       else
               printf("Succeed to set the indirect address for goal acceleration!\n");
}
//Torque On
printf( "Torque On...\n" );
Result = dx1 write byte(Port, ID, P TORQUE ENABLE, 1, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
       printf( "Failed to write!\n" );
       printf( "Press any key to terminate...\n" );
       _getch():
       return 0;
}
int position, velocity, acceleration;
position = 100000;
velocity = 10000;
acceleration = 16;
//Make a tx data
unsigned char data[12];
data[0] = DXL LOBYTE(DXL LOWORD(position));
        = DXL HIBYTE(DXL LOWORD(position));
data[1]
data[2]
        = DXL LOBYTE(DXL HIWORD(position));
data[3] = DXL HIBYTE(DXL HIWORD(position));
data[4] = DXL LOBYTE(DXL LOWORD(velocity));
data[5] = DXL HIBYTE(DXL LOWORD(velocity));
data[6] = DXL LOBYTE(DXL HIWORD(velocity));
       = DXL HIBYTE(DXL HIWORD(velocity));
data[7]
data[8] = DXL LOBYTE(DXL LOWORD(acceleration));
data[9] = DXL HIBYTE(DXL LOWORD(acceleration));
data[10] = DXL LOBYTE(DXL HIWORD(acceleration));
data[11] = DXL HIBYTE(DXL HIWORD(acceleration));
//change the position value, moving speed, acceleration using indirect addr
printf( "Press any key to change the position, speed, acceleration...\n");
getch():
Result = dxl write(Port, ID, P INDIRECT DATA 0, 12, data, &ErrorStatus);
if( Result != COMM RXSUCCESS )
       printf( "Failed to write!\n" );
       printf( "Press any key to terminate...\n" );
       getch();
       return 0;
```

```
else
{
       if (ErrorStatus != 0)
              PrintErrorCode(ErrorStatus);
}
position = -100000;
velocity =
             1000:
acceleration = 2:
//Make a tx data
data[0] = DXL LOBYTE(DXL LOWORD(position));
data[1] = DXL HIBYTE(DXL LOWORD(position));
data[2]
       = DXL LOBYTE(DXL HIWORD(position));
       = DXL HIBYTE(DXL HIWORD(position));
data[3]
       = DXL LOBYTE(DXL LOWORD(velocity));
data[4]
data[5] = DXL_HIBYTE(DXL_LOWORD(velocity));
data[6] = DXL LOBYTE(DXL HIWORD(velocity));
data[7] = DXL HIBYTE(DXL HIWORD(velocity));
data[8] = DXL LOBYTE(DXL LOWORD(acceleration));
data[9] = DXL HIBYTE(DXL LOWORD(acceleration));
data[10] = DXL LOBYTE(DXL HIWORD(acceleration));
data[11] = DXL HIBYTE(DXL HIWORD(acceleration));
//change the position value, moving speed, acceleration using indirect addr
printf( "Press any key to change the position, speed, acceleration...\n" );
getch():
Result = dxl write(Port, ID, P INDIRECT DATA 0, 12, data, &ErrorStatus);
if( Result != COMM RXSUCCESS )
{
       printf( "Failed to write!\n" );
       printf( "Press any key to terminate...\n" );
       getch();
       return 0;
}
else
{
       if(ErrorStatus != 0)
              PrintErrorCode(ErrorStatus);
}
//Close the port of USB2DXL
printf( "Press any key to terminate...\n" );
getch();
dx1 terminate(Port);
return 0;
```

- ii. Reading the temperature and the current position using Indirect Address
 - Implement Indirect Address to obtain read outputsl.
 - Read the Present Position and Present Temperature by using Indirect Address.
 - The program's entire source code is shown below.

main.cpp	main.cpp				
#include	<stdio.h></stdio.h>				
#include	<conio.h></conio.h>				
#include	"dynamixel.h"				
		1.5			
#define	COM_PORT_NUM	17	//Comport Number of USB2DXL		
#define	BAUD_RAIE_NUM	3	//Baudrate Number of Dynamixel PRO		
#dofino	D ODEDATING MODE	11	//Address of Operating Mode in Central Table		
#define	P_OPERATING_MODE	11	//Address of Operating Mode in Control Table		
#define	P_INDIREC1_ADDRES5_0	49	//Address of 1st maneet Address in Control		
Hdofino	D INDIDECT ADDRESS 1	51	//Address of 2nd Indirect Address in Control		
Table	I_INDIRECI_ADDRESS_I	51	//Address of 2nd marreet Address in Control		
#define	P INDIRECT ADDRESS 2	53	//Address of 3rd Indirect Address in Control		
Table		55	marces of sta marcet radiess in control		
#define	P INDIRECT ADDRESS 3	55	//Address of 4th Indirect Address in Control		
Table		00			
#define	P INDIRECT ADDRESS 4	57	//Address of 5th Indirect Address in Control		
Table		0 /			
#define	P TORQUE ENABLE	562	//Address of Torque Enable in Control Table		
#define	P PRESENT POSITION	611	//Address of Present Position in Control Table		
#define	P PRESENT TEMPERATUR	E 625	//Address of Present Temperature in Control		
Table			1		
#define	P_INDIRECT_DATA_0		634 //Address of Goal Indirect Data in		
Control 7	Гable				
#define I	D	1	//ID of Dynamixel PRO you use		
// Print e	rror bit of status packet				
void Prir	ntErrorCode(int ErrorCode)				
{					
]	f(ErrorCode & ERRBIT_VOLT	AGE)			
	printf("Input voltage err	or!\n");			
	Contractor of CODDIT ANCI	E)			
II(EITOICOde & EKKBII_ANGLE)					
printi (Angie initi error! \n);					
if(ErrorCode & EDDDIT OVEDHEAT)					
	n(Enorous & ENDIT_OVENIEAT)				
	printi overheat erfort a	- ,,			
i i	f(ErrorCode & ERRBIT RANC	ΞE)			
	printf("Out of range error	or!\n"):			
	1				

```
if(ErrorCode & ERRBIT CHECKSUM)
               printf("Checksum error!\n");
       if(ErrorCode & ERRBIT OVERLOAD)
               printf("Overload error!\n");
       if(ErrorCode & ERRBIT INSTRUCTION)
               printf("Instruction code error!\n");
}
int main(void)
{
       SerialPort sp = \{0, 0, 0, 0, 0\};
       SerialPort *Port = &sp;
       //Open the port of USB2DXL
       if(dxl initialize(Port, COM PORT NUM, BAUD RATE NUM) ==
COMM RXSUCCESS)
               printf("Succeed to open USB2Dynamixel!\n");
       else
        {
               printf( "Failed to open USB2Dynamixel!\n" );
               printf( "Press any key to terminate...\n" );
               _getch();
               return 0;
        }
       int Result, ErrorStatus;
       //Torque Off
       printf( "Torque Off...\n" );
       Result = dxl write byte(Port, ID, P TORQUE ENABLE, 0, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
        {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               _getch();
               return 0;
        }
       //change the operating mode to wheel mode
       printf( "Make the Mode of Dynamixel PRO to Wheel Mode...\n" );
       Result = dx1 write byte(Port, ID, P OPERATING MODE, 1, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
        ł
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
```

```
//Set the Indirect Address for Present Temperature
       printf("Press any key to set the indirect address for temperature...\n");
       getch():
       Result = dxl write word(Port, ID, P INDIRECT ADDRESS 0,
P PRESENT TEMPERATURE, & ErrorStatus);
       if( Result != COMM RXSUCCESS )
        {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
        }
       else
        ł
               if (ErrorStatus != 0)
                       PrintErrorCode(ErrorStatus);
               else
                       printf("Succeed to set the indirect address for temperature!\n");
        }
       //Set the Indirect Address for Goal Velocity
       printf("Press any key to set the indirect address for present position...\n");
        getch();
       Result = dxl write word(Port, ID, P INDIRECT ADDRESS 1, P PRESENT POSITION,
&ErrorStatus);
       Result = dxl write word(Port, ID, P INDIRECT ADDRESS 2,
P PRESENT POSITION+1, & ErrorStatus);
       Result = dxl write word(Port, ID, P INDIRECT_ADDRESS_3,
P PRESENT POSITION+2, & ErrorStatus);
       Result = dxl write word(Port, ID, P INDIRECT ADDRESS 4,
P PRESENT POSITION+3, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
        {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
        }
       else
               if (ErrorStatus != 0)
                       PrintErrorCode(ErrorStatus);
               else
                       printf("Succeed to set the indirect address for present position!\n");
        }
       //Torque On
       printf( "Torque On...\n" );
```

```
Result = dxl write byte(Port, ID, P TORQUE ENABLE, 1, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
        {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" ):
                getch();
               return 0;
        }
       //Rotating Start
       printf("Rotating Start\n");
       Result = dxl write dword(Port, ID, 600, 5000, &ErrorStatus);
       if( Result != COMM RXSUCCESS )
        {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
        }
       printf("Press any key to terminate...\n");
       printf("\n");
       while(true)
        {
               if( kbhit())
                       break;
               unsigned char data[5];
               dxl read(Port, ID, P INDIRECT DATA 0, 5, data, &ErrorStatus);
               int temp, present position;
               temp = data[0];
               present position = DXL MAKEDWORD( DXL MAKEWORD(data[1], data[2]),
DXL MAKEWORD(data[3], data[4]));
               printf("\r");
               printf("present temperature : %d, presen position : %d", temp, present position);
       printf("\n");
       //Close the port of USB2DXL
       dxl terminate(Port);
       return 0;
```

1.2.5 Using Multiple Dynamixel PROs

- i. Controlling the LEDs of 3 Dynamixel PROs
 - Implement dxl_synch_write to simultaneously send a command to multiple Dynamixel PROs.
 - The program's entire source code is shown below.

main.cpp)				
#include	<stdio.h></stdio.h>				
#include	<conio.h></conio.h>				
#include	"dynamixel.h"				
#define #define	COM_PORT_NUM BAUD_RATE_NUM	17 3	//Comport Number of USB2DXL //Baudrate Number of Dynamixel PRO		
#define	P_LED_RED	563	//Address of LED RED in Control Table		
#define I	D 1	1 /	/ID of Dynamixel PRO you use		
#define I	D_2	2	5		
#define I	D_3	3			
// Print en void Prin	- rror bit of status packet ntErrorCode(int ErrorCode)				
i	f(ErrorCode & ERRBIT_VOL printf("Input voltage er	ΓAGE) ror!\n");			
i	f(ErrorCode & ERRBIT_ANG printf("Angle limit erro	LE) r!\n");			
i	if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n");				
i	f(ErrorCode & ERRBIT_RAN printf("Out of range err	GE) or!\n");			
i	if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n");				
i	if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n");				
i }	<pre>if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n");</pre>				
int main((void)				
	{ SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp				

```
//Open the port of USB2DXL
       if(dxl initialize(Port, COM PORT NUM, BAUD RATE NUM) ==
COMM RXSUCCESS)
               printf("Succeed to open USB2Dynamixel!\n");
       else
        ł
               printf( "Failed to open USB2Dynamixel!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
        }
       int Result;
       unsigned char param[6];
       param[0] = ID 1;
       param[1] = 255;
       param[2] = ID 2;
       param[3] = 255;
       param[4] = ID 3;
       param[5] = 255;
       //LED On
       printf( "Press any to turn on the LED...\n" );
        getch();
       Result = dxl sync write(Port, P_LED_RED, 1, param, 6);
       if( Result != COMM RXSUCCESS )
        {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
               getch();
               return 0;
        }
       param[0] = ID 1;
       param[1] = 0;
       param[2] = ID 2;
       param[3] = 0;
       param[4] = ID 3;
       param[5] = 0;
       //LED Off
       printf( "Press any to turn off the LED...\n" );
        getch();
       Result = dxl sync write(Port, P LED RED, 1, param, 6);
       if( Result != COMM RXSUCCESS )
        {
               printf( "Failed to write!\n" );
               printf( "Press any key to terminate...\n" );
                getch();
```

```
return 0;
}
//Close the port of USB2DXL
printf( "Press any key to terminate...\n" );
_getch();
dxl_terminate(Port);
return 0;
```

-

ii.	Controlling the	Goal Position	of 3 Dynamixel	PRO
-----	-----------------	----------------------	----------------	-----

The program's entire source code is shown below.

<pre>#include <stdio.h> #include <stdio.h> #include 'qnamixel.h" #define COM_PORT_NUM 17 //Comport Number of USB2DXL #define BAUD_RATE_NUM 3 //Baudrate Number of Dynamixel PRO #define P_TORQUE_ENABLE 562 //Address of Torque Enable in Control Table #define ID_1 1 1 //ID of Dynamixel PRO you use #define ID_2 2 2 #define ID_3 3 3 // Print error bit of status packet void PrintErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Overhead error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overlead error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overlead error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); if(ErrorCode & ERBIT_INSTRUCTION) printf("Instruction code error!\n"); }</stdio.h></stdio.h></pre>	main.cpp)					
<pre>#include 'conio.h> #include 'dynamixel.h" #define COM PORT_NUM 17 //Comport Number of USB2DXL #define BAUD_RATE_NUM 3 //Baudrate Number of Dynamixel PRO #define P_GOAL_POSITION 596 //Address of Goal Position in Control Table #define ID_1 1 //ID of Dynamixel PRO you use #define ID_2 2 #define ID_3 3 // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Code & ERRBIT_OVERHEAT) printf("Code & ERRBIT_OVERLOAD) printf("Code & ERRBIT_OVERLOAD) printf("Code & ERRBIT_OVERLOAD) printf("Instruction code error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Dverheat error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Dverheat error!\n"); if(ErrorCode & ERBIT_INSTRUCTION) printf("Dverheat error!\n"); } }</pre>	#include	<stdio.h></stdio.h>					
<pre>#include "dynamixel.h" #define COM_PORT_NUM 17 //Comport Number of USB2DXL #define BAUD_RATE_NUM 3 //Baudrate Number of Dynamixel PRO #define P_TORQUE_ENABLE 562 //Address of Torque Enable in Control Table #define ID_1 1 1 //ID of Dynamixel PRO you use #define ID_2 2 2 #define ID_3 3 // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_CANGE) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_COVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_COVERLOAD) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Instruction code error!\n"); if(ErrorCode & ERBIT_OVERLOAD) printf("In</pre>	#include	<conio.h></conio.h>					
<pre>#define COM_PORT_NUM 17 //Comport Number of USB2DXL #define BAUD_RATE_NUM 3 //Baudrate Number of Dynamixel PRO #define P_COAL_POSITION 596 //Address of Goal Position in Control Table #define ID_1 1 //ID of Dynamixel PRO you use #define ID_2 2 2 #define ID_3 3 // Print error bit of status packet void PrintErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); if(ErrorCode & ERRBIT_NSTRUCTION) printf("Instruction code error!\n"); </pre>	#include	"dynamixel.h"					
<pre>#define COM_PORT_NUM 17 //Comport Number of USB2DXL #define BAUD_RATE_NUM 3 //Baudrate Number of Dynamixel PRO #define P_TORQUE_ENABLE 562 //Address of Torque Enable in Control Table #define D_1 1 1 //ID of Dynamixel PRO you use #define ID_2 2 2 #define ID_3 3 3 // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_CVERHEAT) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { //Open the port of USB2DXL if(#instidiendent error UN POIT_NUM PAUD PATE_NUM) == //Open the port of USB2DXL if(#instidiendent error UN POIT_NUM PAUD PATE_NUM) == //Open the port of USB2DXL if(#instidiendent error UN POIT_NUM PAUD PATE_NUM) == //Open the port of USB2DXL if(#instidiendent error UN POIT_NUM PAUD PATE_NUM) == //Open the port of USB2DXL if(#instidiendent error UN POIT_NUM PAUD PATE_NUM)= //Open the port of USB2DXL</pre>							
<pre>#define BAUD_RATE_NUM 3 //Baudrate Number of Dynamixel PRO #define P_TORQUE_ENABLE 562 //Address of Torque Enable in Control Table #define ID_1 1 1 //ID of Dynamixel PRO you use #define ID_2 2 2 #define ID_3 3 // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Cotecksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Cotecksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Cotecksum error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { //Open the port of USB2DXL if(Merting Port = &sp //Open the port of USB2DXL</pre>	#define	COM_PORT_NUM	17	//Comport Number of USB2DXL			
<pre>#define P_TORQUE_ENABLE 562 //Address of Torque Enable in Control Table #define P_GOAL_POSITION 596 //Address of Goal Position in Control Table #define ID_1 1 //ID of Dynamixel PRO you use #define ID_3 3 // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_RANGE) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Instruction code error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } </pre>	#define	BAUD_RATE_NUM	3	//Baudrate Number of Dynamixel PRO			
<pre>#define P_TORQUE_ENABLE 562 //Address of Torque Enable in Control Table #define P_GOAL_POSITION 596 //Address of Goal Position in Control Table #define ID_1 1 //ID of Dynamixel PRO you use #define ID_2 2 #define ID_3 3 // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_RANGE) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf(INSTRUCTION) printf(INSTRUCTION) printf(INSTRUCTION) pri</pre>							
<pre>#define P_GOAL_POSITION 596 //Address of Goal Position in Control Table #define ID_1 1 1 //ID of Dynamixel PRO you use #define ID_2 2 2 #define ID_3 3 // Print error bit of status packet void PrintErrorCode (int ErrorCode) { if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Instruction code error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf(INSTRUCTION) printf(INSTRUCTION) printf(INSTRUCTION) printf(INSTRUCTION) printf(INSTRUCTION) printf(INSTRUCTION) printf(INSTRUCTION) printf(INSTRUCTION) printf(INSTRUCTION) printf(INST</pre>	#define	P_TORQUE_ENABLE	562	//Address of Torque Enable in Control Table			
<pre>#define ID_1 1 //ID of Dynamixel PRO you use #define ID_2 2 #define ID_3 3 // Print error bit of status packet void PrintErrorCode (int ErrorCode) { if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_RANGE) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Instruction code error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); if(ErrorCode & ERRBIT_NOTERLOAD) printf("Instruction code error!\n"); } </pre>	#define	P_GOAL_POSITION	596	//Address of Goal Position in Control Table			
<pre>#define ID_1 1 //ID of Dynamixel PRO you use #define ID_2 2 #define ID_3 3 // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overhead error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overhead error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Instruction code error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); }</pre>							
<pre>#define ID_2 2 2 #define ID_3 3 // Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_CARAGE) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Instruction code error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp i/(Open the port of USB2DXL iif(del initioningChert COMPORT_NUM_BAUD_BATE_NUM) ==</pre>	#define I	D_1	1	//ID of Dynamixel PRO you use			
<pre>#define ID_3 3 // Print error bit of status packet void PrintErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } </pre>	#define I	D_2	2				
<pre>// Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_ANGLE) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(del_initiate@port_COM_ROPET_NUM_RAUD_RATE_NUM) ==</pre>	#define I	D_3	3				
<pre>// Print error bit of status packet void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_ANGLE) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = & sp; //Open the port of USB2DXL if(deL injuticagent, COM_ROBET_NUM_RAUD_RATE_NUM) ==</pre>							
<pre>void PrintErrorCode(int ErrorCode) { if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(det) initialize(Paet COM_ POPT_NUM_PAUD_PATE_NUM) == </pre>	// Print e	rror bit of status packet					
<pre>{ if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(but initialize(Port COM_PORT_NUM_PAUD_PATE_NUM) == </pre>	void Prin	ntErrorCode(int ErrorCode)					
<pre>if(ErrorCode & ERRBIT_VOLTAGE) printf("Input voltage error!\n"); if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(del initialize(Port COM POPET NUM PATE NUM) == </pre>	{						
<pre>printf("Input voltage error\\n"); if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error\\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort sp = {0,0,0,0,0}; SerialPort sp = {0,0,0,0,0}; SerialPort sp = {0,0,0,0,0}; </pre>	i	f(ErrorCode & ERRBIT_VOL	TAGE)				
<pre>if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort sp = {0,0,0,0,0}; SerialPort sp = {0,0,0,0,0}; SerialPort sp = {0,0,0,0,0}; </pre>		printf("Input voltage en	ror!(n'');				
<pre>if(ErrorCode & ERRBIT_ANGLE) printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort sp = {0,0,0,0,0}; SerialPort sp = {0,0,0,0,0}; SerialPort sp = {0,0,0,0,0}; </pre>							
<pre>printf("Angle limit error!\n"); if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort sp = {0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(del initialize(Port_COM_BORT_NUM_BAUD_BATE_NUM) ==</pre>	i	f(ErrorCode & ERRBIT_ANC	iLE)				
<pre>if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort sp = {0,0,0,0,0,0}; SerialPort sp = {0,0,0,0,0}; SerialPort sp = {0,0,0,0,0,0}; SerialPort sp = {0,0,0,0,0,0}; SerialPort sp = {0,0,0,0,0}; SerialPort sp = {0,0,0,0,0,0}; SerialPort sp = {0,0,0,0,0}; SerialPort sp = {0,0,0,0,0,0}; SerialPort sp = {0,0,0,0,0,0}; SerialPort sp = {0,0,0,0,0,0}; SerialPort sp = {0,0,0,0,0,0,0}; SerialPort sp = {0,0,0,0,0,0,0,0,0,0,0,0}; SerialPort sp = {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,</pre>		printf("Angle limit erro	or!\n");				
<pre>if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n"); if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(dxl_initialize(Bort_COM_PORT_NUM_PATE_NUM) == </pre>							
<pre>printf("Overheat error!\n"); if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(del initialize(Port_COM_POPET_NUM_RAUD_RATE_NUM) == </pre>	i	if(ErrorCode & ERRBIT_OVERHEAT)					
<pre>if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(del initialize(Port COM_ROPET_NUM_DATE_NUM) == </pre>		printf("Overheat error!	\ n");				
<pre>if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(det_initialize(Part_COM_PORT_NUM_PATE_NUM) == </pre>							
<pre>printf("Out of range error!\n"); if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(dxl_initialize(Port_COM_PORT_NUM_PAUD_PATE_NUM) == </pre>	i	f(ErrorCode & ERRBIT_RAN	(GE)				
<pre>if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(dxl_initialize(Part_COM_PORT_NUM_PAUD_PATE_NUM) == </pre>		printf("Out of range error!\n");					
<pre>if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(del_initialize(Bort_COM_ROPET_NUM_RAUD_RATE_NUM) == </pre>							
<pre>printf("Checksum error!\n"); if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(ded_initialize(Port_COM_POPT_NUM_PAUD_PATE_NUM) == </pre>	i	f(ErrorCode & ERRBIT_CHE	CKSUM				
<pre>if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(dxL initialize(Port_COM_PORT_NUM_PAUD_PATE_NUM) ==</pre>		printf("Checksum error	r!\n");				
<pre>if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(del_initialize(Port_COM_POPT_NUM_PATE_NUM) == </pre>							
<pre>printf("Overload error!\n"); if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(dxl_initialize(Port_COM_POPT_NUM_PAUD_PATE_NUM) ==</pre>	i	f(ErrorCode & ERRBIT_OVE	RLOAD)				
<pre>if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(dx1_initialize(Port_COM_PORT_NUM_PAUD_PATE_NUM) ==</pre>		printf("Overload error!	\n") ;				
<pre>if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(dxl_initialize(Port_COM_POPT_NUM_PAUD_PATE_NUM) ==</pre>							
<pre>printf("Instruction code error!\n"); } int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(dxl_initialize(Port_COM_POPT_NUM_PAUD_PATE_NUM) ==</pre>	i	f(ErrorCode & ERRBIT INST	RUCTIC	DN)			
<pre>} int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(dx1_initialize(Port_COM_POPT_NUM_PAUD_PATE_NUM) ==</pre>		printf("Instruction code	e error!\n	");			
<pre>int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(dx1_initialize(Port_COM_POPT_NUM_PAUD_PATE_NUM) ==</pre>	}	-					
<pre>int main(void) { SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(dx1_initialize(Port_COM_POPT_NUM_PAUD_PATE_NUM) ==</pre>							
<pre>{ SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(dx1_initialize(Port_COM_POPT_NUM_PAUD_PATE_NUM) == </pre>	int main((void)					
SerialPort sp = {0,0,0,0,0}; SerialPort *Port = &sp //Open the port of USB2DXL if(dx1_initialize(Port_COM_POPT_NUM_PAUD_PATE_NUM) ==							
SerialPort *Port = &sp //Open the port of USB2DXL if(dx1_initialize(Port_COM_POPT_NUM_PAUD_PATE_NUM) ==		SerialPort sp = $\{0,0,0,0,0\};$					
//Open the port of USB2DXL		SerialPort *Port = &sp					
//Open the port of USB2DXL		• *					
if dyl initializa (Port COM DOPT NUM DAUD DATE NUM)	/	//Open the port of USB2DXL					
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII	i	f(dxl initialize(Port, COM PC	DRT NUI	M, BAUD RATE NUM) ==			

```
COMM RXSUCCESS)
              printf("Succeed to open USB2Dynamixel!\n");
       else
       {
              printf( "Failed to open USB2Dynamixel!\n" );
              printf( "Press any key to terminate...\n" );
              _getch():
              return 0;
       }
       int Result;
       unsigned char param[15];
       //Torque ON
       printf("Torque On\n");
       param[0] = ID 1;
       param[1] = 1;
       param[2] = ID 2;
       param[3] = 1;
       param[4] = ID 3;
       param[5] = 1;
       Result = dxl sync write(Port, P TORQUE ENABLE, 1, param, 6);
       if( Result != COMM RXSUCCESS )
       {
              printf( "Failed to write!\n" );
              printf( "Press any key to terminate...\n" );
              getch();
              return 0;
       }
       param[0] = ID 1;
       param[1] = DXL LOBYTE(DXL LOWORD(100000));
       param[2] = DXL HIBYTE(DXL LOWORD(100000));
       param[3] = DXL LOBYTE(DXL HIWORD(100000));
       param[4] = DXL HIBYTE(DXL HIWORD(100000));
       param[5] = ID 2;
       param[6] = DXL LOBYTE(DXL LOWORD(50000));
       param[7] = DXL HIBYTE(DXL LOWORD(50000));
       param[8] = DXL LOBYTE(DXL HIWORD(50000));
       param[9] = DXL HIBYTE(DXL HIWORD(50000));
       param[10] = ID 3;
       param[11]= DXL LOBYTE(DXL LOWORD(-80000));
       param[12]= DXL HIBYTE(DXL LOWORD(-80000));
       param[13]= DXL LOBYTE(DXL HIWORD(-80000));
       param[14]= DXL HIBYTE(DXL HIWORD(-80000));
       printf( "Press any to change goal position...\n" );
       getch();
       Result = dxl sync write(Port, P GOAL POSITION, 4, param, 15);
       if( Result != COMM RXSUCCESS )
```

```
{
       printf( "Failed to write!\n" );
       printf( "Press any key to terminate...\n" );
       getch();
       return 0;
}
param[0] = ID 1;
param[1] = DXL LOBYTE(DXL LOWORD(0));
param[2] = DXL HIBYTE(DXL LOWORD(0));
param[3] = DXL LOBYTE(DXL HIWORD(0));
param[4] = DXL HIBYTE(DXL HIWORD(0));
param[5] = ID 2;
param[6] = DXL LOBYTE(DXL LOWORD(0));
param[7] = DXL HIBYTE(DXL LOWORD(0));
param[8] = DXL LOBYTE(DXL HIWORD(0));
param[9] = DXL HIBYTE(DXL HIWORD(0));
param[10] = ID 3;
param[11]= DXL LOBYTE(DXL LOWORD(0));
param[12]= DXL HIBYTE(DXL LOWORD(0));
param[13]= DXL LOBYTE(DXL HIWORD(0));
param[14]= DXL HIBYTE(DXL HIWORD(0));
printf( "Press any to change goal position...\n" );
getch();
Result = dxl sync write(Port, P GOAL POSITION, 4, param, 15);
if( Result != COMM RXSUCCESS )
{
       printf( "Failed to write!\n" );
       printf( "Press any key to terminate...\n" );
       getch();
       return 0;
}
//Close the port of USB2DXL
printf( "Press any key to terminate...\n" );
_getch():
dxl terminate(Port);
return 0;
```

- iii. Reading the the Current Position of 3 Dynamixel PROs
 - Implement **dxl_bulk_read** to simultaneously read the values of multiple Dynamixel PROs.
 - The program's entire source code is shown below.

main.cpp)					
#include	<stdio.h></stdio.h>					
#include	<conio.h></conio.h>					
#include	"dynamixel.h"					
#define	COM_PORT_NUM	17	//Comport Number of USB2DXL			
#define	BAUD_KATE_NUM	3	//Baudrate Number of Dynamixel PRO			
#define	P_PRESENT_POSITION	611	//Address of Goal Position in Control Table			
#define I	D 1	1 //	ID of Dynamixel PRO you use			
#define I	D_2	2				
#define I	D_3	3				
// Print ex void Prin	rror bit of status packet ntErrorCode(int ErrorCode)					
{ i	f(ErrorCode & ERRBIT_VOL printf("Input voltage e	LTAGE) rror!\n");				
i	if(ErrorCode & ERRBIT_ANC printf("Angle limit erro	GLE) or!\n");				
i	if(ErrorCode & ERRBIT_OVERHEAT) printf("Overheat error!\n");					
i	if(ErrorCode & ERRBIT_RANGE) printf("Out of range error!\n");					
i	if(ErrorCode & ERRBIT_CHECKSUM) printf("Checksum error!\n");					
i	if(ErrorCode & ERRBIT_OVERLOAD) printf("Overload error!\n");					
i	<pre>if(ErrorCode & ERRBIT_INSTRUCTION) printf("Instruction code error!\n");</pre>					
}	}					
int main(int main(void)					
	SerialPort sp = $\{0,0,0,0,0\}$; SerialPort *Port = &sp					
/	//Open the port of USB2DXL					

```
if(dxl initialize(Port, COM PORT NUM, BAUD RATE NUM) ==
COMM RXSUCCESS)
              printf("Succeed to open USB2Dynamixel!\n");
       else
       {
              printf( "Failed to open USB2Dynamixel!\n" );
              printf( "Press any key to terminate...\n" );
              getch();
              return 0;
       }
       int Result;
       int position length=4;
       unsigned char param[15];
       param[0] = ID 1;
       param[1] = DXL LOBYTE(P PRESENT_POSITION);
       param[2] = DXL HIBYTE(P PRESENT POSITION);
       param[3] = DXL LOBYTE(position length);
       param[4] = DXL HIBYTE(position length);
       param[5] = ID 2;
       param[6] = DXL LOBYTE(P PRESENT POSITION);
       param[7] = DXL HIBYTE(P PRESENT POSITION);
       param[8] = DXL LOBYTE(position length);
       param[9] = DXL HIBYTE(position length);
       param[10] = ID 3;
       param[11] = DXL LOBYTE(P PRESENT POSITION);
       param[12] = DXL HIBYTE(P PRESENT POSITION);
       param[13] = DXL LOBYTE(position length);
       param[14] = DXL HIBYTE(position length);
       BulkData bd[256];
       BulkData *pbd[256];
       for(unsigned int i = 0; i < 256; i++)
       {
              pbd[i] = \&bd[i];
       }
       dxl bulk read(Port, param, 15, pbd);
       int present position1;
       int present position2;
       int present position3;
       dxl get bulk dword(pbd, ID 1, P PRESENT POSITION, (unsigned*)&present position1);
       dxl get bulk dword(pbd, ID 2, P PRESENT POSITION, (unsigned*)&present position2);
       dxl get bulk dword(pbd, ID 3, P PRESENT POSITION, (unsigned*)&present position3);
```

printf("Present Position\n");

printf("ID_1 : %d\n", present_position1); printf("ID_2 : %d\n", present_position2); printf("ID_3 : %d\n", present_position3); //Close the port of USB2DXL printf("Press any key to terminate...\n"); _getch(); dx1_terminate(Port); return 0;

- iv. Read temperature of the first, position of the second, and present current of the third Dynamixel PRO
 - Implement dxl_bulk_read to simultaneously read the values of multiple Dynamixel PROs.
 - The program's entire source code is shown below.

main.cpp					
#include	<stdio.h></stdio.h>				
#include	<conto.h></conto.h>				
#Include	dynamixel.n				
#define	COM PORT NUM	17	//Comport Number of USB2DXL		
#define	BAUD_RATE_NUM	3	//Baudrate Number of Dynamixel PRO		
#define	P_PRESENT_POSITION	611	//Address of Goal Position in Control Table		
#define	P_PRESENT_CURRENT	621	//Address of Present Current in Control Table		
#define	P_PRESENI_IEMPERATU	RE 623	//Address of Present Temperature in Control		
Table					
#define I	D 1	1	//ID of Dynamixel PRO you use		
#define I	D_2	2			
#define I	D_3	3			
	1				
// Print en	rror bit of status packet				
vola Prin	defrorCode(Int ErrorCode)				
i i	f(ErrorCode & ERRBIT_VOI	TAGE)			
-	printf("Input voltage en	$\operatorname{ror!}(n'');$			
		,,			
i	if(ErrorCode & ERRBIT_ANGLE)				
printf("Angle limit error!\n");					
;	(ErrorCode & EDDDIT OVE				
II(EITOTCOde & EKKBI1_UVEKHEAI)					
	printi(overheat error:	" ,			
i	f(ErrorCode & ERRBIT RAN	(GE)			
	printf("Out of range er	ror!\n");			
1	f(ErrorCode & ERRBIT_CHE	CKSUM)		
printf("Checksum error!\n");					
i	if(ErrorCode & ERRBIT OVERLOAD)				
-	printf("Overload error!\n"):				
i	f(ErrorCode & ERRBIT_INST	RUCTIC	DN)		
	printf("Instruction code	e error!\n	");		
}					
int main((void)				
{					

```
SerialPort sp = \{0,0,0,0,0\};
       SerialPort *Port = &sp;
       //Open the port of USB2DXL
       if(dxl initialize(Port, COM PORT NUM, BAUD_RATE_NUM) ==
COMM RXSUCCESS)
              printf("Succeed to open USB2Dynamixel!\n");
       else
       {
              printf( "Failed to open USB2Dynamixel!\n" );
              printf( "Press any key to terminate...\n" );
              getch();
              return 0;
       }
       int Result:
       int current length = 2, position length = 4, temperature length = 1;
       unsigned char param[15];
       param[0] = ID 1;
       param[1] = DXL LOBYTE(P PRESENT CURRENT);
       param[2] = DXL HIBYTE(P PRESENT CURRENT);
       param[3] = DXL LOBYTE(current length);
       param[4] = DXL HIBYTE(current length);
       param[5] = ID 2;
       param[6] = DXL LOBYTE(P PRESENT POSITION);
       param[7] = DXL_HIBYTE(P PRESENT POSITION);
       param[8] = DXL LOBYTE(position length);
       param[9] = DXL HIBYTE(position length);
       param[10] = ID 3;
       param[11] = DXL LOBYTE(P PRESENT TEMPERATURE);
       param[12] = DXL HIBYTE(P PRESENT TEMPERATURE);
       param[13] = DXL LOBYTE(temperature length);
       param[14] = DXL HIBYTE(temperature length);
       BulkData bd[256];
       BulkData *pbd[256];
       for (unsigned int i = 0; i < 256; i++)
              pbd[i] = \&bd[i];
       dxl bulk read(Port, param, 15, pbd);
       int present current;
       int present position;
       int present temperature;
       dxl get bulk word(pbd, ID 1, P PRESENT CURRENT,
&present current);
       dxl get bulk dword(pbd, ID 2, P PRESENT POSITION,
(unsigned*)&present position);
```

```
dxl_get_bulk_byte( pbd, ID_3, P_PRESENT_TEMPERATURE,
&present_temperature);
    present_current = (short int) present_current
    printf("Present Position\n");
    printf("ID_1 : %d\n", present_current);
    printf("ID_2 : %d\n", present_position);
    printf("ID_3 : %d\n", present_temperature);
    //Close the port of USB2DXL
    printf( "Press any key to terminate...\n" );
    _getch();
    dxl_terminate(Port);
    return 0;
}
```
1.3 Linux

- 1.3.1 Set-up
 - i. Checking USB-to-Dynamixel connection in Linux
 - To the PC has detected the USB-to-Dynamixel device simply enter the command ls /dev
 - Linux OS will display a list onder the /dev directory simply look for "ttyUSB0"

root@robotis-li	.nux:/home/robotis# ls	/dev			
alarm	маррег	sda1	tty29	tty6	ttyS31
ashmem	mcelog	sda2	tty3	tty60	ttyS4
ati	mem	sda5	tty30	tty61	ttyS5
autofs	net	serial	tty31	tty62	ttyS6
binder	network_latency	sg0	tty32	tty63	ttyS7
olock	network_throughput	shm	tty33	tty7	ttyS8
osg	null	snapshot	tty34	tty8	ttvS9
otrfs-control	oldmem	snd	tty35	tty9	ttyUSB0
bus	port	stderr	tty36	ttyprintk	uinput
char	PPP	stdin	tty37	ttyS0	urandom
console	psaux	stdout	tty38	ttyS1	VCS
OFE	ntmy	ttv	ttv39	ttvS10	VCS1

- ii. Verify proper serial device connection
 - Note: there may be instances where the USB-to-Dynamixel dongle may show as "ttyUSB1" "instead of the default "ttyUSB0." This happens when another serial device has already been connected to the PC prior to connecting the USB-to-Dynamixel dongle.
 - To get Linux to reassign ttyUSB0 to USB-to-Dynamixel disconnect all serial devices; then connect USB-to-Dynamixel loffowed by inputting the command ls /dev. The device should appear as ttyUSB0. Afterwards, you may connect other serial devices.
- 1.3.2 Download and install Dynamixel SDK 2.0
 - i. Preparing your system in Linux (assuming the PC is connected to the internet)
 - Make sure to update your Linux OS package repositories with the command aptget update.
 - Dynamixel SDK 2.0 requires a C++ compiler (g++). To get it simply enter **apt-get install g++**.
 - A text editor to read and change the code, if desired. Fortunately, Linux releases are accompanied by editors, such as Vim and G-edit.
 - ii. Obtain the source
 - Visit support.robotis.com and go to the Dynamixel PRO section
 - Download Linux version of the zipped SDK file



- iii. Setup the source
 - The current source is a beta release. The next steps will ensure proper setup of the beta release.
 - [1] create a directory under root with the name DXLDSK_cpp_forLinux (mkdir /DXLSDK_cpp_forLinux);
 - [2] move or copy the downloaded zip file to the new directory (cp dxlsdk_cpp_forlinux.zip /DXLDSK_cpp_forLinux);
 - [3] go to the directory (cd /DXLSDK cpp forLinux);
 - [4] unzip the file (unzip dxlsdk_cpp_forlinux.zip).

```
root@robotis-linux:/home/robotis/Downloads# ls
amd-driver-installer-catalyst-13-4-linux-x86.x86_64.zip
amd-driver-installer-catalyst-13-4-x86.x86_64.run
dxlsdk_cpp_forlinux.zip
root@robotis-linux:/home/robotis/Downloads# mkdir /DXLSDK_cpp_forLinux
root@robotis-linux:/home/robotis/Downloads# cp dxlsdk_cpp_forlinux.zip /DXLSDK_cpp_forLinux/
root@robotis-linux:/home/robotis/Downloads# cd /DXLSDK_cpp_forLinux/
root@robotis-linux:/home/robotis/Downloads# cd /DXLSDK_cpp_forLinux/
root@robotis-linux:/DXLSDK_cpp_forLinux# ls
dxlsdk_cpp_forlinux.zip
root@robotis-linux:/DXLSDK_cpp_forLinux# unzip dxlsdk_cpp_forlinux.zip
```

- After decompression verify the contents (type the command ls). Ignoring the zip file there should be the a few sub-directories (build, example, include, lib, and src) and ReleaseNote.txt.

```
root@robotis-linux:/DXLSDK_cpp_forLinux# ls
build dxlsdk_cpp_forlinux.zip example include lib ReleaseNote.txt src
root@robotis-linux:/DXLSDK_cpp_forLinux#
```

- Compile and build the source with the command **make && make install**. This will install and copy the source's library to the PC.

```
mkdir -p ../lib
g++ -O2 -O3 -DLINUX -D GNU SOURCE -Wall -c -I./../include -fPIC -g -c .../src/bul
kread.cpp -o ../build/bulkread.o
../src/bulkread.cpp: In member function 'bool Thor::BulkRead::ChangeBulkReadData
(int, int, int)':
../src/bulkread.cpp:76:1: warning: control reaches end of non-void function [-Wr
eturn-type]
#ar cr /lib/libdxl.so ../build/serialport.o ../build/dynamixel.o ../build/bulkre
ad.o
g++ -shared -fPIC -o ./../lib/libdxl.so ../build/serialport.o ../build/dynamixe
l.o ../build/bulkread.o
mkdir -p ../build
mkdir -p ../lib
#ar cr /lib/libdxl.so ../build/serialport.o ../build/dynamixel.o ../build/bulkre
ad o
g++ -shared -fPIC -o ./../lib/libdxl.so ../build/serialport.o ../build/dynamixe
cp "././lib/libdxl.so" "/usr/local/lib/libdxl.so"
ln -s "/usr/local/lib/libdxl.so" "/usr/local/lib/libdxl.so.2"
ln -s "/usr/local/lib/libdxl.so" "/usr/local/lib/libdxl.so.2.0"
ln -s "/usr/local/lib/libdxl.so" "/usr/local/lib/libdxl.so.2.0.0"
cp -r ./../include/* /usr/local/include/
```

- Set-up is complete
- 1.3.3 Monitoring Dynamixel PRO
 - i. Monitor
 - Linux is not currently supported by RoboPlus software. Fortunately, the downloaded SDK includes an utility called "monitor." "Monitor is a simple command-line based tool that can perform every function of the Windows-based Dynamixel Wizard.
 - To run monitor make sure you are in the example directory (/DXLSDK_cpp_forLinux/example). There will be 4 other sub-directories (each one is an example on implementation of Dynamixel SDK 2.0). Go to the monitor sub-directory (cd monitor).
 - Borwse the contents with the ls command and you will see 3 files (main.cpp, Makefile, and monitor* (*the file monitor, in white, is a remnant during the development of the SDK and it is not functional). Ignore monitor in white font since its not functional simple recompile the code again with the command **make**. After compilation browse the contents again and there should be monitor again, this time in green.



To run monitor simply enter the command ./monitor.

- 1.3.4 Functions of monitor
 - i. Scanning for Dynamixel PRO(s)

- For information on how to operate "monitor" simply enter the command help."Monitor" outpouts a list of available commands.

COMMAND:	
baud [BAUD_NUM]	: Baudrate change to [BAUD_NUM] 0:2400, 1:57600, 2:115200, 3:1M, 4:2M, 5:3M 6:4M, 7:4.5M, 8:10.5M
scan	: Output the current status of all DXLs
ping [ID] [ID]	: Output the current status of [ID]
bp	: Broadcast Ping
wrb w [ID] [ADDR] [VALUE]	: Write 1 byte [VALUE] to [ADDR] of [ID]
WFW [ID] [ADDR] [VALUE]	: Write 2 byte [VALUE] to [ADDR] of [ID]
wrd [ID] [ADDR] [VALUE]	: Write 4 byte [VALUE] to [ADDR] of [ID]
rdb [ID] [ADDR]	: Read 1 byte value from [ADDR] of [ID]
rdw [ID] [ADDR]	: Read 2 byte value from [ADDR] of [ID]
rdd [ID] [ADDR]	: Read 4 byte value from [ADDR] of [ID]
r [ID] [ADDR] [LENGTH]	: Dumps the control table of [ID] [LENGTH] bytes from [ADDR]
mon [ID] [ADDR] b w d	: Refresh byte word dword from [ADDR] of [ID]
reboot [ID]	: Reboot the dynamixel of [ID]
reset [ID] [OPTION]	: Factory reset the dynamixel of [ID] OPTION: 255(All), 1(Except ID), 2(Except ID&Baud)
exit	: Exit this program
[CMD]	

- Notice there 3 types of read and 3 type of write commands. This is because addresses come in 3 different sizes; a byte, a word (2 bytes), or a doule-word "dword" (4 bytes/2 words).
- For bytes apply **rdb** for read and **wrb** for write
- For word apply **rdw** for read and **wrw** for write
- For dword apply **rdd** for read and **wrd** for write.
- Inputting the wrong read or write command for the address will result in a failed or erroneous read or write command
- To scan for Dynamixel PRO(s) simply enter the command **scan**. Another alternative command to check connected Dynamixel PRO(s) is **bp**.



- You will see "[ID:001] ..." This means that the connected Dynamixel PRO is assigned ID 1 (factory default).
- Note: when connecting multiple Dynamxel PROs to the Pc do so one at a time. This is because every new Dynamixel PRO ships with factory default ID of 1; otherwise. When multiple Dynamixel PROs are connected simultaneously "monitor" will not be able to perform a proper scan and it will output as if no Dynamixel PROs are connected (Change of ID number is explained further in this documentation).
- ii. Setting the baud rate of USB-to-Dynamixel dongle
 - This step is very important as the communications speed between USB-to-Dynamixel and Dynamixel PRO(s) must be synchronized. Failing to have a synched baud rate may result in Monitor failing to scan, read, and sent write commands to Dynamixel PRO(s).

- <u>By default this utility sets USB-to-Dynamixel baud rate to 1 (57600 bps)</u>. Changes in baud rate for the USB-to-Dynamixel dongle and Dynamixel PRO will be expleained further in this guide.
- iii. Enabling Torque
 - By default Torque Enable (address #562) is disabled. According to Dynamixel PRO's control table the size of address 562 is of 1 byte. This means that the propercommdn to obtain a readout for address 562 should be **rdb 1 562**



- The read value should output 0 (off), and indeed, "monitor" confirms Torque Enable being off.
- To turn Torque Enable on enter the command **wrb 1 562 1**. (Don't forget to be able to change address values from the EEPROM area ensure that address 562 is set to off).

[CMD] wrb 1	562 1
Success to	write!
[CMD] rdb 1	562
READ VALUE	: 1
[CMD]	

Dynamixel PRO is now ready to operate

- iv. Goal Position
 - According to Dynamixel PRO's Control Table Goal Position is located on address 596 with a size of 4 bytes (a doble-word or "dword").
 - To set a Goal Position to the 9-o-clock position the command should be wrd 1 596 125500 (for 54-series) or wrd 596 75937 (for 42-series)
 - Dynamixel rotates 90 degree.



- v. Present Position
 - According to Dynamixel PRO's Control Table Present Position is located on address 611 with a size of 4 bytes (a doble-word or "dword").
 - To read Present Position the command is rdd 1 611



- "Monitor" will output 2 values, unsigned and signed values. Position values greater or equal than 0 will have the same unsigned and signed values. <u>However</u>, <u>values lesser than 0 will have 2 different values</u> (with the unsigned value never being a negative value). This is due to the sign is assigned a 1 at the most-significant-bit digit of Present Position. In this case ignore the unsigned value.
- Proof: When present position returns a position lesser than 0 the unsigned value always start with a 4e9 value. A value with a magnitude is 4e10 is only possible when is binary equivalent is at least 32-bit long integer. Present Position is 4 bytes in size or 32 bits.

[CMD] wrd 1	596 -125500
Success to	write!
[CMD] rdd 1	611
READ VALUE	: (UNSIGNED) 4294841794 , (SIGNED) -125502
[CMD]	

- vi. Min/Max Positions
 - According to Dynamixel PRO's Control Table Max Position is located on address 36 with a size of 4 bytes (a doble-word or "dword") and Min Position is located on address 40 with 4 bytes as well.
 - Both addresses are in the EEPROM area so Torque Enable must be off. By default the Min and Max positions are set to max (or min) to allow full range of motion.



- Set a sample min or max value (**wrd 1 36 9000**); then set a Goal Position with the allowed range followed by setting a Goal Position outside range. When setting a Goal Position outside range the result is a failure to write.

[CMD] wrd 1 596 8800
Success to write!
[CMD] wrd 1 596 200000
Fail to write!
[CMD] wrd 1 596 20000
Fail to write!
[CMD] []

In this example the value in blue denotes a Goal Position within the max range of 9000 while the values in red are outside 9000.

- vii. Goal Torque
 - According to Dynamixel PRO's Control Table Goal Torque is located on address 604 with a size of 2 bytes (a word).
 - Keep in mind that by default the Operating Mode (address 11 of size 1 byte) is set to 3 (Joint Mode) therefore Operating Mode must be set to 0 (Toque Mode). Operating Mode is in the EEPROM area so Torque Enable must of turned off before making any changes.
 - Before testing Goal Torque make sure to check Torque Limit (address 30, 2 bytes) as Goal Troque cannot exceed Torque Limit. Remember: Torque Limit is also in the EEPROM area so Torque Enable must be turned off to make any changes.
 - Obtain a reading of Torque Limit (**rdw 1 30**) then set Goal Torque within the limit and outside the limit.



In this example Torque Limit outputs a value of 310. Blue denotes a successful write with a Goal Torque of 100 (less than 310), and red denotes a Goal Torque outside the limit

viii. Goal Velocity

- According to Dynamixel PRO's Control Table Goal Velocity is located on address 600 with a size of 4 bytes (a dword).
- Keep in mind that Operating Mode must be set to 1 (Wheel Mode). Operating Mode is in the EEPROM area so Torque Enable must be turned off before making any changes.
- Before testing Goal Velocity make sure to check Velocity Limit (address 32, 2 bytes) as Goal Velocity cannot exceed velocity Limit. Remember: Torque Limit is also in the EEPROM area so Torque Enable must be turned off to make any changes
- Try rotating Dynamixel PRO counterclockwise (CCW direction) by setting values above 0, i.e. wrd 1 600 200. Try the same in the clockwise direction (CW direction) by setting values below 0, i.e. wrd 1 600 -200.
- To stop Dynamizel PRO simply enter a velocity value of 0 (enter the command wrd 1 600 0) or turn Toruq Enable off (enter the command wrb 1 562 0).
- ix. Changing ID
 - According to Dynamixel PRO's Control Table ID is located on address 7 with a size of 1 byte. ID is located under the EEPROM area so Torque Enable (#562) must be turned off before making any changes.
 - The following procedure is the simple way to change ID of Dynamixel PRO (assuming default ID of 1: [1] ensure Torque Enable is off (value = 0) by the command rdb 1 562, [2] enter the command to change ID number to, i.e. #39), with the command wrb 1 7 39, [3] to verify scan for Dynamixel with scan.



This example illustrates a procedure to properly change of ID of a Dynamixel PRO from 1 to 39

- With Dynamixel PRO assigned an ID number dffrent than 1 it is now possible to daisy-chain another Dynamixel PRO.
- x. Baud rate
 - As stated previously the default setting for baud rate of USB-to-Dynamixel is 1 (57600 bps). The factory default setting for Dynamixel PRO is also 1.
 - According to Dynamixel PRO's Control Table ID is located on address 8 with a size of 1 byte. ID is located under the EEPROM area so Torque Enable (#562) must be turned off before making any changes.
 - Assuming that Torque Enable is set to off let's change the baud rate of Dynamixel PRO to 1 Mbps (value = 3). In this case since the ID has been changed from 1 to 39 the proper command should be wrb 39 8 3.
 - Perform a scan with scan (do it twice, three times, etc) and notice that Monitor will be able to scan for ID 39. The reason is because the baud rate of ID 39 and USB-to-Dynamixel don't match.

CMD] scan
[ID:039] Model No : 53768 (0x08 0xD2) SUCCESS
CMD] wrb 39 8 3
Success to write!
CMD] scan
CMD1

In this example the first scan (highlighted in blue) is successful because both USB-to-Dynamixel and Dynamixel PRO have the same baud rate of 57600 bps. After changing baud rate to 1 Mbps to Dynamixel PRO scan fails (highlighted in red).

- To set the baud rate of USB-to-Dynamixel simply enter the command **baud 3**. Afterwards perform a scan and ID 39 will reappear again.

```
[CMD] baud 3
Success to change baudrate! [ BAUD NUM: 3 ]
[CMD] scan
[ID:039] Model No : 53768 (0x08 0xD2) ... SUCCESS
[CMD]
```

After changing the baud rate of USB-to-Dynamixel to match ID 39 the scan is successful

- If there are other Dynamixel PROs connected in daisy-chain make sure to also change the baud rate of these Dynamixel PROs too <u>before changing the baud rate</u> <u>of USB-to-Dynamixel</u>. When performing this procedure the Dynamixel PROs may remain daisy chained.
- xi. Resetting Dynamixel PRO to factory-default settings.
 - Monitor allows for resetting any Dynamixel PRO to factory-default settings. Before exercising this option it is strongly recommended that only one Dynamixel PRO is connected (remove any other daisy-chained Dynamixel PROs).
 - For example the current Dyanmixel PRO in use is currently set with ID 39 and a baud rate of 3 (1 Mbps). The following are the available options
 - The command reset 39 255 resets all settings
 - The command reset 39 1 resets all settings except ID number
 - The command reset 39 2 resets all settings except ID number and baud rate



The illustration above shows a Dynamixel PRO (blue line) with ID 39 and a baud rate of 1 Mbps (not shown) via alternative scan command bp. Then, this Dynamixel PRO is set to factory reset except for the ID number (reset 39 1). The USB-to-Dynamixel is baud rate is set to match the factory –default baudrate of Dynamixel PRO (command baud 1). Factory –default reset (except ID number) is confirmed by a following scan with the bp command highlighted in red.

- 1.4 Examples (Linux)
 - 1.4.1 This section assumes the following:
 - 1) USB-to-Dynamixel dongle is registered under ttyUSB0
 - 2) Dynamixel PRO device is in factory-default settings
 - 3) SDK 2.0 is properly set-up
 - 4) Torque Enable (562) is set to 0 (off state)
 - 1.4.2 Initializing USB-to-Dynamixel and Dynamixel PRO
 - i. Connect(/sdk_20_directory/example/Connect/main.cpp)
 - USB-to-Dynamixel is registered under ttyUSB0 in /dev
 - To initialize (open port) simply enter its location at the course code.
 - Before ending the program USB-to-Dynamixel must be disconnected (close port)

```
/dxl_sdk-20/example/Connect/main.cpp (partly shown)
```

```
using namespace DXL_PRO;
```

```
int main()
```

```
{
```

. . .

Dynamixel DXL("/dev/ttyUSB0"); // must declare location of USB-to-Dynamixel

```
// open device
if( DXL.Connect() == 0 )
{
    printf( "Failed to open USB2Dynamixel!\n" );
    return 0;
}
else
```

printf("Succeed to open USB2Dynamixel!\n");

// must close USB-to-Dynamixel before ending
DXL.Disconnect();
return 0;

- ii. Enabling Dynamixel PRO
 - As previously mentioned baud rate of Dynamixel PRO and USB-to-Dynamixel must match.
 - This can be easily achieved by simply setting the value for baud rate.
 - After setting matching baud rate wirte commands to EEPROM area can be sent
 - Torque Enable must be on to send write commands to the RAM area.
 - Read commands can be sent at any time regardless of Torque Enable value.

```
/dx1 sdk-20/example/TorqueEnable/main.cpp (partly shown)
// declare the following in preprocessor for illustrative purposes
#define TOROUE ENABLE
                                  562
#define DEFAULT BAUDNUM
                                   1
using namespace DXL PRO;
Dynamixel DXL("/dev/ttyUSB0"); // must declare location of USB-to-Dynamixel
int main()
{
    int result, error = 0, On; // declare arbitrary variables for read commands
    // open device
    if (DXL.Connect) == 0
     {
         printf( "Failed to open USB2Dynamixel!\n" );
         return 0;
    else
         printf( "Succeed to open USB2Dynamixel!\n" );
    // Now verify baudrate USB-to-Dynamixel
    if(DXL.SetBaudrate(DEFAULT BAUDNUM) == true)
     ł
       printf( "Succeed to set USB2DXL baudrate to 57600 bps!\n" );
    else
         printf( "Failed to set USB2DXL baudrate to 57600 bps!\n" );
         printf( "Press any key to end ...\n" );
         getch();
         return 0:
    printf( "Turning Torque Enable on for ID 1\n" );
```

// This step is essential. After Torque Enable is on RAM area addresses can be changed. DXL.WriteByte(1, TORQUE_ENABLE, 1, 0); // after enabling Torque Enable // sad main routine involving RAM address(es) here (i.e. Goal Position) ... // the following lines verify that Dynamixel's Torque Enable has been turned on // by outputting a "1" onscreen result = DXL.ReadByte(1, TORQUE_ENABLE, &On, &error); if (result == COMM_RXSUCCESS) printf("%d\n", On); // must close USB-to-Dynamixel before ending DXL.Disconnect(); return 0;

- 1.4.3 Implementing read/write functions
 - i. Write functions
 - Monitor for Dynamixel PRO in Linux emphasized that different size addresses required different type of write commands. The same concept applies at the source code level.
 - There are 4 types of write commands for Dynamixel PRO
 - Implement WriteByte for addresses with size of 1 byte
 - Implement WriteWord for addresses with size of 2 bytes
 - Implement WriteDWord for addresses with size of 4 bytes
 - Implement SyncWrite (explained later) for for multiple Dynamixel PROs at once.
 - ii. Read functions
 - Monitor for Dynamixel PRO in Linux emphasized that different size addresses required different type of read commands. The same concept applies at the source code level.
 - There are 4 types of write commands for Dynamixel PRO
 - Implement ReadByte for addresses with size of 1 byte
 - Implement ReadWord for addresses with size of 2 bytes
 - Implement ReadDWord for addresses with size of 4 bytes
 - Implement BulkRead (explained later) for multiple Dynamixel PROs at once.

The following example is a simple read/write sample code of Dynamixel PRO in Joint Mode (default)

/dxl_sdk-20/example/ReadWrite/main.cpp (partly shown)		
// declare the following in preprocessor for illustrative purposes		
#define P_TORQUE_ENABLE	562	
#define P GOAL POSITION LL	596	
#define P_PRESENT_POSITION_LL	611	
#define P_MOVING	610	

```
#define DEFAULT BAUDNUM
                                       1
#define DEFAULT ID
                                      1
using namespace DXL PRO;
Dynamixel DXL("/dev/ttyUSB0"); // must declare location of USB-to-Dynamixel
int main()
{
    int result, error = 0, index = 0, PresentPos = 0, On; // declare arbitrary variables for read
commands
    int GoalPos[2] = (-150000, 150000);
    // open device
    if (DXL.Connect) == 0
    ł
         printf( "Failed to open USB2Dynamixel!\n" );
         return 0;
    }
    else
         printf( "Succeed to open USB2Dynamixel!\n" );
    // Now verify baudrate USB-to-Dynamixel
    if(DXL.SetBaudrate(DEFAULT BAUDNUM) == true)
    {
       printf( "Succeed to set USB2DXL baudrate to 57600 bps!\n" );
    }
    else
    Ş
         printf( "Failed to set USB2DXL baudrate to 57600 bps!\n" );
         printf( "Press any key to end ...\n" );
         _getch():
         return 0;
    printf( "Turning Torque Enable on for ID 1\n" );
    // This step is essential. After Torque Enable is on RAM area addresses can be changed.
    DXL.WriteByte(DEFAULT ID, P TORQUE ENABLE, 1, &error);
    While(1);
    DXL.WriteDWord(DEFAULT ID, P GOAL POSITION LL, GoalPos[index], &error);
    do {
    result = DXL.ReadDWord(DEFAULT ID, P PRESENT POSITION LL, &PresentPos,
&error);
    if (result == COMM RXSUCCESS)
       printf( " %d %d\n", GoalPos[index], PresentPos);
    // must close USB-to-Dynamixel before ending
    DXL.Disconnect();
    return 0;
```

}

Similarly several types of write commands can be implemented according to the size of the corresponding address.

The following example combines several features of Dyanmixel PRO, such as Operating Mode (11) and its different type of operations; and implementing their respective write function.

This example also illustrates how the user can easily turn Torque Enable on and off at will enabling easy manipulation of EEPROM addresses.

```
/dxl_sdk-20/example/OperatingMode/main.cpp (partly shown)
// declare the following in preprocessor for illustrative purposes
#define OPERATING MODE
                                   11
#define TORQUE ENABLE
                                   562
#define GOAL POSITION
                                   596
#define GOAL TORQUE
                                   604
#define GOAL VELOCITY
                                  600
#define DEFAULT BAUDNUM
                                      1
#define CONTROL PERIOD
                                     (10) // time period, arbitrary value
using namespace DXL PRO;
Dynamixel DXL("/dev/ttyUSB0"); // must declare location of USB-to-Dynamixel
int main()
{
    int GoalPos 250000; // for 54-series
    // int GoalPos = 151875; // for 42-series
    int GoalVel = -30001
    int GoalTorque = 120;
    // open device and set baud rate here
    if (DXL.Connect) == 0
    // Torque Enable is off by default
    // The following line is not necessary but it ensures
    // Torque Enable is off regardless
    DXL.WriteByte(1, TORQUE ENABLE, 0, 0);
    While(1); {
         switch( getch()) { // use simply switch loop to change between modes
          case 0x1b: // ESC key to set to default settings
              DXL.WriteByte(1, TORQUE ENABLE, 0, 0);
              usleep(CONTROL PERIOD*100);
              DXL.WriteByte(1, OPERATING MODE, 3, 0);
              break;
         case 0x30: // "0" key to set in Torque Mode
              DXL.WriteByte(1, OPERATING MODE, 0, 0);
              usleep(CONTROL PERIOD*100);
              DXL.WriteByte(1, TORQUE ENABLE, 1, 0);
              usleep(CONTROL PERIOD*100);
              // Goal Torque (604) size is 2 bytes (a word)/
```

<pre>// WriteWord is the appropriate function</pre>
DXL.WriteWord(1, GOAL_TORQUE, GoalTorque, 0);
break;
case 0x31: // "1" key to set in Wheel Mode
DXL.WriteByte(1, OPERATING_MODE, 1, 0);
usleep(CONTROL_PERIOD*100);
DXL.WriteByte(1, TORQUE_ENABLE, 1, 0);
usleep(CONTROL_PERIOD*100);
// Goal Velocity (600) size is 4 bytes (a double-word)
// WriteDWord is the appropriate function
DXL.WriteDWord(1, GOAL_VELOCITY, GoalVel, 0);
break;
defaultt: // every other key and Dynamixel PRO remains in Joint Mode
DXL.WriteByte(1, OPERATING_MODE, 3, 0);
usleep(CONTROL_PERIOD*100);
DXL.WriteByte(1, TORQUE_ENABLE, 1, 0);
usleep(CONTROL_PERIOD*100);
// Goal Position (596) size is 4 bytes (a double-word)
// WriteDWord is the appropriate function
DXL.WriteDWord(1, GOAL_POSITION, GoalPos, 0);
break; }
}
// must close USB-to-Dynamixel before ending
DXL.Disconnect();
return 0;

Similarly as multiple write functions can be applied; multiple read functions can be implemented.

/dxl_sdk-20/example/ReadValues/main.cpp (partly shown)			
// declare the following in preprocessor for illustrative purposes			
#define OPERATING_MODE	11		
#define TORQUE_ENABLE	562		
#define GOAL_POSITION	596		
#define GOAL_TORQUE	604		
#define GOAL_VELOCITY	600		
#define PRESENT_POSITION	611		
#define PRESENT_CURRENT	621		
#define PRESENT_VELOCITY	615		
#define DEFAULT_BAUDNUM	1		
#define CONTROL_PERIOD	(10) // time period, arbitrary value		
using namespace DXL_PRO;			
Dynamixel DXL("/dev/ttyUSB0");	<pre>// must declare location of USB-to-Dynamixel</pre>		
int main()			
int GoalPos 250000; // for 54-series			

```
// int GoalPos = 151875; // for 42-series
int GoalVel = -30001
int GoalTorque = 120;
int result = COMM TXFAIL;
int error = 0;
int val1, val2, val3;
. . .
// open device and set baud rate here
if (DXL.Connect) == 0)
// Torque Enable is off by default
// The following line is not necessary but it ensures
// Torque Enable is off regardless
DXL.WriteByte(1, TORQUE ENABLE, 0, 0);
While(1); {
    switch( getch()) { // use simply switch loop to change between modes
     case 0x1b: // ESC key to set to default settings
          DXL.WriteByte(1, TOROUE ENABLE, 0, 0);
          usleep(CONTROL PERIOD*100);
          DXL.WriteByte(1, OPERATING MODE, 3, 0);
          break; }
     case 0x30: // "0" key to set in Torque Mode
          DXL.WriteByte(1, OPERATING MODE, 0, 0);
          usleep(CONTROL PERIOD*100);
          DXL.WriteByte(1, TORQUE ENABLE, 1, 0);
          usleep(CONTROL PERIOD*100);
          // Goal Torque (604) size is 2 bytes (a word)/
          // WriteWord is the appropriate function
          DXL.WriteWord(1, GOAL TORQUE, GoalTorque, & error);
           usleep(CONTROL PERIOD*200);
          // after an arbitrary period (i.e. 2 secs in this case)
          // output onscreen
          // Present Current (621) size is 2 bytes (a word)
          // ReadWord is the appropriate function
          result = DXL.ReadWord(1, PRESENT CURRENT, &val1, &error);
          if(result == COMM RXSUCCESS)
                  printf( "%d\n", val1);
          break; }
   case 0x31: // "1" key to set in Wheel Mode
          DXL.WriteByte(1, OPERATING MODE, 1, 0);
          usleep(CONTROL PERIOD*100);
          DXL.WriteByte(1, TORQUE ENABLE, 1, 0);
          usleep(CONTROL PERIOD*100);
          // Goal Velocity (600) size is 4 bytes (a double-word)
          // WriteDWord is the appropriate function
          DXL.WriteDWord(1, GOAL VELOCITY, GoalVel, &error);
          usleep(CONTROL PERIOD*200);
          // after an arbitrary period (i.e. 2 secs in this case)
          // output onscreen
```

```
// Present Velocity (615) size is 4 bytes (a double-word)
          // ReadDWord is the appropriate function
          result = DXL.ReadDWord(1, PRESENT VELOCITY, &val2, &error);
          if(result == COMM RXSUCCESS)
                  printf( "%d\n", val2);
          break; }
   defaultt: // every other key and Dynamixel PRO remains in Joint Mode
          DXL.WriteByte(1, OPERATING MODE, 3, 0);
          usleep(CONTROL PERIOD*100);
          DXL.WriteByte(1, TORQUE ENABLE, 1, 0);
          usleep(CONTROL PERIOD*100);
          // Goal Position (596) size is 4 bytes (a double-word)
          // WriteDWord is the appropriate function
          DXL.WriteDWord(1, GOAL POSITION, GoalPos, & error);
          usleep(CONTROL PERIOD*200);
          // after an arbitrary period (i.e. 2 secs in this case)
          // output onscreen
          // Present Position (611) size is 4 bytes (a double-word)
          // ReadDWord is the appropriate function
          result = DXL.ReadDWord(1, PRESENT POSITION, &val3, &error);
          if(result == COMM RXSUCCESS)
                  printf( "%d\n", val3);
          break; }
   }
// must close USB-to-Dynamixel before ending
DXL.Disconnect();
return 0;
```

- iii. SyncWrite
 - While the abovementioned read and write commands are simple to implement the amount of code lines necessary for multiple addresses on multiple Dynamixel PROs can be rather staggering.
 - SynchWrite simplifies commands by setting, preparing and then transmitting data packets.
 - SyncWrite allows one single commands to address multiple values for multiple Dynamixel PROs at a given address.

The following example illustrates a single SyncWrite command to set different Goal Positions to more than 1 Dynamixel PRO

/dxl_sdk-20/example/SyncWrite/main.cpp (partly shown)			
// declare the following in preprocessor for illustrative purposes			
#define OPERATING MODE	11		
#define TORQUE ENABLE	562		
#define GOAL POSITION	596		
#define GOAL_TORQUE	604		

```
. . .
using namespace DXL PRO;
Dynamixel DXL("/dev/ttyUSB0"); // must declare location of USB-to-Dynamixel
int main()
{
    int GoalPos, i;
    int id[NUM ACTUATOR];
    int phase[NUM ACTUATOR];
    // open device and set baud rate here
    if (DXL.Connect() == 0)
    DXL.WriteByte(1, TORQUE ENABLE, 1, 0); // for ID 1
    // do the same for other IDs
    do {
        for (i= 0; I < NUM ACTUATOR; i++) {
          GoalPos = (int)((sin(theta+phase[i]))(double)AmPos);
          // packet preparation set below.
          // ID requires 1 byte
          // Goal Position requires 4 bytes
          // in this case different Goal Position values for different ID
          param[i^{(1+4)+0]} = (unsigned char)id[i];
          param[i*(1+4)+1] = DXL LOBYTE(DXL LOWORD(GoalPos));
          param[i*(1+4)+2] = DXL HIBYTE(DXL LOWORD(GoalPos));
          param[i*(1+4)+3] = DXL LOBYTE(DXL HIWORD(GoalPos));
          param[i^{(1+4)+4}] = DXL HIBYTE(DXL HIWORD(GoalPos));
       }
    // 4 is for size of Goal Position, NUM ACTUATOR is for size of array
    result = DXL.SyncWrite(P GOAL POSITION LL, 4, param, NUM ACTUATOR*(1+4));
    . . .
    while (theta < 2*PI);
    // must close USB-to-Dynamixel before ending
    DXL.Disconnect();
    return 0;
```

iv. BulkRead

- Like SyncWrite BulkRead allows reading multiple addresses of multiple Dynamixel PROs simultaneously

/dxl_sdk-20/example/BulkRead/main.cpp (partly shown)		
 using namespace DXL_PRO;		
Dynamixel DXL("/dev/ttyUSB0"); // must declare location of USB-to-Dynamixel		
int main()		

```
{
    int id[NUM ACTUATOR];
    int phase[NUM ACTUATOR];
    // open device and set baud rate here
    if (DXL.Connect) == 0)
    DXL.WriteByte(1, TOROUE ENABLE, 1, 0); // for ID 1
    // do the same for other IDs
    while(1) {
        result = bulkread.SendTxPacket();
        for (i = 0; i < NUM ACTUATOR; i++) (
             bulkread.GetDWordValue(id[i], addr[i] (long*)&read val);
             printf("ID:%d, READ VALUE:%d\n", id[i], read val);
             // print out address(es) values of IDs invollved
    }
    // must close USB-to-Dynamixel before ending
    DXL.Disconnect();
    return 0;
```

- v. WriteByte/WriteWord/WriteDWord vs SyncWrite
 - Although both are write commands their implementation can be quite different.
 - WriteByte WriteWord, WriteDWord are <u>explicit write commands</u> that allow writing specific IDs at specific addresses (i.e. LED control on ID 1 while ID 2 rotates in wheel mode). The user has much greater control of Dynamixel PRO with these commands. The drawback on implementing this fuction is that with multiple IDs with multiple addresses then the code can become very long.
 - SyncWrite, in the other hand, is a more <u>implicit write command</u>. One command is enough for multiple IDs. The drawback is that the user does not have as much control as with WriteByte/Word/DWord since SynchWrite does not allow control of specific addresses from specific IDs at a given moment.
 - Notice the difference from the sample codes from OperatingMode and SyncWrite. OperatingMode require many write commands for different Control Table addresses, while SynchWrite is much shorter but only deals with one Control Table address

2 Reference

- 2.1 Default values by model
 - 2.1.1 H Series
 - i. H54-200-S500-R

Name	Default Value
ID	1
BaudRate	1 (57600bps)
Max Position Limit	251000
Min Position Limit	-251000
Velocity Limit	16600
Current Limit	620
Velocity I Gain	14
Velocity P Gain	399
Position P Gain	32

ii. H54-100-S500-R

Name	Default Value		
ID	1		
BaudRate	1 (57600bps)		
Max Position Limit	251000		
Min Position Limit	-251000		
Velocity Limit	17000		
Current Limit	310		
Velocity I Gain	16		
Velocity P Gain	256		
Position P Gain	32		

iii. H42-20-S300-R

Name	Default Value		
ID	1		
BaudRate	1 (57600bps)		
Max Position Limit	151875		
Min Position Limit	-151875		
Velocity Limit	10300		
Current Limit	465		
Velocity I Gain	40		
Velocity P Gain	440		
Position P Gain	32		

2.1.2 M Series

i. M54-60-S250-R

Name	Default Value
ID	1
BaudRate	1 (57600bps)

Max Position Limit	125700
Min Position Limit	-125700
Velocity Limit	8000
Current Limit	180
Velocity I Gain	16
Velocity P Gain	256
Position P Gain	32

ii. M54-40-S250-R

Name	Default Value
ID	1
BaudRate	1 (57600bps)
Max Position Limit	125700
Min Position Limit	-125700
Velocity Limit	8000
Current Limit	180
Velocity I Gain	16
Velocity P Gain	256
Position P Gain	32

2.1.3 L Series

i. L54-50-S290-R

Name	Default Value
ID	1
BaudRate	1 (57600bps)
Max Position Limit	103860
Min Position Limit	-103860
Velocity Limit	8000
Current Limit	180
Velocity I Gain	16
Velocity P Gain	256
Position P Gain	32

ii. L54-30-S400-R

Name	Default Value		
ID	1		
BaudRate	1 (57600bps)		
Max Position Limit	144180		
Min Position Limit	-144180		
Velocity Limit	13000		
Current Limit	180		
Velocity I Gain	16		
Velocity P Gain	256		
Position P Gain	32		

iii. L42-10-S300-R

Name	Default Value
ID	1
BaudRate	1 (57600bps)
Max Position Limit	151875
Min Position Limit	-151875
Velocity Limit	
Current Limit	
Velocity I Gain	
Velocity P Gain	
Position P Gain	

2.2 Control Table of Dynamixel Pro

(R : Read, RW : Read and Write)

AREA	address	size (byte)	feature	description	type	Default value
	0	2	Model Number	Model number	R	-
	6	1	Version of Firmware	Firmware version info	R	-
	7	1	ID	ID of Dynamixel PRO	RW	1
	8	1	Baud Rate	Baud rate	RW	1
	9	1	Return Delay Time	Return delay time	RW	250
	11	1	Operating mode	Operating mode	RW	3
	13	4	Homing offset	Homing value	RW	0
	17	4	Moving threshold	Moving threshold	RW	50
	21	1	Temperature limit	Internal temperature limit	RW	80
	22	2	Max Voltage Limit	Opeting upper limit voltage	RW	400
	24	2	Min Voltage Limit	Operating lower limit voltage	RW	150
	26	4	acceleration Limit	Acceleration limit	RW	-
EEPROM	30	2	Torque limit	Torque limit	RW	-
	32	4	Velocity Limit	Velocity limit	RW	-
	36	4	Max Position Limit	Position upper limit	RW	-
	40	4	Min Position Limit	Position lower limit	RW	-
	44	1	External Port Mode 1	External Port Mode 1	RW	0
	45	1	External Port Mode 2	External Port Mode 2	RW	0
	46	1	External Port Mode 3	External Port Mode 3	RW	0
	47	1	External Port Mode 4	External Port Mode 4	RW	0
	48	1	Shutdown	Shutdown	RW	26
	49	2	Indirect Address 1	Indirect Address 1	RW	634
	51	2	Indirect Address 2	Indirect Address 2	RW	635
	53	2	Indirect Address 3	Indirect Address 3	RW	636
				Indirect Address N	RW	-
	569	2	Indirect Address 256	Indirect Address 256	RW	889

	562	1	Torque Enable	torque On/Off	RW	0
	563	1	LED RED	RED LED intensity value	RW	0
	564	1	LED GREEN	GREEN LED intensity value	RW	0
	565	1	LED BLUE	BLUE LED intensity value	RW	0
	586	2	Velocity I Gain	Velocity I Gain	RW	-
	588	2	Velocity P Gain	Velocity P Gain	RW	-
	594	2	Position P Gain	Velocity P Gain	RW	-
	596	4	Goal position	Target position value	RW	-
	600	4	Goal velocity	Target velocity value	RW	0
	604	2	Goal Torque	Target torque value	RW	0
	606	4	Goal acceleration	Target acceleration value	RW	0
	610	1	Moving	moving	R	-
	611	4	Present position	Current position value	R	-
	615	4	Present velocity	Current velocity	R	-
	621	2	Present Current	Present current value	R	-
DANG	623	2	Present input voltage	Current input voltage	R	-
RAM	625	1	Present temperature	Current internal temperature	R	-
	626	2	External Port Data 1	External Port Data 1	R / RW	0
	628	2	External Port Data 2	External Port Data 2	R / RW	0
	630	2	External Port Data 3	External Port Data 3	R / RW	0
	632	2	External Port Data 4	External Port Data 4	R / RW	0
	634	1	Indirect Data 1	Indirect Data 1	RW	0
	635	1	Indirect Data 2	Indirect Data 2	RW	0
	636	1	Indirect Data 3	Indirect Data 3	RW	0
			Indirect Data N	Indirect Data N	RW	0
	889	1	Indirect Data 256	Indirect Data 256	RW	0
	890	1	Registered Instruction	Registered Instruction value	R	0
	891	1	Status Return Level	Status Return Level value	RW	2
	892	2	Hardware error status	Hardware error status value	R	0

2.3 Features (by width size)

2.3.1 54-series (H54, M54, L54)





2.3.2 42-series (H42, L42)



- 2.4 Dimensions
 - 2.4.1 H Series
 - i. H54-200-S500-R



ii. H54-100-S500-R



iii. H42-20-S300-R



- 2.4.2 M Series
 - i. M54-60-S250-R



ii. M54-40-S250-R



- 2.4.3 L Series
 - i. L54-50-S290-R



ii. L54-30-S400-R



iii. L42-10-S300-R









2.5 Model notation



Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

ROBOTIS:

<u>902-0079-000</u> <u>902-0078-000</u> <u>902-0089-000</u> <u>902-0090-000</u> <u>902-0080-000</u> <u>902-0088-000</u> <u>902-0081-000</u> <u>902-0105-</u> 000 902-0092-000 902-0106-000 902-0091-000