

LAMPIRAN-LAMPIRAN

Lampiran 1. Sistem yang berhasil dibuat bersama peneliti



1. *Listing* Program Robot 1

```
def variabel():  
    global homeX, homeY, homeZ, pickX, pickY, pickZ, prePickY, placeX, placeY,  
    placeZ, prePlaceY  
  
    homeX = 200  
  
    homeY = 0  
  
    homeZ = 80  
  
    pickX = 236.312  
  
    pickY = 12.6766  
  
    pickZ = -64.7932  
  
    prePickY = 119.8329  
  
    placeX = 201.811  
  
    placeY = 220.9658  
  
    placeZ = 52.8422  
  
    prePlaceY = -119.8329  
  
  
def menungguSinyalPushButton():  
    while not (dType.GetIODI(api, 19)[0]) == 0:  
        pass  
  
  
def pickAndPlace():  
    global homeX, homeY, homeZ, prePickY, pickX, pickY, pickZ, prePlaceY,  
    placeX, placeY, placeZ  
  
    current_pose = dType.GetPose(api)  
  
    dType.SetPTPCmdEx(api, 2, homeX, homeY, homeZ, current_pose[3], 1)  
  
    current_pose = dType.GetPose(api)
```

```
dType.SetPTPCmdEx(api, 2, homeX, prePickY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, pickX, pickY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, pickX, pickY, pickZ, current_pose[3], 1)
dType.SetEndEffectorSuctionCupEx(api, 1, 1)
dType.dSleep(1000)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, pickX, pickY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, pickX, prePickY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, homeX, homeY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, homeX, prePlaceY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, placeX, placeY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, placeX, placeY, placeZ, current_pose[3], 1)
dType.SetEndEffectorSuctionCupEx(api, 0, 1)
dType.dSleep(1000)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, placeX, placeY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, homeX, prePlaceY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
```

```

dType.SetPTPCmdEx(api, 2, homeX, homeY, homeZ, current_pose[3], 1)

def kirimSinyalKubusSiap():
    dType.SetIODOEx(api, 13, 1, 1)
    dType.dSleep(1000)
    dType.SetIODOEx(api, 13, 0, 1)

def gerakConveyor1():
    while (dType.GetIODI(api, 20)[0]) == 1:
        STEP_PER_CRICLE = 360.0 / 1.8 * 10.0 * 16.0
        MM_PER_CRICLE = 3.1415926535898 * 36.0
        vel = float(40) * STEP_PER_CRICLE / MM_PER_CRICLE
        dType.SetEMotorEx(api, 0, 1, int(vel), 1)
        STEP_PER_CRICLE = 360.0 / 1.8 * 10.0 * 16.0
        MM_PER_CRICLE = 3.1415926535898 * 36.0
        vel = float(0) * STEP_PER_CRICLE / MM_PER_CRICLE
        dType.SetEMotorEx(api, 0, 0, int(vel), 1)

dType.SetHOMECmdEx(api, 0, 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 1, current_pose[0], current_pose[1], current_pose[2],
0, 1)
dType.SetEndEffectorParamsEx(api, 59.7, 0, 0, 1)
dType.SetIOMultiplexingEx(api, 19, 3, 1)
dType.SetIOMultiplexingEx(api, 20, 3, 1)
dType.SetIOMultiplexingEx(api, 13, 1, 1)

```

```
variabel()
while True:
    menungguSinyalPushButton()
    pickAndPlace()
    kirimSinyalKubusSiap()
    gerakConveyor1()
```

2. *Listing Program Robot 2*

```
def variabel():
    global stepHijauX, stepHijauY, stepHijauZ, homeX, homeY, homeZ, pickX,
    pickY, pickZ, hijauX, hijauY, hijauZAt, preHijauY, lainX, lainY, lainZ,
    preLainY, colorX, colorY, colorZ

    stepHijauX = 0
    stepHijauY = 0
    stepHijauZ = 0

    homeX = 200
    homeY = 0
    homeZ = 80

    pickX = 259.581
    pickY = 16.452
    pickZ = 13.6179

    hijauX = 33.9745
    hijauY = 274.0255
    hijauZAt = 16.979

    preHijauY = 148.8501

    lainX = 4.9991
    lainY = -230.1826
    lainZ = 15.5659
```

```
preLainY = -105.5374
```

```
colorX = 285.3352
```

```
colorY = 88.7919
```

```
colorZ = 39
```

```
def menungguSinyalInfraRed1():
```

```
    while not (dType.GetIODI(api, 19)[0]) == 0:
```

```
        pass
```

```
def kirimSinyalKubusHijau():
```

```
    dType.SetIODOEx(api, 6, 1, 1)
```

```
    dType.dSleep(1000)
```

```
    dType.SetIODOEx(api, 6, 0, 1)
```

```
def pickAndPlaceHijau():
```

```
    global colorX, colorY, homeZ, homeX, homeY, preHijauY, hijauX, hijauY,  
    hijauZAt
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, colorX, colorY, homeZ, current_pose[3], 1)
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, homeX, homeY, homeZ, current_pose[3], 1)
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, homeX, preHijauY, homeZ, current_pose[3], 1)
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, hijauX, hijauY, homeZ, current_pose[3], 1)
```

```
    current_pose = dType.GetPose(api)
```

```

dType.SetPTPCmdEx(api, 2, hijauX, hijauY, hijauZAt, current_pose[3], 1)
dType.SetEndEffectorSuctionCupEx(api, 0, 1)
dType.dSleep(1000)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, hijauX, hijauY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, homeX, preHijauY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, homeX, homeY, homeZ, current_pose[3], 1)

def kirimSinyalKubusLain():
    dType.SetIODOEx(api, 4, 1, 1)
    dType.dSleep(1000)
    dType.SetIODOEx(api, 4, 0, 1)

def menungguSinyalConveyor3():
    while not (dType.GetIODI(api, 7)[0]) == 0:
        pass

def pickAndPlaceLain():
    global colorX, colorY, homeZ, homeX, homeY, preLainY, lainX, lainY, lainZ
    current_pose = dType.GetPose(api)
    dType.SetPTPCmdEx(api, 1, current_pose[0], current_pose[1], current_pose[2],
0, 1)
    current_pose = dType.GetPose(api)
    dType.SetPTPCmdEx(api, 2, colorX, colorY, homeZ, current_pose[3], 1)

```

```

current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, homeX, homeY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, homeX, preLainY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, lainX, lainY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, lainX, lainY, lainZ, current_pose[3], 1)
dType.SetEndEffectorSuctionCupEx(api, 0, 1)
dType.dSleep(1000)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, lainX, lainY, homeZ, current_pose[3], 1)
 kirimSinyalKubusLain()
  menungguSinyalConveyor3()
STEP_PER_CRICLE = 360.0 / 1.8 * 10.0 * 16.0
MM_PER_CRICLE = 3.1415926535898 * 36.0
vel = float(40) * STEP_PER_CRICLE / MM_PER_CRICLE
dType.SetEMotorEx(api, 0, 1, int(vel), 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, homeX, preLainY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, homeX, homeY, homeZ, current_pose[3], 1)
dType.dSleep(2500)
STEP_PER_CRICLE = 360.0 / 1.8 * 10.0 * 16.0
MM_PER_CRICLE = 3.1415926535898 * 36.0
vel = float(0) * STEP_PER_CRICLE / MM_PER_CRICLE

```



```
dType.SetEMotorEx(api, 0, 0, int(vel), 1)
```

```
def deteksiWarna():
```

```
    global homeX, homeY, homeZ, pickX, pickY, pickZ, colorX, colorY, colorZ,  
    hijauX, stepHijauX, hijauY, stepHijauY, hijauZAt, stepHijauZ
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, homeX, homeY, homeZ, current_pose[3], 1)
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, pickX, pickY, homeZ, current_pose[3], 1)
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, pickX, pickY, pickZ, current_pose[3], 1)
```

```
    dType.SetEndEffectorSuctionCupEx(api, 1, 1)
```

```
    dType.dSleep(1000)
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, pickX, pickY, homeZ, current_pose[3], 1)
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, colorX, colorY, homeZ, current_pose[3], 1)
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, colorX, colorY, colorZ, current_pose[3], 1)
```

```
    dType.dSleep(1000)
```

```
    if (dType.GetColorSensorEx(api, 1)) == 1:
```

```
        kirimSinyalKubusHijau()
```

```
        pickAndPlaceHijau()
```

```
        hijauX = hijauX + 30
```

```
        stepHijauX = stepHijauX + 1
```

```
        if stepHijauX == 2:
```

```

hijauX = 33.9745
stepHijauX = 0
hijauY = hijauY + 30
stepHijauY = stepHijauY + 1
if stepHijauY == 2:
    hijauY = 274.0255
    stepHijauY = 0
    hijauZAt = hijauZAt + 30
    stepHijauZ = stepHijauZ + 1
    if stepHijauZ == 2:
        hijauZAt = 16.979
        stepHijauZ = 0
else:
    pass

def menungguSinyalConveyor2():
    while (dType.GetIODI(api, 20)[0]) == 0:
        STEP_PER_CRICLE = 360.0 / 1.8 * 10.0 * 16.0
        MM_PER_CRICLE = 3.1415926535898 * 36.0
        vel = float(40) * STEP_PER_CRICLE / MM_PER_CRICLE
        dType.SetEMotorEx(api, 0, 1, int(vel), 1)
    STEP_PER_CRICLE = 360.0 / 1.8 * 10.0 * 16.0
    MM_PER_CRICLE = 3.1415926535898 * 36.0
    vel = float(0) * STEP_PER_CRICLE / MM_PER_CRICLE
    dType.SetEMotorEx(api, 0, 0, int(vel), 1)

```

```

dType.SetHOMECmdEx(api, 0, 1)
dType.SetIOMultiplexingEx(api, 19, 3, 1)
dType.SetIOMultiplexingEx(api, 20, 3, 1)
dType.SetIOMultiplexingEx(api, 7, 3, 1)
dType.SetIOMultiplexingEx(api, 4, 1, 1)
dType.SetIOMultiplexingEx(api, 7, 1, 1)
dType.SetEndEffectorParamsEx(api, 59.7, 0, 0, 1)
variabel()
while True:
    menungguSinyalInfraRed1()
    current_pose = dType.GetPose(api)
    dType.SetPTPCmdEx(api, 1, current_pose[0], current_pose[1], current_pose[2],
0, 1)
    deteksiWarna()
    menungguSinyalConveyor2()

```

3. Listing Program Robot 3

```

def variabel():
    global homeX, homeY, homeZ, pickX, pickY, pickZ, preY, placeX, placeY,
placeZ
    homeX = 200
    homeY = 0
    homeZ = 80
    pickX = 236.312
    pickY = 12.6766
    pickZ = -64.7932
    preY = 119.8329

```

```
placeX = 201.811
```

```
placeY = 210.9658
```

```
placeZ = 52.8422
```

```
def menungguSinyalInfraRed2():
```

```
    while not (dType.GetIODI(api, 19)[0]) == 0:
```

```
        pass
```

```
def gerak():
```

```
    global homeX, homeY, homeZ, pickX, pickY, pickZ, preY, placeX, placeY,  
    placeZ
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, homeX, homeY, homeZ, current_pose[3], 1)
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, pickX, pickY, homeZ, current_pose[3], 1)
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, pickX, pickY, pickZ, current_pose[3], 1)
```

```
    dType.SetEndEffectorSuctionCupEx(api, 1, 1)
```

```
    dType.dSleep(1000)
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, pickX, pickY, homeZ, current_pose[3], 1)
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, homeX, preY, homeZ, current_pose[3], 1)
```

```
    current_pose = dType.GetPose(api)
```

```
    dType.SetPTPCmdEx(api, 2, placeX, placeY, homeZ, current_pose[3], 1)
```

```
    current_pose = dType.GetPose(api)
```

```

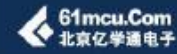
dType.SetPTPCmdEx(api, 2, placeX, placeY, placeZ, current_pose[3], 1)
dType.SetEndEffectorSuctionCupEx(api, 0, 1)
dType.dSleep(1000)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, placeX, placeY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, homeX, preY, homeZ, current_pose[3], 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 2, homeX, homeY, homeZ, current_pose[3], 1)

dType.SetHOMECmdEx(api, 0, 1)
dType.SetEndEffectorParamsEx(api, 59.7, 0, 0, 1)
dType.SetIOMultiplexingEx(api, 19, 3, 1)
current_pose = dType.GetPose(api)
dType.SetPTPCmdEx(api, 1, current_pose[0], current_pose[1], current_pose[2],
0, 1)
variabel()
while True:
    menungguSinyalInfraRed2()
    gerak()

```

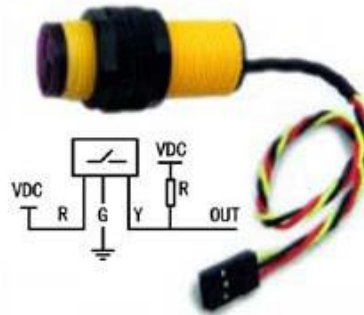
E18-D80NK-N

Adjustable Infrared Sensor Switch Manual



Introduction

This is an infrared distance switch. It has an adjustable detection range, 3cm - 80cm. It is small, easy to use/assemble, inexpensive. Useful for robot, interactive media, industrial assembly line, etc.



Specification

Model NO: E18-D80NK-N

Diameter: 18mm, Length: 45mm

Sensing range: 3-80cm adjustable

Appearance: Threaded cylindrical

Sensing object: Translucency, opaque

Material: Plastic

Supply voltage: DC5V

Guard mode: Reverse polarity protection

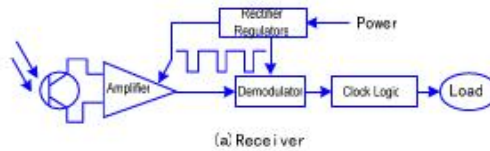
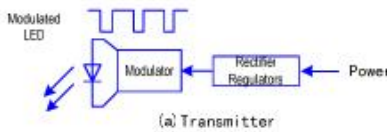
Load current: 100mA

Ambient temperature: -25-70°C

Output operation: Normally open(O)

Red: +5V; Yellow:Signal;Green:GND

Output: DC three-wire system(NPN)



Tel: 010-62669059
 Website: www.61mcu.com
 E-mail: fae_61mcu@163.com



Specifications- Dobot Magician 2.0 Arm



Introduction

The Dobot Magician is a revolutionary robot arm. It is very safe, easy to deploy, and versatile. The Dobot Magician 2.0 is only of its kind Desktop Robotic Arm.

Unlike traditional robots, the Magician is supplied with a lot of end-effectors bundled with it. This makes the deployment very fast. The controller inbuilt and software is freely available.

Further, it is open-source (SDK & API is freely available). It is perfect for a variety of second development and integration.

Available Versions & Accessories

Standard-

- Complete arm with power supply
- USB connectivity,
- Pneumatic Pick-n-place kit (Pump, suction gripper, 2-finger gripper)
- 3D Printing Kit
- Drawing Pen
- Auto-Leveling tool & essential Tool-kit

Advanced | Educational- Apart from all items from the standard version, following additional accessories are provided-

- Laser Engraver
- BlueTooth/WiFi connectivity modules
- Joystick



Specification for APC-817 Series

APC-817

DIP4, DC Input, Photo Transistor Coupler

The APC-817 series combines an AlGaAs infrared emitting diode as the emitter which is optically coupled to a silicon planar phototransistor detector in a plastic DIP4 package with different lead forming options.

With the robust coplanar double mold structure, APC-817 series provide the most stable isolation feature.



Features:

- High isolation V_{rms} : 5000V
- CTR flexibility available
- DC input with transistor output
- Operating temperature: - 55 °C to 110 °C
- RoHS & REACH Compliance
- MSL Class 1
- Halogen free (Optional)
- UL - UL1577
- VDE - EN60747-5-5(VDE0884-5)
- CQC - GB4943.1, GB8898
- cUL- CSA Component Acceptance Service Notice No. 5A

Applications:

- Switch mode power supplies
- Programmable controllers
- Household appliances
- Office equipment