

Hardware Project deployment and user manual

FLIR AX8 Thermal Camera



Product specifications: <https://www.flir.com.au/products/ax8-automation>

Product manual: <https://docs.rs-online.com/5bde/0900766b814343f3.pdf>

A-series feature matrix:

https://www.flir.com/globalassets/imported-assets/document/a-series-matrix_smarts.pdf

Raspberry Pi Overview:

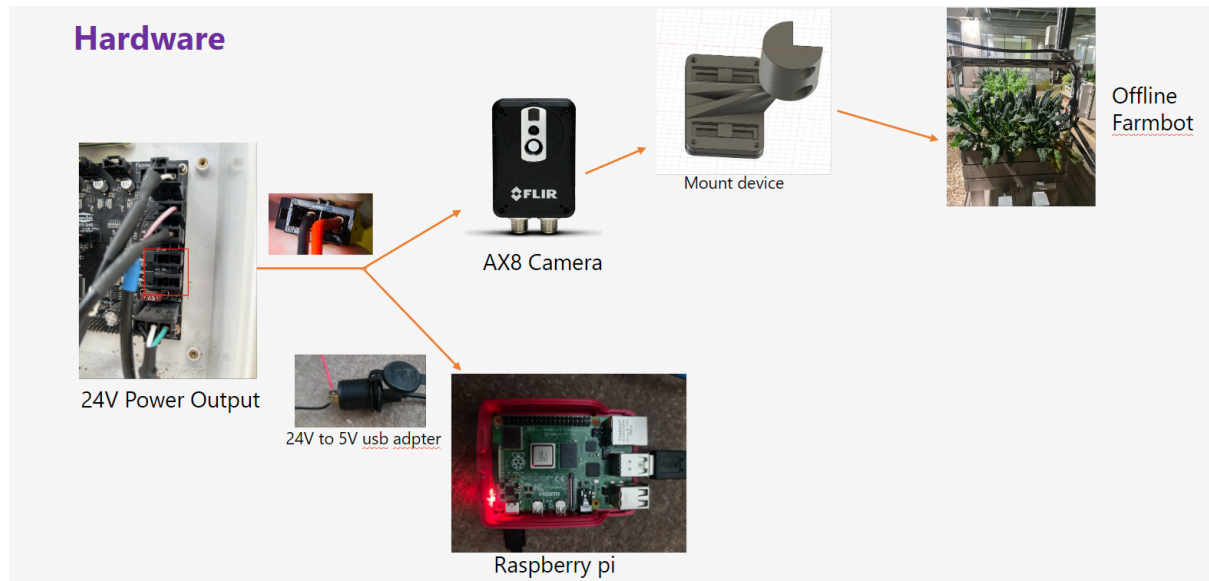
Using a Raspberry Pi 4 Model B Rev 1.5 with the Debian GNU/Linux 12 (Bookworm) operating system.

Username: farmbot

Password: farmbot

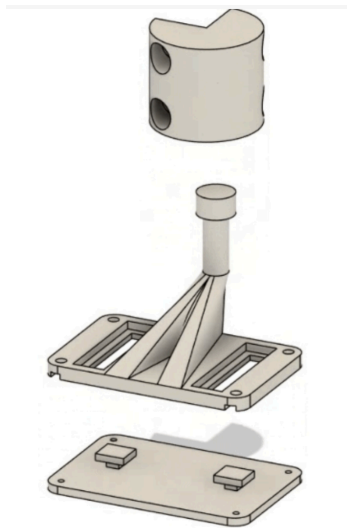
To connect the Pi to a PC, we can use VNC or the power cable for the Raspberry Pi display.

Hardware Integration solutions



Mounting:

Due to the insufficient compatibility of last year's bracket design with the actual FarmBot, we have redesigned a 3D-printed bracket that perfectly fits the FarmBot. Building on the original design, we have incorporated a rotatable feature and some vertical mobility, and improved the screw connection with the FarmBot's mechanical arm.



1. Modeling and File Conversion:

- We used CAD software for modeling and finally converted the model files into STL format for the 3D printing program.

2. 3D Printing Process:

- The bracket was printed using an FDM 3D printer in one piece, requiring no additional assembly or installation.
- The material used is basic PLA.
- More detailed information, including STL files and 3D printing slice files, will be included in the documentation.

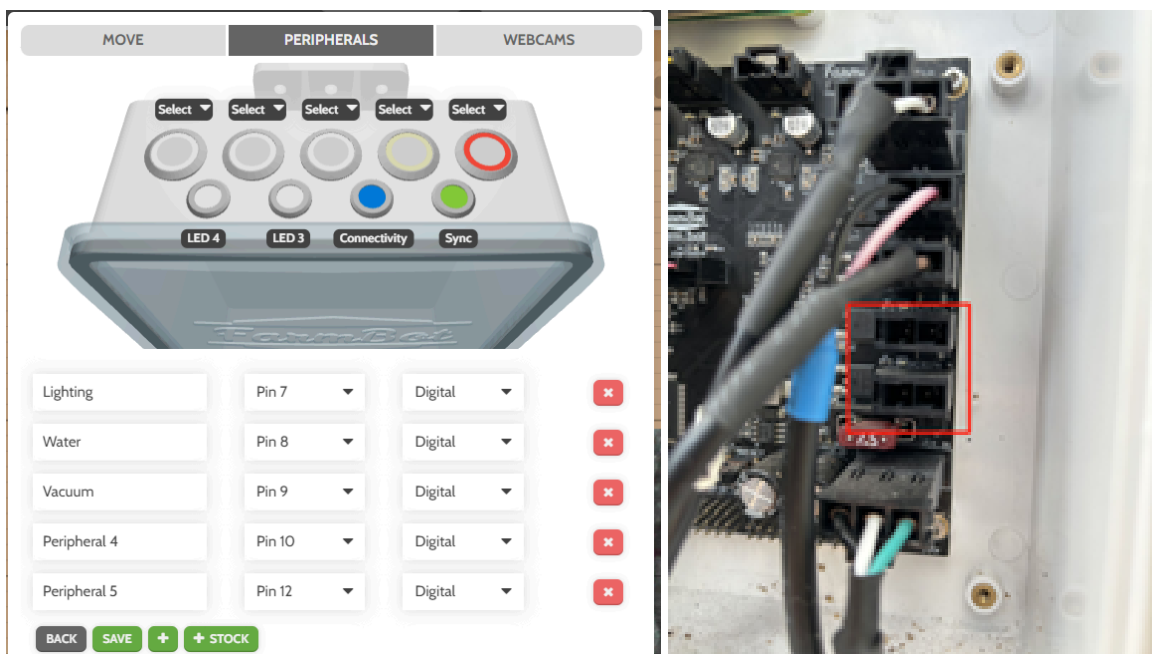
3. Bracket Performance:

- The printed bracket has a high degree of freedom, allowing for rotation and horizontal displacement.

Power Supply:

After investigating the official FLIR specifications, we found that the AX-8 operating voltage is 12-24V DC. Upon reviewing the official documentation, we determined the final solution: using the PWR/IO interface to power the camera and the ETH interface for data transmission (refer to pages 121-125 of the Product Manual PDF).

We have soldered the power and ground wires of the PWR/IO interface and integrated them into the special power interface of the FarmBot. The ETH interface can be connected to the Raspberry Pi for data transmission.



To ensure stable power supply to the camera, we need to adjust the voltage of the backup power supply in the FarmBot Web App. After testing, we found that only the currently unused backup water pump interface (pin8) on the FarmBot can provide power to the camera.

Additionally, to power the Raspberry Pi, we used a 24V to 5V transformer and successfully powered the Raspberry Pi through the power interface shown in the red box in Figure 2.

Steps to Assign a Fixed IP Address and Configure DHCP for the Camera

<https://www.flir.com.au/support-center/instruments2/how-do-i-connect-to-my-flir-ax8-to-a-computer-what-ip-address-should-i-use/>

Based on our investigation, we need to assign a fixed IP address to the camera and configure DHCP to complete the connection between the camera and the Raspberry Pi. The steps are as follows:

Step 1: Connect the Camera to the Raspberry Pi:

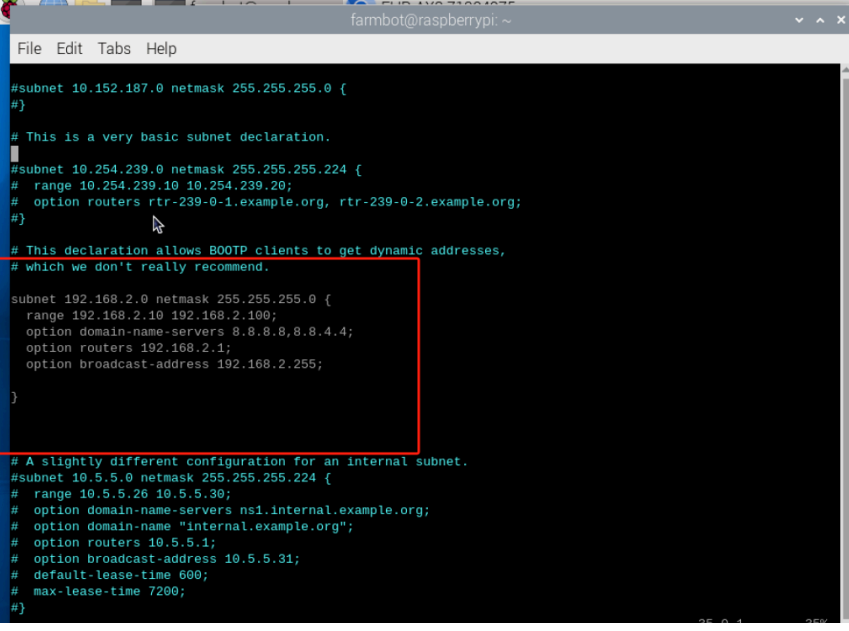
Connect the AX-8 camera to the Raspberry Pi using the ETH interface.

Step 2: Assign a Fixed IP Address:

Edit the DHCP configuration file on the Raspberry Pi to assign a fixed IP address to the camera.

Open the `dhcpd.conf` file for editing

```
sudo vim /etc/dhcp/dhcpd.conf
```



```
File Edit Tabs Help

#subnet 10.152.187.0 netmask 255.255.255.0 {
#}

# This is a very basic subnet declaration.

#subnet 10.254.239.0 netmask 255.255.255.224 {
# range 10.254.239.10 10.254.239.20;
# option routers rtr-239-0-1.example.org, rtr-239-0-2.example.org;
#}

# This declaration allows BOOTP clients to get dynamic addresses,
# which we don't really recommend.

subnet 192.168.2.0 netmask 255.255.255.0 {
    range 192.168.2.10 192.168.2.100;
    option domain-name-servers 8.8.8.8,8.8.4.4;
    option routers 192.168.2.1;
    option broadcast-address 192.168.2.255;
}

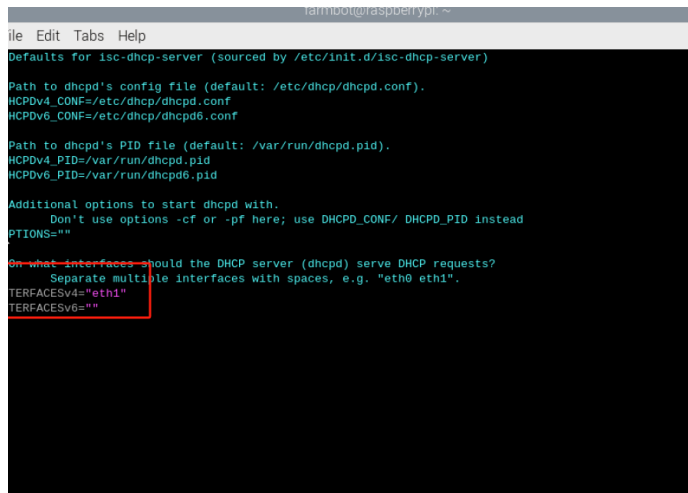
# A slightly different configuration for an internal subnet.
#subnet 10.5.5.0 netmask 255.255.255.224 {
# range 10.5.5.26 10.5.5.30;
# option domain-name-servers ns1.internal.example.org;
# option domain-name "internal.example.org";
# option routers 10.5.5.1;
# option broadcast-address 10.5.5.31;
# default-lease-time 600;
# max-lease-time 7200;
#}
```

step 3: Edit the ISC DHCP Server Configuration:

Open the ISC DHCP server configuration file for editing:

```
sudo vim /etc/default/isc-dhcp-server
```

Modify the configuration to specify the correct network interface, changing `INTERFACESv4` to `eth1` :

A screenshot of a terminal window showing the configuration file /etc/default/isc-dhcp-server being edited with vim. The terminal title is 'rahm001@aspen:~\$'. The editor shows the file's defaults, paths for configuration and PID files, and options for starting the service. A red box highlights the line 'INTERFACESv4="eth1"', which is being edited. The line above it is commented out: '# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?'. The line below it is 'INTERFACESv6=""'.

```
file Edit Tabs Help
Defaults for isc-dhcp-server (sourced by /etc/init.d/isc-dhcp-server)

Path to dhcpd's config file (default: /etc/dhcp/dhcpd.conf).
DHCPDv4_CONF=/etc/dhcp/dhcpd.conf
DHCPDv6_CONF=/etc/dhcp/dhcpd6.conf

Path to dhcpd's PID file (default: /var/run/dhcpd.pid).
DHCPDv4_PID=/var/run/dhcpd.pid
DHCPDv6_PID=/var/run/dhcpd6.pid

Additional options to start dhcpd with.
Don't use options -cf or -pf here; use DHCPD_CONF/ DHCPD_PID instead
OPTIONS=""

# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
# Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACESv4="eth1"
INTERFACESv6=""
```

step 4: Restart Networking and DHCP Services:

Restart the networking service:

```
sudo systemctl restart networking
```

Bring up the `eth1` interface:

```
sudo ip link set eth1 up
```

Restart the DHCP service to apply the changes:

```
sudo systemctl restart isc-dhcp-server
```

Remove the default route via `eth1` to avoid routing conflicts:

```
sudo ip route del default via 192.168.2.1 dev eth1
```

Step 5: Verify the IP Address:

Verify that the camera has been assigned the correct IP address

```
eth0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether d8:3a:dd:5a:fc:ee txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.2.2 netmask 255.255.255.0 broadcast 192.168.2.255
    inet6 fe80::ca78:7dff:febe:31ed prefixlen 64 scopeid 0x20<link>
    ether c8:78:7d:be:31:ed txqueuelen 1000 (Ethernet)
    RX packets 176588 bytes 236828605 (225.8 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 96069 bytes 12807213 (12.2 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 4485 bytes 350988 (342.7 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4485 bytes 350988 (342.7 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.132 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::9cb8:4a5e:e7cf:5978 prefixlen 64 scopeid 0x20<link>
    ether d8:3a:dd:5a:fc:ef txqueuelen 1000 (Ethernet)
    RX packets 337673 bytes 34072389 (32.4 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 494817 bytes 611322046 (583.0 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

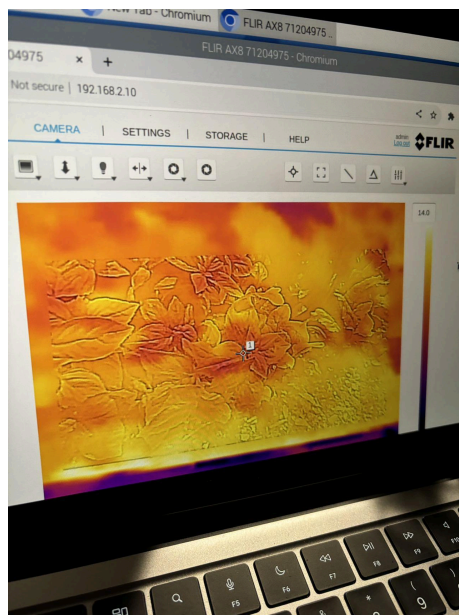
Through these steps, we can access the camera's services using the **eth1** address.

Plug and Play: Simplifying Camera Connections on Raspberry Pi with a Single Script

On our Raspberry Pi, the port settings are already configured. We only need to run `setup.sh` each time we start the Raspberry Pi and connect the camera.

```
File Edit Tabs Help
GNU nano 7.2                                setup.sh
#!/bin/bash#
sudo systemctl restart networking
sudo ip link set eth1 up
sudo systemctl restart isc-dhcp-server
sudo ip route del default via 192.168.2.1 dev eth1
./frp_0.48.0_linux_arm/frpc -c ./frp_0.48.0_linux_arm/frpc.ini
```

In addition to the regular camera domain services:



We also found the following API endpoints for our project:

<https://www.flir.com.au/support-center/instruments2/can-i-view-the-video-stream-from-the-ax8-without-the-user-interface-and-it-is-possible-to-view-the-stream-in-the-browser-without-logging-in/>

https://flir.custhelp.com/app/answers/detail/a_id/3602/~/getting-started-using-rest-api-with-au-tomation-cameras