Kontrol dan komunikasi servo Dynamixel AX-18A menggunakan Raspberry Pi dapat dilakukan dengan memanfaatkan sistem publish-subscribe. Artinya, Raspberry Pi berfungsi sebagai publisher dan subscriber, yaitu pengirim perintah dan penerima pesan terkait kontrol servo.

Berikut komponen-komponen yang diperlukan:

1. Raspberry Pi: otak sistem.
2. Servo Dynamixel AX-18A: servo yang akan dikontrol.
3. Modul U2D2: untuk komunikasi antara Raspberry Pi dan servo.
4. Kabel power dan data: untuk menyambungkan servo ke modul U2D2.
5. Breadboard dan kabel jumper: Untuk membuat koneksi yang lebih mudah (jika diperlukan).
6. Python atau ROS: untuk pemrograman dan kontrol.

Berikut langkah-langkah kontrol dan komunikasi dengan servo:

1. Instalasi dan Persiapan Raspberry Pi
   - Instalasi OS: Pastikan Raspberry Pi Anda terpasang dengan Raspberry Pi OS atau Raspbian yang terbaru.
   - Perbarui Sistem: Jalankan perintah berikut untuk memperbarui sistem:
     a. `sudo apt update`
     b. `sudo apt upgrade`

2. Instalasi ROS (optional)
   - Tambahkan repository:
     a. `sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -cs) main" > /etc/apt/sources.list.d/ros-latest.list'`
   - Instal GPG key:
     a. `sudo apt install curl`
     b. `curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo apt-key add -`
   - Instal ROS:
     a. `sudo apt update`
     b. `sudo apt install ros-noetic-desktop-full`
   - Setup environment: tambahkan ke .bashrc:
     a. `echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc`
     b. `source ~/.bashrc`
   - Instal dependencies:
     a. `sudo apt install python-rosdep`
     b. `sudo rosdep init`

```
c. rosdep update
```

3. Koneksi U2D2 dan Servo
   - Sambungkan Servo: Hubungkan kabel servo ke modul U2D2. Pastikan untuk menghubungkan kabel power (VDD), ground (GND), dan data (DYNAMIXEL).
   - Hubungkan U2D2 ke Raspberry Pi: Gunakan USB untuk menghubungkan modul U2D2 ke Raspberry Pi.

4. Instalasi Pustaka Dynamixel
   - Install pustaka Dynamixel untuk Python agar dapat berkomunikasi dengan servo:

```
a. pip install dynamixel-sdk
```

5. Membuat Node ROS untuk Mengontrol Servo
   - Buat folder dan file node:

```
a. mkdir -p ~/catkin_ws/src/dynamixel_control/src
b. cd ~/catkin_ws/src/dynamixel_control/src
c. touch dynamixel_control.py
d. chmod +x dynamixel_control.py
```

   - Tulis kode di dynamixel_control.py:

```
a. import rospy
b. from std_msgs.msg import Int32
c. from dynamixel_sdk import *  # Dynamixel SDK
d.
e. # Mengatur parameter servo
f. DEVICENAME = '/dev/ttyUSB0'  # Ganti dengan nama port
   U2D2 Anda
g. BAUDRATE = 57600
h. PROTOCOL_VERSION = 2.0
i. DXL_ID = 1  # ID servo
j. ADDR_MX_TORQUE_ENABLE = 64  # Alamat untuk
   mengaktifkan torsi
k. ADDR_MX_PRESENT_POSITION = 132  # Alamat untuk posisi
   saat ini
l. ADDR_MX_GOAL_POSITION = 116  # Alamat untuk posisi
   tujuan
m. TORQUE_ENABLE = 1  # Aktifkan torsi
n. TORQUE_DISABLE = 0  # Nonaktifkan torsi
o.
p. # Inisialisasi port dan servo
q. port_handler = PortHandler(DEVICENAME)
r. packet_handler = PacketHandler(PROTOCOL_VERSION)
```

```
s.
t. # Fungsi untuk menginisialisasi
u. def init():
v.     rospy.init_node('dynamixel_control',
   anonymous=True)
w.     port_handler.openPort()
x.     port_handler.setBaudRate(BAUDRATE)
y.     packet_handler.write1ByteTxRx(port_handler,
   DXL_ID, ADDR_MX_TORQUE_ENABLE, TORQUE_ENABLE)
z.     rospy.loginfo("Dynamixel connected")
aa.
bb.   # Fungsi untuk mengontrol servo
cc.   def set_position(data):
dd.       goal_position = data.data
ee.       packet_handler.write2ByteTxRx(port_handler,
   DXL_ID, ADDR_MX_GOAL_POSITION, goal_position)
ff.       rospy.loginfo(f"Set position to:
   {goal_position}")
gg.
hh.   def listener():
ii.       rospy.Subscriber('servo_position', Int32,
   set_position)
jj.       rospy.spin()
kk.
ll.   if __name__ == '__main__':
mm.       try:
nn.           init()
oo.           listener()
pp.       except rospy.ROSInterruptException:
qq.           pass
rr.       finally:
ss.           packet_handler.write1ByteTxRx(port_handler,
   DXL_ID, ADDR_MX_TORQUE_ENABLE, TORQUE_DISABLE)
tt.           port_handler.closePort()
```

6. Membuat Publisher untuk Mengirim Posisi ke Servo
   - Buat folder dan file publisher:

```
a. cd ~/catkin_ws/src/dynamixel_control/src
b. touch dynamixel_publisher.py
```

```
c. chmod +x dynamixel_publisher.py
```

- Tulis kode di dynamixel_control.py:

```
a. import rospy
b. from std_msgs.msg import Int32
c.
d. def talker():
e.     pub = rospy.Publisher('servo_position', Int32,
   queue_size=10)
f.     rospy.init_node('dynamixel_publisher',
   anonymous=True)
g.     rate = rospy.Rate(1)  # 1 Hz
h.     while not rospy.is_shutdown():
i.         position = int(input("Enter desired position
   (0 - 1023): "))
j.         pub.publish(position)
k.         rate.sleep()
l.
m. if __name__ == '__main__':
n.     try:
o.         talker()
p.     except rospy.ROSInterruptException:
q.         pass
```

7. Compile dan Run
   - Compile workspace:

     ```
     a. cd ~/catkin_ws
     b. catkin_make
     ```

   - Run node: buka terminal baru dan run node kontrol servo:

     ```
     a. rosrun dynamixel_control dynamixel_control.py
     ```

   - Run publisher: buka terminal baru dan run publisher:

     ```
     a. rosrun dynamixel_control dynamixel_publisher.py
     ```

8. Pengujian
   - Masukkan nilai posisi (antara 0 – 1023) saat diminta oleh publisher.
     a. Servo seharusnya akan bergerak sesuai dengan nilai tersebut.
     b. Jika pergerakannya tidak sesuai, coba debug (mengecek kesalahan prosedural).
     c. Jika debug dan tidak kunjung berhasil, coba ganti servo (mungkin terdapat error di servonya itu sendiri).