

# 빠른 시작: MXCHIP AZ3166 DevKit를 IoT Hub에 연결

2021. 06. 09. • 읽는 데 28분 걸림 •  

## 이 문서의 내용

[사전 요구 사항](#)

[개발 환경 준비](#)

[클라우드 구성 요소 만들기](#)

[디바이스 준비](#)

[디바이스 속성 보기](#)

[원격 분석 보기](#)

[디바이스에서 직접 메서드 호출](#)

[문제 해결 및 디버그](#)

[리소스 정리](#)

[다음 단계](#)

**적용 대상:** 임베디드 디바이스 개발

**총 완료 시간:** 30분



이 빠른 시작에서는 Azure RTOS를 사용하여 MXCHIP AZ3166 IoT DevKit(이하 MXCHIP DevKit)를 Azure IoT에 연결합니다.

또한 IoT Explorer 및 IoT 플러그 앤 플레이를 사용하여 MXCHIP DevKit를 관리합니다. IoT 플러그 앤 플레이는 애플리케이션이 프로그래밍 방식으로 디바이스의 기능을 쿼리하고 상호 작용할 수 있도록 하는 개방형 디바이스 모델을 제공합니다. 디바이스는 이 모델을 사용하여 해당 기능을 IoT 플러그 앤 플레이 지원 애플리케이션에 브로드캐스트합니다. 이 모델을 사용하면 디바이스를 추가, 구성 및 관리하는 작업을 간소화하고 개선할 수 있습니다. 자세한 내용은 [IoT 플러그 앤 플레이 설명서](#)를 참조하세요.

다음 작업을 완료합니다.

- C에서 MXChip DevKit를 프로그래밍하기 위한 임베디드 개발 도구 세트 설치
- 이미지를 빌드하고 MXCHIP DevKit로 플래시
- Azure CLI를 사용하여 MXCHIP DevKit가 안전하게 연결할 Azure IoT Hub를 만들고 관리
- Azure IoT Explorer를 사용하여 IoT Hub에 디바이스를 등록하고, 디바이스 속성을 보고, 디바이스 원격 분석을 보고, 디바이스에서 직접 명령 호출

# 사전 요구 사항

- Microsoft Windows 10을 실행하는 PC
- Azure 구독이 아직 없는 경우 시작하기 전에 [무료 구독을 만듭니다](#) .
- 리포지토리를 복제하기 위한 [Git](#)
- Azure CLI. 이 빠른 시작에서 Azure CLI 명령을 실행하기 위한 두 가지 옵션이 있습니다.
  - 브라우저에서 CLI 명령을 실행하는 대화형 셸인 Azure Cloud Shell을 사용합니다. 이 옵션은 아무 것도 설치할 필요가 없으므로 권장됩니다. 처음으로 Cloud Shell을 사용하는 경우 [Azure Portal](#) 에 로그인합니다. [Cloud Shell 빠른 시작](#)의 단계를 따라 **Cloud Shell을 시작하고 Bash 환경을 선택합니다**.
  - 선택적으로 로컬 컴퓨터에서 Azure CLI를 실행합니다. Azure CLI가 이미 설치된 경우 `az upgrade`를 실행하여 CLI 및 확장을 현재 버전으로 업그레이드합니다. Azure CLI를 설치하는 방법은 [Azure CLI 2.0 설치](#)를 참조하세요.
- [Azure IoT Explorer](#) : Azure IoT 모니터링하고 관리하는 플랫폼 간 유틸리티
- 하드웨어
  - [MXCHIP AZ3166 IoT DevKit](#) (MXCHIP DevKit)
  - Wi-Fi 2.4GHz
  - USB 2.0 A~마이크로 USB 수 케이블


## 개발 환경 준비

개발 환경을 설정하려면 먼저 빠른 시작에 필요한 모든 자산이 포함되어 있는 GitHub 리포지토리를 복제합니다. 그런 다음, 프로그래밍 도구 세트를 설치합니다.

### 빠른 시작에 사용할 리포지토리를 복제합니다.

모든 샘플 디바이스 코드, 설치 스크립트 및 오프라인 버전 설명서를 다운로드하려면 다음 리포지토리를 복제합니다. 이전에 다른 빠른 시작에서 이 리포지토리를 복제한 경우 다시 복제할 필요가 없습니다.

리포지토리를 복제하려면 다음 명령을 실행합니다.

shell	 복사
<code>git clone --recursive https://github.com/azure-rtos/getting-started.git</code>	

## 도구 설치

복제된 리포지토리에는 필요한 도구를 설치하고 구성하는 설치 스크립트가 포함되어 있습니다. 다른 포함된 디바이스 빠른 시작에서 이 도구를 설치한 경우 다시 설치할 필요가 없습니다.

### ① 참고

설치 스크립트에서 설치하는 도구는 다음과 같습니다.


- **CMake** : 빌드
- **ARM GCC** : 컴파일
- **Termite** : 연결된 디바이스 리소스의 직렬 포트 출력 모니터링

도구를 설치하려면

1. 파일 탐색기에서 리포지토리의 다음 경로로 이동하여 *get-toolchain.bat* 이라는 설치 스크립트를 실행합니다.

*getting-started\tools\get-toolchain.bat*

2. 설치 후 새 콘솔 창을 열어 설치 스크립트에서 변경한 구성을 인식합니다. 이 콘솔을 사용하여 빠른 시작의 나머지 프로그래밍 작업을 완료합니다. Windows용 Windows CMD, PowerShell 또는 Git Bash를 사용할 수 있습니다.
3. 다음 코드를 실행하여 CMake 버전 3.14 이상이 설치되어 있는지 확인합니다.

shell	 복사
cmake --version	

## 클라우드 구성 요소 만들기

### IoT Hub 만들기

Azure CLI를 사용하여 디바이스에 대한 이벤트 및 메시징을 처리하는 IoT Hub를 만들 수 있습니다.

IoT Hub를 만듭니다.

1. CLI 앱을 시작합니다. 이 빠른 시작의 나머지 부분에서 CLI 명령을 실행하려면 명령 구문을 복사하고 CLI 애플리케이션에 붙여넣은 후 변수 값을 편집하고 Enter 키를

누릅니다.

- 또는 Cloud Shell을 사용하려면 [Cloud Shell](#)에 대한 링크를 마우스 오른쪽 단추로 클릭하고 새 탭에서 열기 옵션을 선택합니다.
- Azure CLI를 로컬에서 사용하고 있는 경우 CLI 콘솔 앱을 시작하고 Azure CLI에 로그인합니다.


2. `az extension add`를 실행하여 `azure-iot` 확장을 현재 버전으로 설치하거나 업그레이드합니다.

Azure CLI	 복사	 사용해 보세요.
<pre>az extension add --upgrade --name azure-iot</pre>		

3. `az group create`를 실행하여 리소스 그룹을 만듭니다. 다음 명령을 사용하면 `centralus` 지역에 `MyResourceGroup`이라는 리소스 그룹을 만듭니다.

#### ❗ 참고


선택적으로 대체 `location`을 설정할 수 있습니다. 사용 가능한 위치를 보려면 `az account list-locations`를 실행합니다.

Azure CLI	 복사
<pre>az group create --name MyResourceGroup --location centralus</pre>	

4. `az iot hub create` 명령을 사용하여 IoT Hub를 만듭니다. IoT Hub를 만드는 데 몇 분 정도 걸릴 수 있습니다.

`YourIotHubName`. 이 자리 표시자를 IoT 허브용으로 선택한 이름으로 바꿉니다. IoT Hub 이름은 Azure에서 전역적으로 고유해야 합니다. 이 자리 표시자는 이 빠른 시작의 나머지 부분에서 고유한 IoT Hub 이름을 표시하는 데 사용됩니다.

`--sku F1` 매개 변수는 무료 계층에 IoT Hub를 만듭니다. 무료 계층 허브에는 제한된 기능 세트가 있으며 개념 증명 애플리케이션에 사용됩니다. IoT Hub 계층, 기능 및 가격 책정에 대한 자세한 내용은 [Azure IoT Hub 가격 책정](#)을 참조하세요.

Azure CLI	 복사
<pre>az iot hub create --resource-group MyResourceGroup --name {YourIoTHubName} --sku F1 --partition-count 2</pre>	

- IoT Hub를 만든 후 콘솔에서 JSON 출력을 보고, 이후 단계에서 사용할 `hostName` 값을 복사합니다. `hostName` 값은 다음 예제와 같습니다.


```
{Your IoT hub name}.azure-devices.net
```

## IoT Explorer 구성

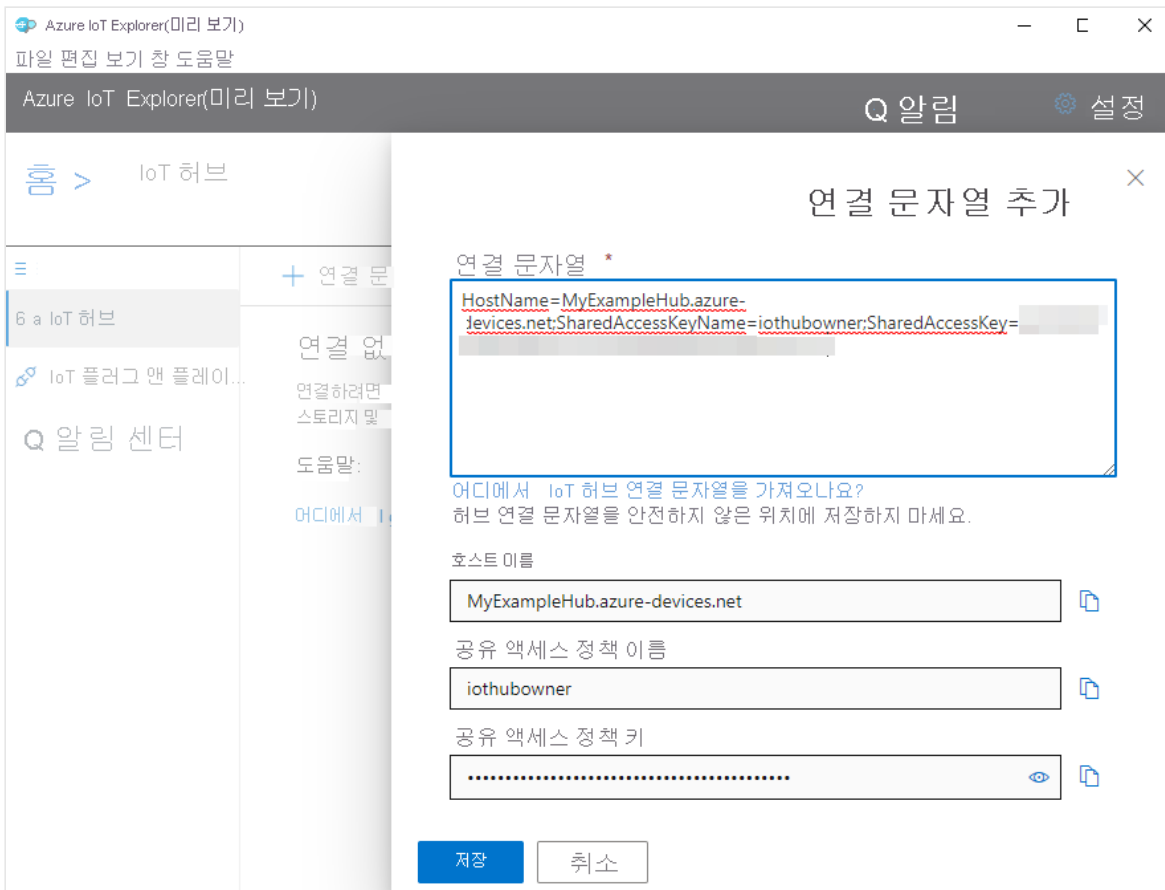
이 빠른 시작의 나머지 단계에서는 IoT Explorer를 사용하여 IoT Hub에 디바이스를 등록하고, 디바이스 속성 및 원격 분석을 보고, 디바이스에 명령을 보냅니다. 이 섹션에서는 방금 만든 IoT Hub에 연결하고 퍼블릭 모델 리포지토리에서 플러그 앤 플레이 모델을 로드하도록 IoT Explorer를 구성합니다.

IoT Hub에 대한 연결을 추가하려면

- CLI 앱에서 `az iot hub connection-string show` 명령을 실행하여 IoT Hub에 대한 연결 문자열을 가져옵니다.

Azure CLI	 복사
<pre>az iot hub connection-string show --hub-name {YourIoTHubName}</pre>	

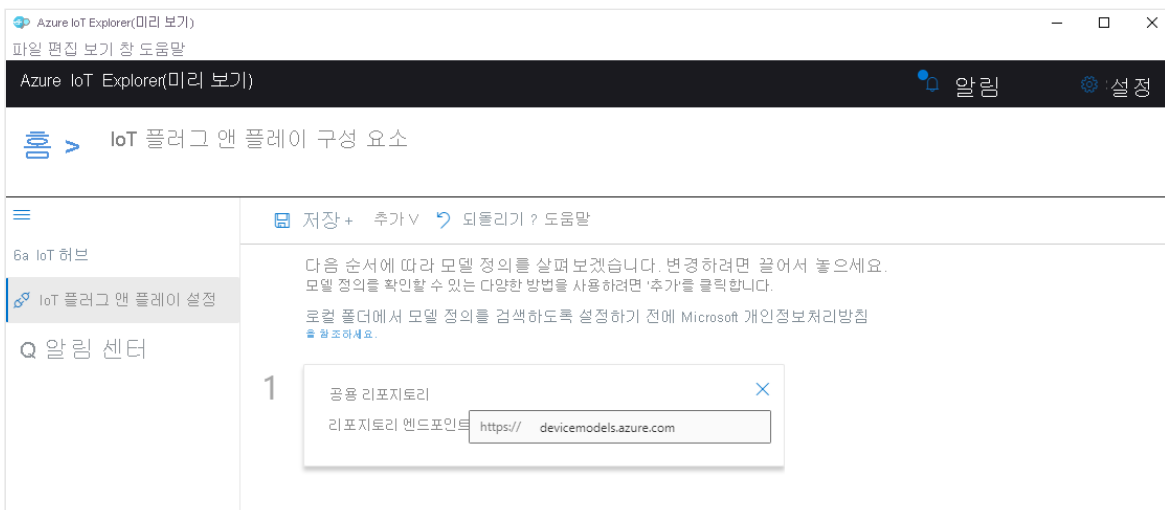
- 따옴표 문자로 묶지 말고 연결 문자열을 복사합니다.
- Azure IoT Explorer의 왼쪽 메뉴에서 **IoT Hub**를 선택한 다음, **+ 연결 추가**를 선택합니다.
- 연결 문자열을 **연결 문자열** 상자에 붙여넣습니다.
- 저장**을 선택합니다.



6. 연결에 성공하면 IoT Explorer가 **디바이스** 보기로 전환됩니다.

퍼블릭 모델 리포지토리를 추가하려면

1. IoT Explorer에서 **홈** 을 선택하여 홈 보기로 돌아갑니다.
2. 왼쪽 메뉴에서 **IoT 플러그 앤 플레이 설정** 을 선택한 다음, **+추가** 를 선택하고 드롭 다운 메뉴에서 **퍼블릭 리포지토리** 를 선택합니다.
3. <https://devicemodels.azure.com>에 있는 퍼블릭 모델 리포지토리에 대한 항목이 나타납니다.



4. **저장** 을 선택합니다.

## 디바이스 등록

이 섹션에서는 새 디바이스 인스턴스를 만들고 만든 IoT Hub에 등록합니다. 이후 섹션에서 새로 등록한 디바이스에 대한 연결 정보를 사용하여 물리적 디바이스를 안전하게 연결합니다.

디바이스를 등록하려면

1. IoT Explorer의 홈 보기에서 **IoT Hub** 를 선택합니다.
2. 이전에 추가한 연결이 표시됩니다. 연결 속성 아래에서 **이 허브의 디바이스 보기** 를 선택합니다.
3. + **새로 만들기** 를 선택하고 디바이스의 디바이스 ID(예: *mydevice*)를 입력합니다. 다른 모든 속성은 동일하게 유지합니다.
4. **만들기** 를 선택합니다.

Azure IoT Explorer(미리 보기)

파일 편집 보기 창 도움말

Azure IoT Explorer(미리 보기) 알림 to 설정

홈 > MyExampleHub > 디바이스 > mydevice > 디바이스 ID

저장키 관리

디바이스 ID

디바이스 ID ⓘ

mydevice

기본 키 ⓘ

.....

보조 키 ⓘ

.....

기본 연결 문자열 ⓘ

.....

보조 연결 문자열 ⓘ

.....

✓ SAS 토큰을 사용하는 연결 문자열 ⓘ

IoT 허브에 이 디바이스 연결

☒ 사용

5. 복사 단추를 사용하여 **디바이스 ID** 및 **기본 키** 필드를 복사하고 적어 둡니다.

다음 섹션을 계속 진행하기 전에 다음 값을 복사했는지 확인합니다.

- `hostName`
- `deviceId`
- `primaryKey`

## 디바이스 준비


MXCHIP DevKit를 Azure에 연결하려면 Wi-Fi 및 Azure IoT 설정에 대한 구성 파일을 수정하고, 이미지를 다시 빌드하고, 해당 이미지를 디바이스로 플래시합니다.

## 구성 추가

1. 텍스트 편집기에서 다음 파일을 엽니다.

`getting-started\MXChip\AZ3166\app\azure_config.h`

2. 표시된 것처럼 파일 위쪽에 있는 다음 줄을 주석으로 처리합니다.

C	 복사
<pre>// #define ENABLE_DPS</pre>	

3. Wi-Fi 상수를 로컬 환경의 다음 값으로 설정합니다.

상수 이름	값
WIFI_SSID	{사용자의 Wi-Fi SSID}
WIFI_PASSWORD	{사용자의 Wi-Fi 암호}
WIFI_MODE	{파일에 열거된 Wi-Fi 모드 값 중 하나}

4. Azure IoT 디바이스 정보 상수를 Azure 리소스를 만든 후에 저장한 값으로 설정합니다.

상수 이름	값
IOT_HUB_HOSTNAME	{IoT Hub 호스트 이름 값}
IOT_DPS_REGISTRATION_ID	{사용자의 디바이스 ID 값}
IOT_DEVICE_SAS_KEY	{사용자의 기본 키 값}

5. 파일을 저장하고 닫습니다.



## 이미지 빌드

1. 콘솔 또는 파일 탐색기에서 다음 경로에 있는 *rebuild.bat* 스크립트를 실행하여 이미지를 빌드합니다.

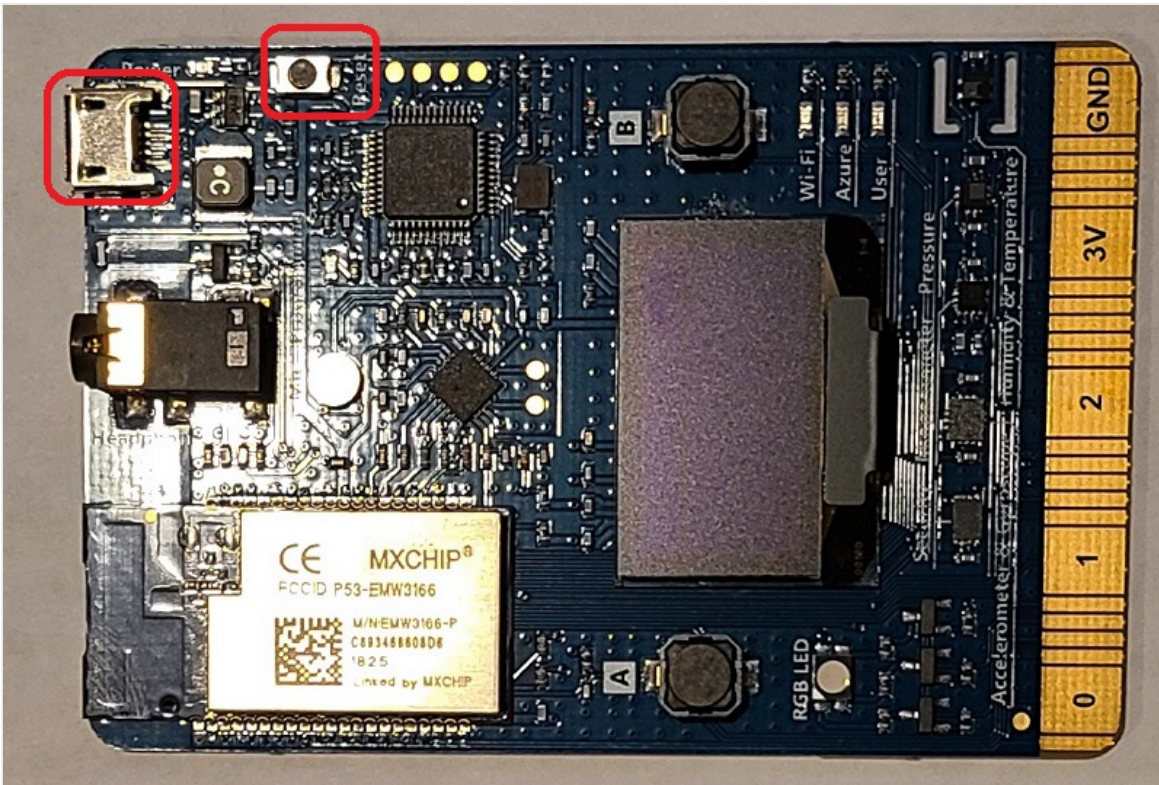
```
getting-started\MXChip\AZ3166\tools\rebuild.bat
```

2. 빌드가 완료되면 이진 파일이 다음 경로에 만들어졌는지 확인합니다.

```
getting-started\MXChip\AZ3166\build\app\mxchip_azure_iot.bin
```

## 이미지 플래시

1. MXCHIP DevKit에서 **다시 설정** 단추 및 마이크로 USB 포트를 찾습니다. 이러한 구성 요소는 다음 단계에서 사용합니다. 다음 그림에는 두 가지 구성 요소가 모두 강조 표시되어 있습니다.



2. 마이크로 USB 케이블을 MXCHIP DevKit의 마이크로 USB 포트에 연결한 다음, 컴퓨터에 연결합니다.
3. 파일 탐색기에서 이전 섹션에서 만든 이진 파일을 찾습니다.
4. *mxchip\_azure\_iot.bin* 이진 파일을 복사합니다.
5. 파일 탐색기에서 컴퓨터에 연결된 MXCHIP DevKit 디바이스를 찾습니다. 디바이스는 드라이브 레이블이 **AZ3166** 인 드라이브로 시스템에 표시됩니다.

6. 이진 파일을 MXCHIP DevKit의 루트 폴더에 붙여넣습니다. 깜박임이 자동으로 시작되고 몇 초 후에 완료됩니다.

### ❗ 참고

깜박임 프로세스 중에 녹색 LED에서 MXCHIP DevKit를 켵니다.

## 디바이스 연결 세부 정보 확인

Termite 앱을 사용하여 통신을 모니터링하고 디바이스가 올바르게 설정되었는지 확인할 수 있습니다.

1. Termite 를 시작합니다.

### 💡 팁

Termite를 DevKit에 연결할 수 없는 경우 **ST-LINK 드라이버** 를 설치하고 다시 시도합니다. 추가 단계는 **문제 해결**을 참조하세요.

2. **설정** 을 선택합니다.
3. **직렬 포트 설정** 대화 상자에서 다음 설정을 확인하고, 필요한 경우 업데이트합니다.
  - **전송 속도**: 115,200
  - **포트**: MXCHIP DevKit가 연결된 포트입니다. 드롭다운에 여러 포트 옵션이 있는 경우 사용할 올바른 포트를 찾을 수 있습니다. Windows **디바이스 관리자**를 열고, **포트** 를 확인하여 사용할 포트를 식별합니다.

직렬 포트 설정

<b>포트 구성</b> 포트: COM3 전송 속도: 115200 데이터 비트: 8 검지 비트: 1 Parity: 없음 흐름 제어: 없음 전달: 없음		<b>전송 텍스트</b> <input type="radio"/> 추가 안 함 <input type="radio"/> CR 추가 <input checked="" type="radio"/> LF 추가 <input type="radio"/> CR-LF 추가 <input checked="" type="checkbox"/> 로컬 에코	<b>옵션</b> <input type="checkbox"/> 맨 위에 유지 <input checked="" type="checkbox"/> Esc를 누르면 종료 <input checked="" type="checkbox"/> 편집 줄 자동 완성 <input checked="" type="checkbox"/> 기록 보관 <input type="checkbox"/> 사용하지 않을 때 포트 닫기
		<b>수신 텍스트</b> 폴링: 100 ms 최대 줄 수: 글꼴: 기본값 <input type="checkbox"/> 자동 줄 바꿈	<b>플러그인</b> <input type="checkbox"/> 자동 수신 <input type="checkbox"/> 기능 키 <input type="checkbox"/> 16진수 보기 <input type="checkbox"/> 강조 표시 로그 파일

사용자 인터페이스 언어: 한국어(ko) 취소 확인

4. **확인**을 선택합니다.

5. 디바이스의 **다시 설정** 단추를 누릅니다. 이 단추는 디바이스에 대한 레이블을 표시하고 마이크로 USB 커넥터 근처에 있습니다.
6. **Termite** 앱에서 다음 검사점 값을 확인하여 디바이스가 초기화되고 Azure IoT에 연결되었는지 확인합니다.

출력	복사
<pre>Starting Azure thread  Initializing WiFi   MAC address: C8:93:46:8A:4C:43   Connecting to SSID 'iot' SUCCESS: WiFi connected to iot  Initializing DHCP   IP address: 192.168.0.18   Mask: 255.255.255.0   Gateway: 192.168.0.1 SUCCESS: DHCP initialized  Initializing DNS client   DNS address: 75.75.75.75 SUCCESS: DNS client initialized  Initializing SNTP client   SNTP server 0.pool.ntp.org   SNTP IP address: 157.245.166.169   SNTP time update: Jun 8, 2021 18:16:50.807 UTC SUCCESS: SNTP initialized  Initializing Azure IoT Hub client   Hub hostname: *.azure-devices.net   Device id: mydevice   Model id: dtmi:azurertos:devkit:gsgmxchip;1 Connected to IoT Hub SUCCESS: Azure IoT Hub client initialized</pre>	

다음 단계에서 디바이스 출력을 모니터링하려면 Termite를 열어 둡니다.

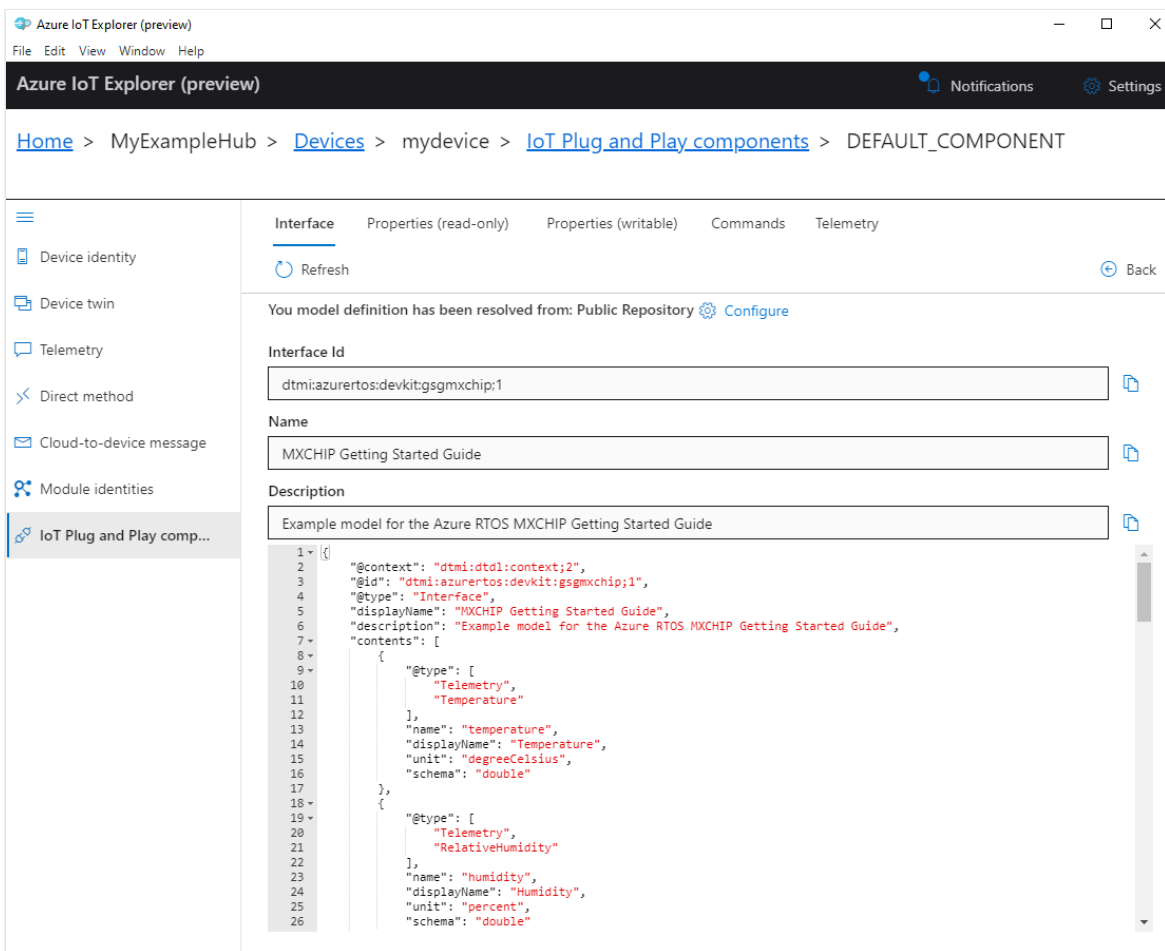
## 디바이스 속성 보기

Azure IoT Explorer를 사용하여 디바이스의 속성을 보고 관리할 수 있습니다. 이 섹션 및 다음 섹션에서는 IoT Explorer에 표시되는 플러그 앤 플레이 기능을 사용하여 MXCHIP DevKit를 관리하고 상호 작용합니다. 이러한 기능은 퍼블릭 모델 리포지토리에서 MXCHIP DevKit에 대해 게시된 디바이스 모델을 사용합니다. 이 빠른 시작의 앞부분에서는 이 리포지토리에서 디바이스 모델을 검색하도록 IoT Explorer를 구성했습니다. 대부분의 경우에는 IoT Explorer에 있는 디바이스 창의 왼쪽 메뉴에서 동일한 작업을 선택하여 플러그 앤 플레이를 사용하지 않고도 동일한 작업을 수행할 수 있습니다. 그러나 플러그 앤 플레이를 사용하면 개선된 환경을 얻을 수 있습니다. IoT Explorer는 플러그 앤 플레이

디바이스에서 지정된 디바이스 모델을 읽고 해당 디바이스와 관련된 정보를 제공할 수 있기 때문입니다.

IoT Explorer에서 디바이스에 대한 IoT 플러그 앤 플레이 구성 요소에 액세스하려면

1. IoT Explorer의 홈 보기에서 **IoT Hub** 를 선택한 다음, **이 허브의 디바이스 보기** 를 선택합니다.
2. 디바이스를 선택합니다.
3. **IoT 플러그 앤 플레이 구성 요소** 를 참조하세요.
4. **기본 구성 요소** 를 선택합니다. IoT Explorer에는 디바이스에 구현된 IoT 플러그 앤 플레이 구성 요소가 표시됩니다.



5. **인터페이스** 탭에서 디바이스 모델 **설명** 에 있는 JSON 콘텐츠를 확인합니다. JSON에는 디바이스 모델의 각 IoT 플러그 앤 플레이 구성 요소에 대한 구성 정보가 포함되어 있습니다.

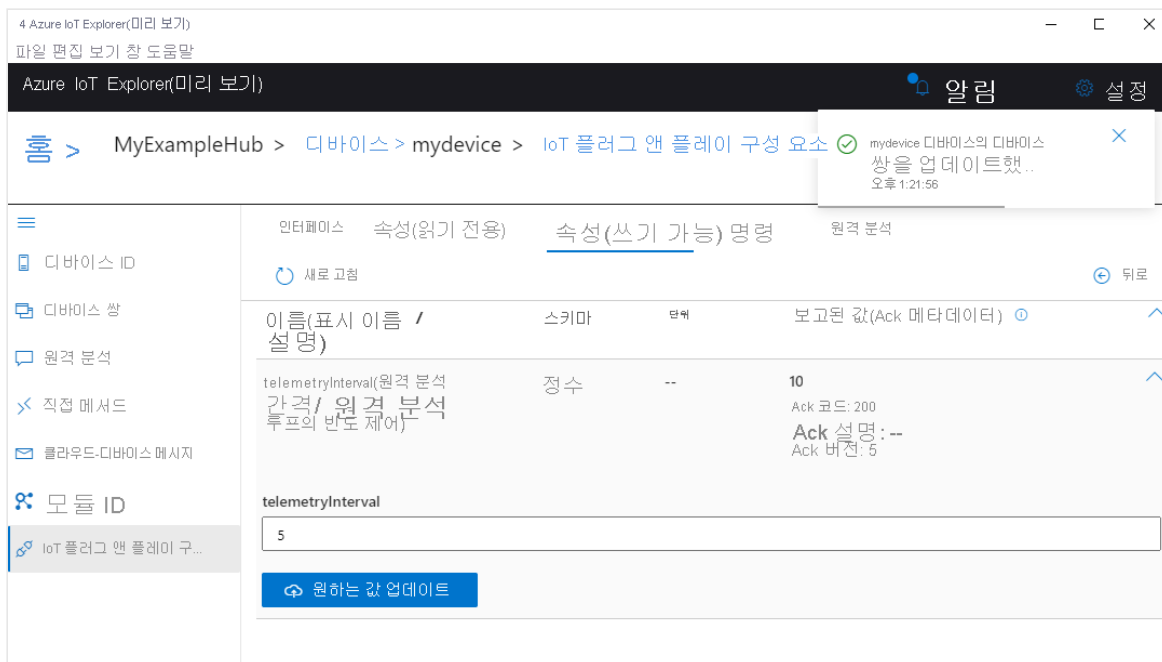
IoT Explorer의 각 탭은 디바이스 모델의 IoT 플러그 앤 플레이 구성 요소 중 하나에 해당합니다.

탭	Type	Name	Description
---	------	------	-------------

탭	Type	Name	Description
인터페이스	인터페이스	MXCHIP Getting Started Guide	MXCHIP DevKit에 대한 예제 모델
속성(읽기 전용)	속성	--	이 모델에는 현재 읽기 전용 속성이 없습니다.
속성(쓰기 가능)	속성	telemetryInterval	디바이스에서 원격 분석을 전송하는 간격
명령	명령	setLedState	LED 켜기 또는 끄기
원격 분석	원격 분석	temperature	섭씨 온도

Azure IoT Explorer를 사용하여 디바이스 속성을 보려면

1. **속성(읽기 전용)** 탭을 선택합니다. 현재, 디바이스 모델에 의해 노출되는 읽기 전용 속성은 없습니다.
2. **속성(쓰기 가능)** 탭을 선택합니다. 원격 분석이 전송되는 간격이 표시됩니다.
3. telemetryInterval을 5로 변경한 후 **원하는 값 업데이트**를 선택합니다. 이제 디바이스에서 이 간격을 사용하여 원격 분석을 보냅니다.



4. IoT Explorer는 알림으로 응답합니다. Termite에서 업데이트를 확인할 수도 있습니다.
5. 원격 분석 간격을 10으로 다시 설정합니다.

Azure CLI를 사용하여 디바이스 속성을 보려면

1. `az iot hub device-identity show` 명령을 실행합니다.

Azure CLI	 복사
<pre>az iot hub device-identity show --device-id mydevice --hub-name {YourIoTHubName}</pre>	

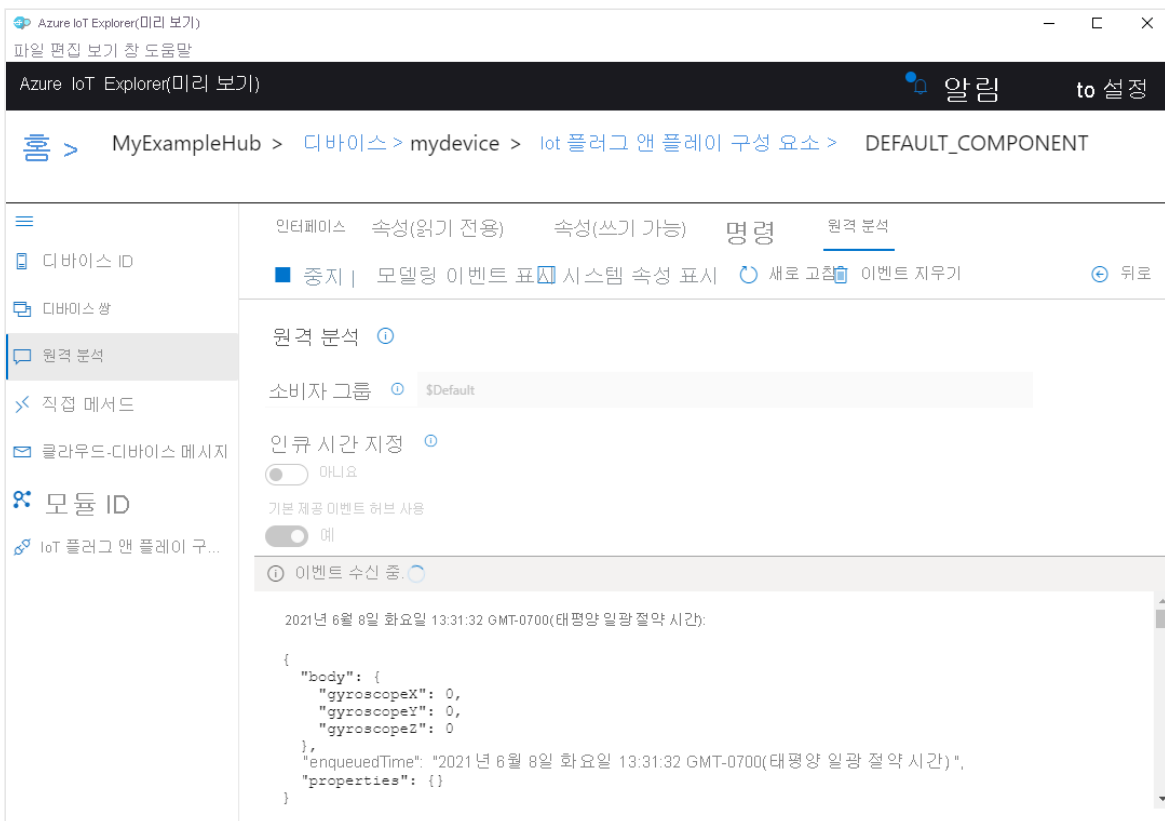
2. 콘솔 출력에서 디바이스의 속성을 검사합니다.

## 원격 분석 보기

Azure IoT Explorer를 사용하면 디바이스에서 클라우드로의 원격 분석 흐름을 볼 수 있습니다. 필요에 따라 Azure CLI를 사용하여 동일한 작업을 수행할 수 있습니다.

Azure IoT Explorer에서 원격 분석을 보려면

1. IoT Explorer의 디바이스에 대한 **IoT 플러그 앤 플레이 구성 요소**(기본 구성 요소) 창에서 **원격 분석** 탭을 선택합니다. 기본 제공 이벤트 허브 사용 이 예 로 설정되어 있는지 확인합니다.
2. **시작** 을 선택합니다.
3. 디바이스에서 메시지를 클라우드에 보낼 때 원격 분석을 봅니다.

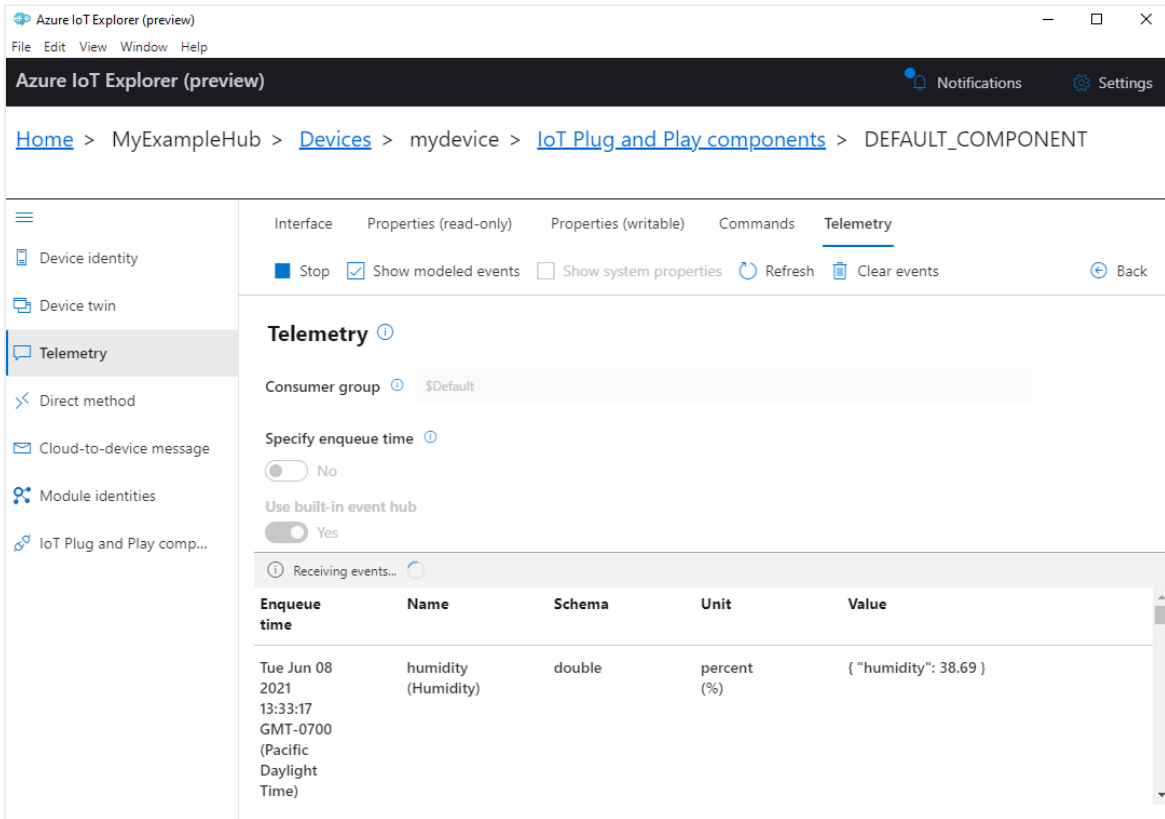


The screenshot shows the Azure IoT Explorer web application. The breadcrumb navigation is: 홈 > MyExampleHub > 디바이스 > mydevice > IoT 플러그 앤 플레이 구성 요소 > DEFAULT\_COMPONENT. The '원격 분석' (Remote Analysis) tab is selected in the top navigation bar. On the left sidebar, '원격 분석' is also highlighted. The main content area shows the '원격 분석' section with a '소버자 그룹' (Owner Group) of '\$Default'. Below this, there are settings for '인큐 시간 지정' (Incubation Time) set to '아니오' (No) and '기본 제공 이벤트 허브 사용' (Use Default Event Hub) set to '예' (Yes). The '이벤트 수신 중' (Receiving Events) status is '중지' (Stopped). A sample event message is displayed, showing a JSON payload with gyroscope data and a timestamp.

## ❗ 참고

Termite 앱을 사용하여 디바이스에서 원격 분석을 모니터링할 수도 있습니다.

- 디바이스 모델에서 지정한 데이터 형식으로 이벤트를 보려면 **모델링된 이벤트 표시 확인란**을 선택합니다.



- 중지** 를 선택하여 수신 이벤트를 종료합니다.

Azure CLI를 사용하여 디바이스 원격 분석을 보려면

- `az iot hub monitor-events` 명령을 실행합니다. 이전에 디바이스 및 IoT Hub에 대해 Azure IoT에서 만든 이름을 사용합니다.

Azure CLI	복사
<pre>az iot hub monitor-events --device-id mydevice --hub-name {YourIoTHubName}</pre>	

- 콘솔에서 JSON 출력을 봅니다.

JSON	복사
<pre>{   "event": {     "origin": "mydevice",     "module": "",</pre>	

```

"interface": "dtmi:azureertos:devkit:gsgmxchip;1",
"component": "",
"payload": "
{ \"humidity\":41.21, \"temperature\":31.37, \"pressure\":1005.18 }
}

```

3. Ctrl+C를 선택하여 모니터링을 종료합니다.

## 디바이스에서 직접 메서드 호출

Azure IoT Explorer를 사용하여 디바이스에서 구현한 직접 메서드를 호출할 수도 있습니다. 직접 메서드에는 이름이 있으며, 필요에 따라 JSON 페이로드, 구성 가능한 연결 및 메서드 시간 제한이 있을 수 있습니다. 이 섹션에서는 LED를 설정하거나 해제하는 메서드를 호출합니다. 필요에 따라 Azure CLI를 사용하여 동일한 작업을 수행할 수 있습니다.

Azure IoT 탐색기에서 메서드를 호출하려면

1. IoT Explorer의 디바이스에 대한 **IoT 플러그 앤 플레이 구성 요소**(기본 구성 요소) 창에서 **명령** 탭을 선택합니다.
2. **setLedState** 명령의 경우 **상태** 를 **true** 로 설정합니다.
3. **명령 보내기** 를 선택합니다. IoT Explorer에 알림이 표시되고 디바이스의 노란색 사용자 LED 표시등이 켜집니다.

The screenshot shows the Azure IoT Explorer (preview) interface. The breadcrumb navigation is: Home > MyExampleHub > Devices > mydevice > IoT Plug and Play components. The left sidebar lists various options, with 'IoT Plug and Play components' selected. The main area shows the 'Commands' tab for the 'mydevice' device. It lists two commands: 'setLedState (Set LED state / Sets the state of the onboard LED.)' and 'setDisplayText (Display Text / Display text on screen.)'. The 'setLedState' command is selected, and the 'true' radio button is chosen. A 'Send command' button is visible. A notification bubble indicates the command was successfully invoked: 'Successfully invoked the IoT Plug and Play command 'setLedState' on device 'mydevice': {\"status\":200,\"payload\":{}}. 1:45:25 PM'.


Name (Display Name / Description)	Request schema	Response schema	Command type
setLedState (Set LED state / Sets the state of the onboard LED.)	boolean	--	--
state			
<input checked="" type="radio"/> true			
<input type="radio"/> false			
<input type="button" value="Send command"/>			
setDisplayText (Display Text / Display text on screen.)	string	--	--
text			
<input type="text"/>			
<input type="button" value="Send command"/>			




4. **상태** 를 **false** 로 설정한 다음, **명령 보내기** 를 선택합니다. 노란색 사용자 LED 표시 등이 꺼집니다.
5. 필요에 따라 Termite에서 출력을 확인하여 메서드의 상태를 모니터링할 수 있습니다.

Azure CLI를 사용하여 메서드를 호출하려면


1. `az iot hub invoke-device-method` 명령을 실행하고 메서드 이름과 페이로드를 지정합니다. 이 메서드의 경우 LED를 켜려면 `method-payload` 를 `true` 로 설정하고, LED를 끄려면 `false` 로 설정합니다.

Azure CLI	 복사
<pre>az iot hub invoke-device-method --device-id mydevice --method-name setLedState --method-payload true --hub-name {YourIoTHubName}</pre>	

CLI 콘솔은 디바이스에서 메서드 호출의 상태를 표시합니다. 여기서 204 는 성공을 나타냅니다.

JSON	 복사
<pre>{   "payload": {},   "status": 200 }</pre>	

2. 디바이스에서 LED 상태를 확인합니다.
3. Termite 터미널을 보고 출력 메시지를 확인합니다.

출력	 복사
<pre>Receive direct method: setLedState   Payload: true LED is turned ON Device twin property sent: {"ledState":true}</pre>	

## 문제 해결 및 디버그

디바이스 코드 빌드, 디바이스 플래시 또는 연결에 문제가 발생하는 경우 [문제 해결](#)을 참조하세요.

애플리케이션 디버깅에 대한 자세한 내용은 [Visual Studio Code를 사용하여 디버깅](#) 을 참조하세요.

# 리소스 정리

이 빠른 시작에서 만든 Azure 리소스가 더 이상 필요하지 않은 경우 Azure CLI를 사용하여 리소스 그룹 및 모든 리소스를 삭제할 수 있습니다.

## ❗ 중요

리소스 그룹을 삭제하면 다시 되돌릴 수 없습니다. 리소스 그룹 및 그 안에 포함된 모든 리소스가 영구적으로 삭제됩니다. 잘못된 리소스 그룹 또는 리소스를 자동으로 삭제하지 않도록 해야 합니다.

리소스 그룹을 이름으로 삭제하려면:

1. `az group delete` 명령을 실행합니다. 그러면 만든 리소스 그룹, IoT Hub 및 디바이스 등록이 제거됩니다.

Azure CLI

복사

사용해 보세요.

```
az group delete --name MyResourceGroup
```

2. `az group list` 명령을 실행하여 리소스 그룹을 삭제했는지 확인합니다.

Azure CLI

복사

사용해 보세요.

```
az group list
```

## 다음 단계

이 빠른 시작에서는 Azure RTOS 샘플 코드가 포함된 사용자 지정 이미지를 빌드한 다음, 해당 이미지를 MXCHIP DevKit 디바이스로 플래시했습니다. 또한 Azure CLI 및/또는 IoT Explorer를 사용하여 Azure 리소스를 만들고, MXCHIP DevKit를 Azure에 안전하게 연결하고, 원격 분석을 보고, 메시지를 보냈습니다.

다음 단계로, 다음 문서에서 IoT 디바이스 SDK를 사용하여 디바이스를 Azure IoT 연결하는 방법을 자세히 알아봅니다.

IoT Central에 MXCHIP AZ3166 devkit 연결

IoT Hub에 시뮬레이션된 디바이스 연결

IoT Hub에 시뮬레이션된 디바이스 연결

**① 중요**

Azure RTOS는 기본 MCU/MPU 하드웨어 보호 메커니즘을 사용하여 통신을 보호하고 코드 및 데이터 격리를 만드는 구성 요소를 OEM에 제공합니다. 그러나 각 OEM은 궁극적으로 디바이스에서 진화하는 보안 요구 사항을 충족하는지 확인해야 합니다.

## 이 페이지가 도움이 되었나요?

 Yes  No