

Technical Note

Action Command の使用方法

Action Command の使用方法

Action Command は GigE Vision 規格に規定された機能の 1 つで、この文章で使用例を挙げて使用方法を説明します。

1. 基本概念

同じネットワークに配置されている複数のデバイスを同時に動作させる時、Action Command が便利です。カメラ（または他のデバイス）をグループ分け、1つのコマンドで、それぞれ動作させることが可能です。

Action Command は、あらかじめカメラにキーデータを設定し、ネットワーク上、同じデータが送信されたときにそのアクションが動作します。

JAI GigE Vision カメラは 1 台あたり 2 つの Action があり、それぞれ Acquisition Start/Acquisition Stop/Frame Start/Transfer Start の入力ソースとして使用できます。Action Command を使用するときにはあらかじめトリガーの入力ソースを Action 1 又は 2 に指定しておく必要があります。

2. 受信デバイス（カメラ）側の設定

カメラの下記パラメータを設定する必要があります：

ActionDeviceKey

ActionDeviceKey は Action Command 機能のセキュリティのためのパラメータで、パスワードに相当します。正しい ActionDeviceKey を持っていない Action Command に対して、デバイスが反応しません。書き込み専用なので、他のデバイスに読み取られることを防止できます。ユーザーはカメラに設定した値をメモし、プログラムの中で同じ値を使用する必要があります。デバイスが Action Command を受信すると、ActionDeviceKey が一致するかどうかを最初にチェックします。

ActionSelector

JAI GigE Vision カメラには Action 1 と Action2 の 2 セットが用意されています。Action1 の設定に当てはまるコマンドを受信した時、入力ソースが Action1 に設定された機能が動作します。Action2 の設定に当てはまるコマンドを受信した時、入力ソースが Action2 に設定された機能が動作します。

ActionGroupKey



Technical Note

Action Command の使用方法

このアクションを動作させるためのキー（値）です。Device Key が一致した場合、Group Key が一致するかがチェックされます。

ActionGroupMask

アクションの動作をグループに分けるためのマスク値です。Group Key が一致した場合、最後に、該当 Action がこのコマンドの動作グループに入っているかがチェックされます。ActionGroupMask の設定値とコマンドの Group Mask が論理積演算され、論理積演算の結果が True の場合、該当動作グループに入っていることになりアクションの動作を実行します。

i Action Control	
Action Device Key	0x00
Action Selector	1
Action Group Key	0x00
Action Group Mask	0x00

図 1: Action Command パラメータ

3.Action Command 使用例

ここでは、2 つの GigE Vision カメラと 3 つの GigE Vision ストロボコントローラーを用いた例をいくつか示します。すべてのデバイスは Action Command 機能を持っており、3 つのストロボコントローラーは異なる波長のライトソース：IR、UV と白色(可視)になります。カメラにトリガーをかける同時に、どのライトソースを使用するかを、Action Command で制御します。

各デバイスは下記のようにパラメータを設定します：

●Camera1：

//カメラ 1 の Action1 を FrameStart の入力ソースとして使用します。

TriggerSelector = FrameStart

TriggerMode = ON

TriggerSource = Action1

//Action1 の設定

ActionDeviceKey = 0x12345678

ActionSelector = 1



Technical Note

Action Command の使用方法

```
ActionGroupKey[1] = 0x00000001
```

```
ActionGroupMask[1] = 0x00000001
```

//カメラ 1 の Action2 を AcquisitionEnd の入力ソースとして使用します。

```
TriggerSelector = AcquisitionEnd
```

```
TriggerMode = ON
```

```
TriggerSource = Action2
```

//Action2 の設定

```
ActionDeviceKey = 0x12345678
```

```
ActionSelector = 2
```

```
ActionGroupKey[2] = 0x88888888
```

```
ActionGroupMask[2] = 0x00001000
```

●Camera2 :

//カメラ 2 の Action2 を FrameStart の入力ソースとして使用します。

```
TriggerSelector = FrameStart
```

```
TriggerMode = ON
```

```
TriggerSource = Action2
```

//Action2 の設定

```
ActionDeviceKey = 0x12345678
```

```
ActionSelector = 2
```

```
ActionGroupKey[2] = 0x00000001
```

```
ActionGroupMask[2] = 0x00000002
```



Technical Note

Action Command の使用方法

//カメラ 2 の Action1 を AcquisitionEnd の入力ソースとして使用します。

TriggerSelector = AcquisitionEnd

TriggerMode = ON

TriggerSource = Action1

//Action1 の設定

ActionDeviceKey = 0x12345678

ActionSelector = 1

ActionGroupKey[1] = 0x88888888

ActionGroupMask[1] = 0x00002000

●IR ストロボ :

//Trigger Strobe with Action1

TriggerSelector = Strobe

TriggerMode = ON

TriggerSource = Action1

//Action1 Command

ActionDeviceKey = 0x12345678

ActionSelector = 1

ActionGroupKey[1] = 0x00000001

ActionGroupMask[1] = 0x00010000

●UV ストロボ :



Technical Note

Action Command の使用方法

```
//Trigger Strobe with Action5
```

```
TriggerSelector = Strobe
```

```
TriggerMode = ON
```

```
TriggerSource = Action5
```

```
//Action5 Command
```

```
ActionDeviceKey = 0x12345678
```

```
ActionSelector = 5
```

```
ActionGroupKey[5] = 0x00000001
```

```
ActionGroupMask[5] = 0x00020000
```

●白色ストロボ :

```
//Trigger Strobe with Action1
```

```
TriggerSelector = Strobe
```

```
TriggerMode = ON
```

```
TriggerSource = Action1
```

```
//Action1 Command
```

```
ActionDeviceKey = 0x12345678
```

```
ActionSelector = 1
```

```
ActionGroupKey[1] = 0x00000001
```

```
ActionGroupMask[1] = 0x00040000
```

- ・ Action Command 送信後、すべての機器が反応してくれるため、ActionDeviceKey は同じ値に設定する必要があります。



Technical Note

Action Command の使用方法

・動作の入力ソースをどの Action Command 番号に指定するかは自由です。何番まで指定できるかはデバイスに依存します。JAI GigE Vision カメラは 2 セット持っているので、番号 2 まで指定できます。

3.1 使用例 1 : カメラ 1 と白色ストロボにトリガーをかける

カメラ 1 と白色ストロボを同時に動作させ、他のデバイスに影響しないようにするには、SDK で下記 Action Command を送信します :

```
#define DEVICE_KEY 0x12345678

#define GROUP_KEY 0x00000001

//ビット演算 0x00000001|0x00040000 = 0x00040001

J_Factory_SendActionCommand(hFactory, DEVICE_KEY, GROUP_KEY, 0x00040001);
```

3.2 使用例 2 : カメラ 1、カメラ 2 と UV ストロボにトリガーをかける

```
#define DEVICE_KEY 0x12345678

#define GROUP_KEY 0x00000001

//ビット演算 0x00000001|0x00000002|0x00020000 = 0x00020003

J_Factory_SendActionCommand(hFactory, DEVICE_KEY, GROUP_KEY, 0x00020003);
```

3.3 使用例 3 : すべてのカメラとすべてストロボにトリガーをかける

```
#define DEVICE_KEY 0x12345678

#define GROUP_KEY 0x00000001

J_Factory_SendActionCommand(hFactory, DEVICE_KEY, GROUP_KEY, 0xFFFFFFFF);
```



Technical Note

Action Command の使用方法

3.4 使用例 4 : すべてのカメラにトリガーをかける

```
#define DEVICE_KEY 0x12345678

#define GROUP_KEY 0x00000001

//ビット演算 0x00000001|0x00000002 = 0x00000003

J_Factory_SendActionCommand(hFactory, DEVICE_KEY, GROUP_KEY, 0x00000003);
```

3.5 使用例 5 : カメラ 2 にトリガーをかける

```
#define DEVICE_KEY 0x12345678

#define GROUP_KEY 0x00000001

J_Factory_SendActionCommand(hFactory, DEVICE_KEY, GROUP_KEY, 0x00000002);
```

3.6 使用例 6 : カメラ 1 を撮像停止させる

```
#define DEVICE_KEY 0x12345678

#define GROUP_KEY_STOP 0x88888888

J_Factory_SendActionCommand(hFactory, DEVICE_KEY, GROUP_KEY_STOP, 0x00001000);
```

3.7 使用例 7 : カメラ 1 と 2 両方を撮像停止させる

```
#define DEVICE_KEY 0x12345678

#define GROUP_KEY_STOP 0x88888888

//ビット演算 0x00001000|0x00002000= 0x00003000

J_Factory_SendActionCommand(hFactory, DEVICE_KEY, GROUP_KEY_STOP, 0x00003000);
```



Technical Note

Action Command の使用方法

4. Scheduled Action Command

IEEE1588-2008 規格では、Precision Time Protocol (PTP) が定義されています。PTP により、複数のデバイスをエポック秒 (UNIX 時間) と同期させることができます。さらに、Scheduled Action Command も規格で定義されています。Scheduled Action Command により、(PTP で同期された) 複数のカメラが同じ時刻 (エポック秒) でスケジュールされたアクションを実行できます。許容誤差はナノ秒オーダーです。

このセクションでは、eBus Player for JAI を使用して Scheduled Action Command を設定する方法について説明します。

PTP の詳細プロトコルと Scheduled Action Command につきましてはは、IEEE1588-2008 規格ドキュメントをご参照下さい。

JAI カメラは PTP スレーブとして機能します。そのため、ネットワーク上少なくとも 1 つの PTP サーバーが存在する必要があります。PTP サーバー、ホスト PC、およびカメラの構成例は次のとおりです：

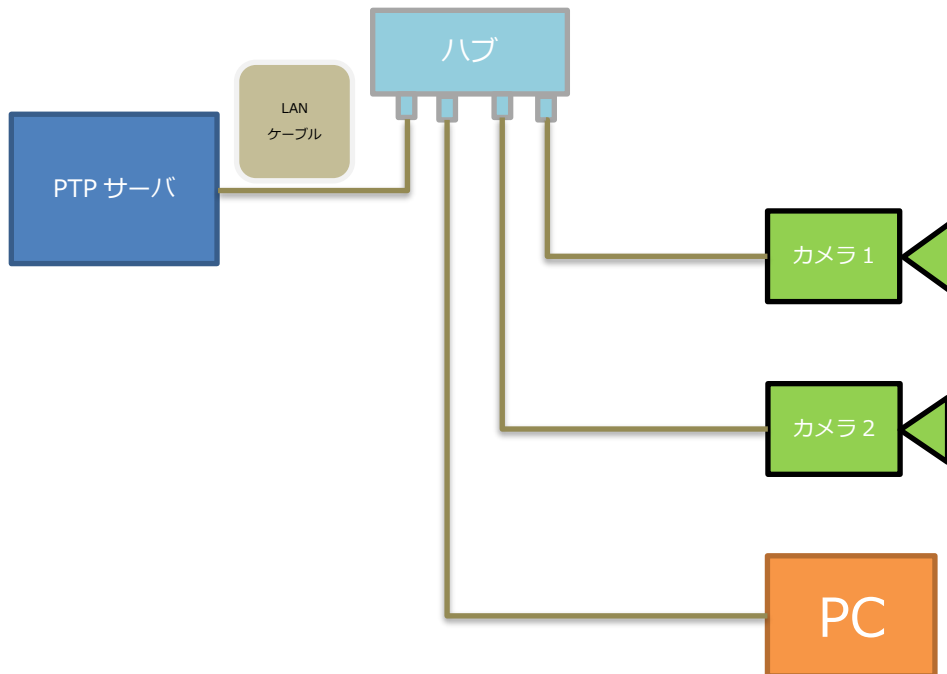


図 2:構成例

eBus Player for JAI では、次のように PTP と Scheduled Action Command を設定できます：

1. [Device Control] で、[ActionDeviceKey], [ActionGroupMask], [ActionGroupMask] を設定します。
2. [Device Control] で、GevIEEE1588 (TransportLayerControl) を[True] に設定します。



Technical Note

Action Command の使用方法

3. [Device Control] で、GevIEEE1588Status が Listening から Slave になりましたら、接続完了となります。
4. eBUS Player for JAI の[Tools] から[GigEVision Action Command] を選択し、[GigEVision Action Command] ダイアログを開きます。

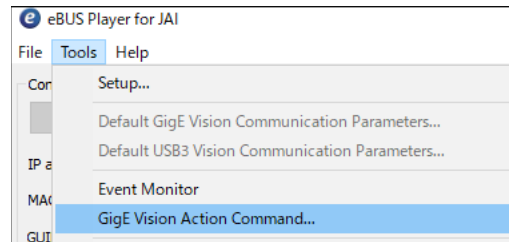


図 3:メニューバー

5. [Local Interfaces] グループボックスから、アクションを指示するデバイスのインタフェースを選択します。

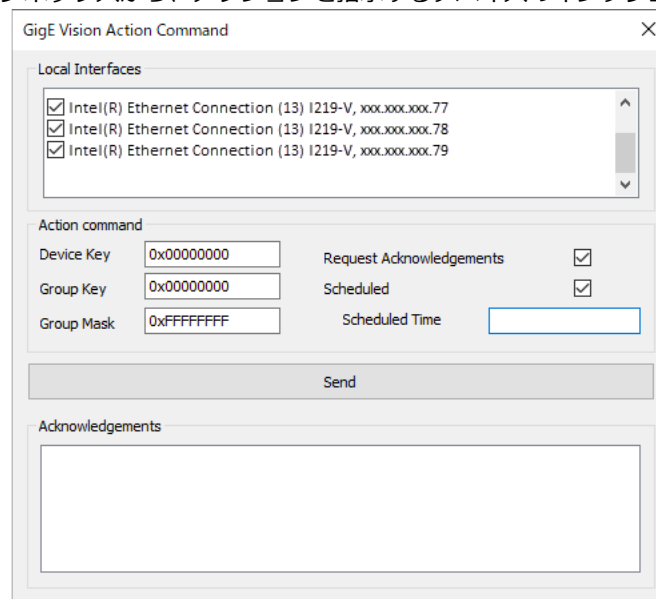


図 4:Action Command ダイアログ

6. [Action command] グループボックスの[Device Key], [Group Key], [Group Mask] に、ステップ 1 で指定した値を入力します(値は一致する必要があります)。
7. [Scheduled] チェックボックスを On にし、アクションコマンドを実行する時刻を[Scheduled Time] ボックスに入力します。[Scheduled Time]はエポック秒で入力する必要があります。下記 URL を参照して、エポックタイムを計算できます：



Technical Note

Action Command の使用方法

<https://www.epochconverter.com/>

8. [Send] をクリックし、ステップ 6 と 7 で指定した Key 及び時間で、ステップ 5 で指定したデバイスにアクションコマンドを送信します。

9. [Request Acknowledgements] チェックボックスを On にした場合、Acknowledgement が表示されます：

- ・ OK : アクションコマンドが送信され、指定のカメラによって受信されました。指定の時刻にアクションが実行されます。
- ・ No Ref Time : [Device Control]→TransportLayerControl→GevIEEE1588 (TransportLayerControl) が[False] になっています。
- ・ Overflow : アクションコマンドのキューバッファがいっぱいです。
- ・ Late : [Scheduled Time]の時刻が過ぎているため、コマンドが即時実行されました。

※注意：通信環境が悪いとタイミングに差分が出やすくなります。

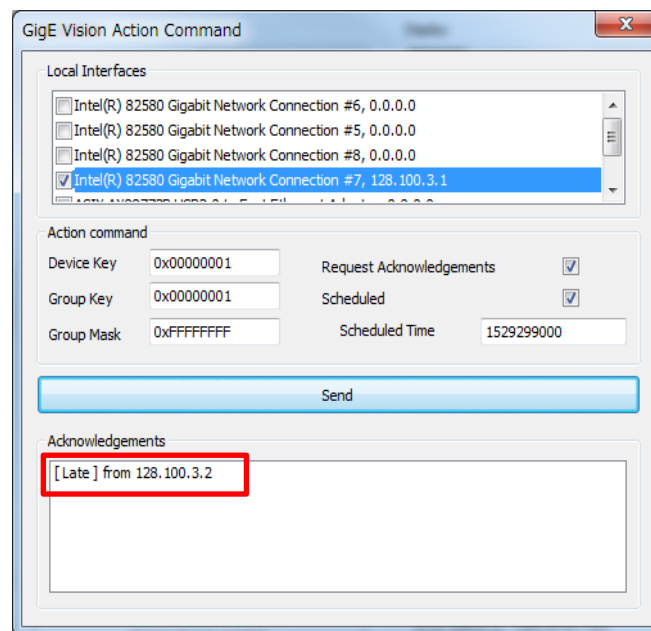


図 5:遅れた場合のメッセージ

以上



Technical Note

Action Command の使用方法

變更履歷

[illegible]