BU シリーズ
CMOS カメラ

取扱説明書

適用機種

白黒カメラ　：BU2006MG

カラーカメラ：BU2006MCF

この度は、弊社製品をお買い上げいただきまして、誠にありがとうございます。お求め頂いた CMOS カメラを安全に正しく使っていただくために、ご使用になる前にこの『取扱説明書』をよくお読みください。
お読みになった後は、いつでも手元においてご使用ください。

東芝テリー株式会社

改善の為予告なく変更することがありますので、最新の仕様書・取扱説明書にて機能・性能をご確認ください。

本文中の規格名は、各社各団体における商標または登録商標の場合があります。

DAA03566B

https://www.toshiba-teli.co.jp/

もくじ

安全上のご注意

ご使用の前に、この安全上のご注意をよくお読みのうえ、正しくお使いください。この取扱説明書には、お使いになるかたや他の人への危害と財産の損害を未然に防ぎ、安全に正しくお使いいただくために、重要な内容を記載しています。

次の内容(表示・図記号)を良く理解してから本文をお読みになり、記載事項をお守りください。

[表示の説明]

表示	表示の意味
⚠ 警告	”取扱いを誤った場合、使用者が死亡または重傷(*1)を負うことが想定されること”を示します。
⚠ 注意	”取扱いを誤った場合、使用者が傷害(*2)を負うことが想定されるか、または物的損害(*3)の発生が想定されること”を示します。

*1：重傷とは、失明やけが、やけど(高温・低温)、感電、骨折、中毒などで、後遺症が残るもの、および治療に入院・長期の通院を要するものをさします。

*2：傷害とは、治療に入院や長期の通院を要さない、けが・やけど・感電などをさす。

*3：物的損害とは、家屋・財産・および家畜・ペット等にかかわる拡大損害をさす。

[図記号の説明]

図記号	図記号の意味
禁止	**禁止**(してはいけないこと)を示します。 具体的な禁止内容は、図記号の中や近くに絵や文章で示しています。
指示	**指示する行為の強制**(必ずすること)を示します。 具体的な指示内容は、図記号の中や近くに絵や文章で示しています。

取扱全般について

⚠ 警 告

プラグを抜け

● **異常や故障のときは、すぐ使用をやめること**

煙が出る、こげくさい、落として破損した、内部に水や異物が入ったなどの異常状態で使用すると、火災・感電の原因となります。

すぐに機器の電源プラグをコンセントから抜き、販売元にご連絡ください。

水ぬれ禁止

● **水がかかる場所で使用しないこと**

火災・感電の原因となります。

分解禁止

● **分解・修理・改造はしないこと**

火災・感電の原因となります。

内部の修理・点検・清掃は販売元にご依頼ください。

禁止

● **本機の上に物を置かないこと**

金属類や液体など、異物が内部に入った場合、火災・感電の原因となります。

禁止

● **不安定な場所、傾いた所、振動・衝撃のある所に置かないこと**

落ちたり倒れたりして、けがの原因となります。

接触禁止

● **雷が鳴り出したら、機器の電源コードや接続ケーブルに触れないこと**

感電の原因となります。

指示

● **指定された電源電圧を使用すること**

指定された電源電圧以外では、火災・感電の原因となります。

禁止

● **電源コード・接続ケーブルを傷つけたり、破損したり、加工したり、無理に曲げたり引っ張ったり、ねじったり、束ねたり、重い物を乗せたり、加熱したりしないこと**

火災・感電の原因となります。

⚠ 注 意


指示

● **設置の際は次のことを守ること**

・布などで包まない

・熱のこもりやすい狭い場所に押し込まない

内部に熱がこもり、火災の原因となることがあります。


禁止

● **湿気・油煙・湯気・ほこりの多い場所に置かないこと**

火災・感電の原因となることがあります。


禁止

● **直射日光の当たる場所や温度の高い場所に置かないこと**

内部の温度が上がり、火災の原因となることがあります。


指示

● **指定された電源ケーブル・接続ケーブルを使用すること**

ケーブルを傷めたり、断線の原因となります。


禁止

● **接続ケーブルを強く引っ張ったり回したりしないでください**

故障の原因となることがあります。


指示

● **接続の際は電源を切る**

電源ケーブルや接続ケーブルを接続するときは、電源を切ってください。

感電や故障の原因となることがあります。


禁止

● **過大な光(太陽光等)に長時間さらさないこと**

故障の原因となることがあります。


禁止

● **信号の出力は短絡しないこと**

故障の原因となることがあります。


禁止

● **カメラ本体に強い衝撃を与えないこと**

故障・破損の原因となることがあります。

カメラコネクタ部に強い衝撃が加わるシステムで使用された場合、カメラコネクタが破損する場合があります。その様なシステムで使用される場合、カメラケーブルをなるべくカメラ本体に近い所で束線し、カメラコネクタに衝撃がつたわらないようにしてください。


指示

● **定期的(おおむね５年に１度)に点検・清掃を販売店にご依頼ください**

内部にほこりがたまると、火災・故障の原因となることがあります。

点検・清掃費用については販売店にお尋ねください。

免責事項

- 地震、雷などの自然災害、火災、第三者による行為、その他事故、お客様の故意または過失、誤用、その他異常な条件下での使用によって生じた損害に関して、弊社は一切責任を負いません。

- 本製品の使用または使用不能から生じる付随的な損害(事業利益の損失・事業の中断・記憶内容の変化・消失など)に関して、弊社は一切責任を負いません。

- 仕様書や取扱説明書の記載内容を守らないことによって生じた損害に関して、弊社は一切責任を負いません。

- 仕様書や取扱説明書に記載されている以外の操作方法によって生じた損害に関して、弊社は一切責任を負いません。

- 弊社が関与しない接続機器(USB インターフェースボード、レンズ含む)、ソフトウェア等との意図しない組み合わせによる誤動作等から生じた損害に関して、弊社は一切責任を負いません。

- お客様ご自身又は権限のない第三者(指定外のサービス店等)が修理・改造を行った場合に生じた損害に関して、弊社は一切責任を負いません。

- 本製品に関し、いかなる場合も弊社の費用負担は本製品の個品価格以内とします。

- 本製品の仕様書に記載のない項目につきましては、保証対象外とします。

- ケーブルの取り付けミスによるカメラ破損に関しては、保証の対象外とさせていただきます。

用途制限

● 次に示すような条件や環境で使用する場合は、安全対策への配慮を頂くとともに、弊社にご連絡くださるようお願いいたします。

1．明記されている仕様以外の条件や環境、屋外での使用。

2．人や財産に大きな影響が予想され、特に安全が要求される用途への使用。

● 本製品は、使用される条件が多様なため、その装置・機器への適合性の決定は装置・機器の設計者または仕様を決定する人が、必要に応じて分析やテストを行ってから決定してください。この装置・機器の性能および安全性は、装置・機器への適合性を決定されたお客様において保証してください。

● 本製品は、人の生命に直接関わる装置(*1)や人の安全に関与し公共の機能維持に重大な影響を及ぼす装置(*2)などの制御に使用するよう設計・製造されたものではないため、それらの用途に使用しないでください。

(*1)：人の生命に直接関わる装置とは、次のものをさします。

- 生命維持装置や手術室用機器などの医療機器
- 有毒ガスなどの排ガス、排煙装置
- 消防法、建築基準法などの各種法令により設置が義務づけられている装置
- 上記に準ずる装置

(*2)：人の安全に関与し公共の機能維持に重大な影響を及ぼす装置とは、次のものをさします。

- 航空、鉄道、道路、海運などの交通管制装置
- 原子力発電所などの装置
- 上記に準ずる装置

使用上のお願い

● CMOS センサの寿命について

本製品で使用しているCMOS センサは他製品と異なり、産業機器用 CMOS センサではございません。従いまして、カメラの使用条件（周囲温度、放熱条件等）によっては CMOS センサの寿命が短くなる場合がありますので、あらかじめご了承願います。

● 取り扱いはていねいに

落下させたり強い衝撃や振動を与えたりしないでください。故障の原因になります。また、接続ケーブルは乱暴に取り扱わないでください。ケーブル断線の恐れがあります。

● 使用温度・湿度

仕様を超える温度・湿度の場所では使用しないでください。

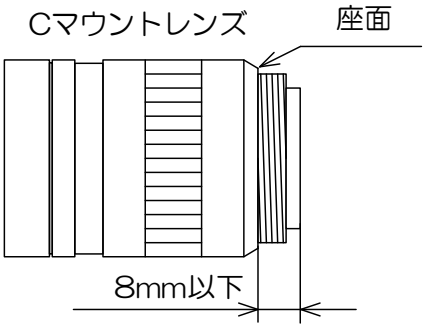
画質の低下の他、内部の部品に悪影響を与えます。直射日光の当たる所でのご使用には特にご注意ください。また、高温時での撮影では被写体やカメラの状態(ゲインを上げている場合等)によっては縦スジや白点状のノイズが発生することがありますが、故障ではありません。

● 組み合わせレンズについて

ご使用になられるレンズ及び照明の組み合わせによっては、撮像エリアにゴーストとして映り込む場合がありますが、本製品の故障ではありません。また、レンズによっては周辺部の解像度及び明るさの低下、収差等、カメラの性能を充分に発揮できないことがあります。ご使用になられるレンズ及び照明で、本製品との組み合わせ確認を行って頂けるようお願いいたします。

カメラにレンズ等を取付けるときは、傾きがないよう良く確かめてから取付けてください。またマウントのネジ部にキズやゴミ等がない物をご使用ください。カメラが外れなくなる場合があります。

本製品と組み合せて使用するレンズは、レンズが取り付かない場合がありますので座面からの突出寸法が8mm 以下の C マウントレンズを使用してください。



● カメラの取り付けについて

本製品を台座等に取り付ける場合には、レンズと台座等が接触しないよう、お客様にて十分配慮した取り付けをお願いいたします。

● 撮像面を直接太陽や、強烈なライトなどに向けない

CMOS センサが熱的に損傷することがあります。

● モアレの発生

細かい縞模様を撮ると実際にはない縞模様(モアレ)が干渉ジマとして現れることがありますが、故障ではありません。

● 画面ノイズの発生

カメラの設置ケーブル類の配線に際し、強い磁気を発するものの近くや、強力な電波を発するものの近くにあると、画面ノイズが入ることがあります。そのときは位置や配線を変えてください。

● 保護キャップの取り扱い

カメラをご使用にならない時は、撮像面の保護のためレンズキャップを取り付けてください。

● 長時間ご使用にならないとき

安全のため電源の供給を停止しておいてください。

● お手入れ

電源を切って乾いた布で拭いてください。

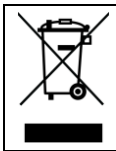
汚れのひどい場合には、うすめた中性洗剤を柔らかい布に染み込ませて軽く拭いてください。アルコール、ベンジン、シンナーなどは使用しないでください。塗装や表示がはげたり、変質したりすることがあります。

万一撮像面にゴミ・汚れ・キズなどがついた場合には、販売店にご相談ください。

● 破棄をするとき

本製品は、環境汚染を防止する為、各国の法律や地方自治体の法令などに従い、適切な分別破棄をしてください。

尚、ＥＵ環境規制(廃電気電子機器指令(WEEE))により、製品本体に下記シンボルを表示しています。



「このシンボルはＥＵ加盟国にのみ適用されます」
“This symbol is applicable for EU member states only”

本製品は、ＦＣＣ規則第１５条クラスＡの制限にしたがって試験されたデジタル機器です。この制限は工業的環境で製品が運用された時の有害な妨害から適度な保護をする為に設定されました。この製品を使い、発生したラジオ周波数のエネルギー放射は、取扱説明書と違う設置や使い方によってラジオコミュニケーションに有害な妨害を与える場合があります。この製品を住宅で取り扱う事は、妨害の原因となる事が十分に考えられ、自身の責任で妨害を矯正する事が必須となります。

［CMOS センサ特有の現象］

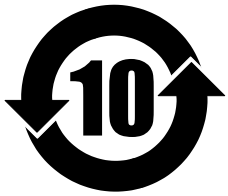
■欠陥画素

CMOS イメージセンサはフォトセンサ素子が縦・横に並んで配置されており、フォトセンサ素子のいずれかに欠陥があると、その部分の画像が映らず、モニタ画面上に於いて白又は黒のキズが発生します。キズの数量及び明るさは定温状態に比べ高温状態に於いて増加します。また、露光時間が短い時に比べ露光時間が長い場合に於いて増加します。

この時キズがノイズ状に見える場合がありますが、CMOS イメージセンサの特性であり故障ではありませんのでご注意ください。

■画像シェーディング

画面上部と下部の明るさが異なる現象が発生する場合がありますが、CMOS イメージセンサの特性であり故障ではありませんのでご注意ください。

 中华人民共和国 环保使用期限	环保使用期限标识，是根据电子信息产品污染控制管理办法以及，电子信息产品污染控制标识要求(SJ/T11364-2014)、电子信息产品环保使用期限通则，制定的适用于中国境内销售的电子信息产品的标识。 电子信息产品只要按照安全及使用说明内容，正常使用情况下，从生产月期算起，在此期限内，产品中含有的有毒有害物质不致发生外泄或突变，不致对环境造成严重污染或对其人身、财产造成严重损害。 产品正常使用后，要废弃在环保使用年限内或者刚到年限的产品时，请根据国家标准采取适当的方法进行处置。 另外，此期限不同于质量/功能的保证期限。 The Mark and Information are applicable for People's Republic of China only.

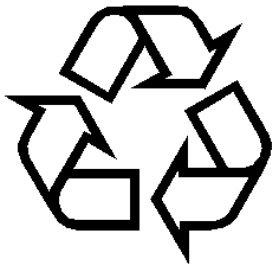
＜产品中有毒有害物质或元素的名称及含量＞

部件名称	有毒有害物质或元素					
	铅（Pb）	汞（Hg）	镉（Cd）	六价铬（Cr(VI)）	多溴联苯（PBB）	多溴二苯醚(PBDE）
相机本体	×	○	○	○	○	○

本表格依据SJ/T 11364的规定编制
○：表示该有毒有害物质在该部件所有均质材料中的含量均在电子信息产品中有毒有害物质的限量要求标准规定的限量要求(GB/T26572)以下
×：表示该有毒有害物质至少在该部件的某一均质材料中的含量超出电子信息产品中有毒有害物质的限量要求标准规定的限量要求(GB/T26572)
This information is applicable for People's Republic of China only.

リサイクルに関する情報（包装物）
有关再利用的信息(包装物)
Information on recycling of wrapping composition

箱 / 箱子 / Box



ペーパーボード
纸板
Paper board

内部緩衝材料・袋
内部缓冲材料・袋
Internal buffer materials・Bag

インストール

本カメラシリーズを使用するに当たり、画像表示用アプリケーション、カメラ制御用レジスタコントローラが必要となります。

弊社 USB I/F デジタルカメラシリーズを PC から制御するためのソフトウェア開発キット(TeliCamSDK)は、弊社ホームページからダウンロードすることができます。

ダウンロードサービスをご利用になるにはユーザー登録が必要となりますので、ユーザー登録をしていただくか、弊社営業担当へお問い合わせください。

東芝テリー株式会社ホームページ

https://www.toshiba-teli.co.jp/

サービス&サポート

https://www.toshiba-teli.co.jp/cgi/ss/jp/service_j.cgi

TeliCamSDK の動作環境、インストール、セットアップについては、TelCamSDK 付属のスタートアップガイドをご参照ください。

仕様

概要

BU2006M シリーズは、20M 画素(1 型)のローリングシャッタ方式 CMOS センサを採用した一体型カメラです。カラーモデルは機種名末尾に[C]もしくは[CF]が付きます。映像出力・カメラ制御には USB3.1 Gen1 (USB3.0)規格を採用しており、高速で高解像度の画像処理に適しています。また、カメラ本体は、小型・軽量で機器組み込みに最適です。

特長

- 高速フレームレート
  20M 画素 19fps の高速フレームレートを実現します。

- グローバルリセット
  ランダムシャッタモード時には、全ラインで同時に撮像を開始するグローバルリセットで動作します。ストロボ照明と組み合わせることで、グローバルシャッタと同様の効果を得ることができます。

- USB3.1 Gen1 (USB3.0)インターフェース
  映像出力及びカメラ制御は USB3.1 Gen1 (USB3.0、以降 USB3.1 Gen1)インターフェースを介して行います。データ転送を 5Gbps(最大)で行い、非圧縮の出力画像を高速フレームレートで出力可能です。

- USB3 Vision 採用
  国際的工業用カメラ規格である USB3 Vision を採用しているため、カメラ制御を容易に行うことができます。

- GenICam Ver. 2.4, Ver. 3.0 採用
  国際的工業用カメラ規格である GenICam (Generic Interface for Cameras) Ver. 2.4 及び Ver. 3.0 を採用しているため、カメラ制御を容易に行うことができます。

- IIDC2 Digital Camera Control Specification Ver. 1.1.0 採用
  国際的工業用カメラ規格である IIDC2 Digital Camera Control Specification Ver. 1.1.0 を採用しているため、カメラ制御を容易に行うことができます。

- e-CON 規格コネクタ採用
  センサーコネクタの業界標準である e-CON 規格コネクタの採用により、専用工具が不要となり、容易にケーブルの製作ができます。

● ランダムトリガシャッタ機能

外部トリガ信号と同期して露光を開始するランダムトリガシャッタを装備していますので、高速移動物体を定位置に捕らえ、正確な画像処理ができます。

● スケーラブル機能

映像出力範囲を任意に指定することができます。垂直方向の出力範囲を制限することにより、更なる高速読み出しが可能になります。また水平方向の出力範囲を制限することにより、USB 転送線路の占有帯域を軽減できます。

● カラープロセス内蔵

カラーモデルは、カラープロセスを内蔵しております。RGB、BGR、YUV 4:2:2、YUV4:1:1、Bayer、Mono の出力モードを有しています。

● 防塵ガラス

標準で防塵ガラスが組み込まれています。

防塵ガラス組み込みモデルは機種名末尾に[G]が付きます。（例：BU2006MG）

● IR カットフィルタ

IR カットフィルタ組み込みのオプションを選択することができます。IR カットフィルタ組み込みモデルは機種名末尾に[F]が付き、人の目に近い色再現性を得ることができます。

※本資料内で共通仕様部分に関しては末尾の[F]は省略します。

● 小型、軽量

小型・軽量で耐振動、衝撃性に優れています。

● RoHS 指令対応

有害物質の使用禁止を定めた RoHS 指令に対応しています。

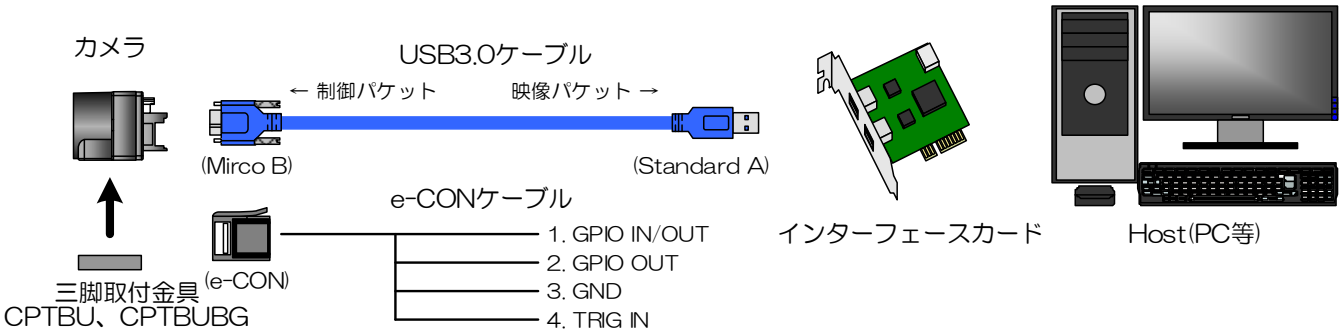
構成

構成は以下の通りになります。本製品には付属品がありません。カメラ以外は別売りとなりますので、必要に応じて別途準備してください。

- カメラ：　本製品
- 三脚取付金具 CPTBU, CPTBUBG (※1)：　三脚等を使用する場合、カメラの底面に取り付けます。
- USB3.0 ケーブル(※2)：　カメラ背面の USB コネクタに接続します。Standard A -micro B の USB3.0 ケーブルをご使用ください。本製品はスクリューロック機構のあるUSBケーブルも接続可能ですので、必要に応じてご使用ください。
- USB3.1 Gen1 インターフェースカード(※2)：カメラと接続するインターフェースカードです。通常 PC 等のホスト側の拡張スロットに挿入します。
- e-CON ケーブル(※2)：　外部トリガ、GPIO 機能を使用する場合、カメラ背面の e-CON コネクタに接続します。カメラの使用環境によっては、ノイズの影響を受ける可能性があるため、シールドケーブルの使用を推奨致します。

※1 弊社オプション品。オプション品の詳細は、弊社営業担当にお問い合わせください。
※2 市販品。

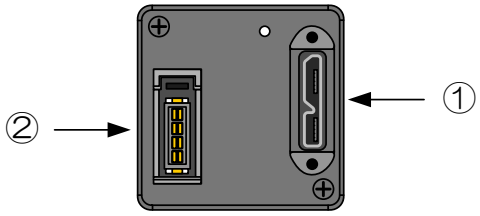
接続例



お願い：接続について

- コネクタ部に強い衝撃が加わるシステムで使用される場合は、ロックネジ付きの USB ケーブルをご使用ください。また、ケーブルをなるべくカメラ本体に近いところで束線し、コネクタに衝撃が伝わらないようにしてください。
- カメラケーブルについて、電線の種類・長さによっては電圧降下により、カメラの電源電圧仕様を満たさない場合がありますので、ご使用前に十分ご確認ください。
- 使用する USB ケーブル、インターフェースカードについて、伝送路の電気的特性によりパケット落ちが発生する場合があります。

コネクタピン配置



カメラ背面

①USB3.0 インターフェースコネクタ

コネクタ型名　　WMUR-10F6L1PH5N（WIN WIN PRECISION INDUSTRIAL 製）

Pin No.	I/O	信号名	機能
1	-	VBUS	Power
2	I/O	D-	USB2.0 differential pair
3	I/O	D+	
4	-	NC	Not connected
5	-	GND	Ground for power return
6	O	SSTX-	SuperSpeed transmitter differential pair
7	O	SSTX+	
8	-	GND_DRAIN	Ground for SuperSpeed signal return
9	I	SSRX-	SuperSpeed receiver differential pair
10	I	SSRX+	

②I/O コネクタ

コネクタ型名（カメラ側）　　37204-62B3-004PL（スリーエムジャパン製）

適合コネクタ（ケーブル側）　　e-CON 準拠コネクタ

例：37104 シリーズ（スリーエムジャパン製）

RITS 4P シリーズ（TE Connectivity 製）

※ 本製品に適合コネクタは付属していません。

コネクタピン配列

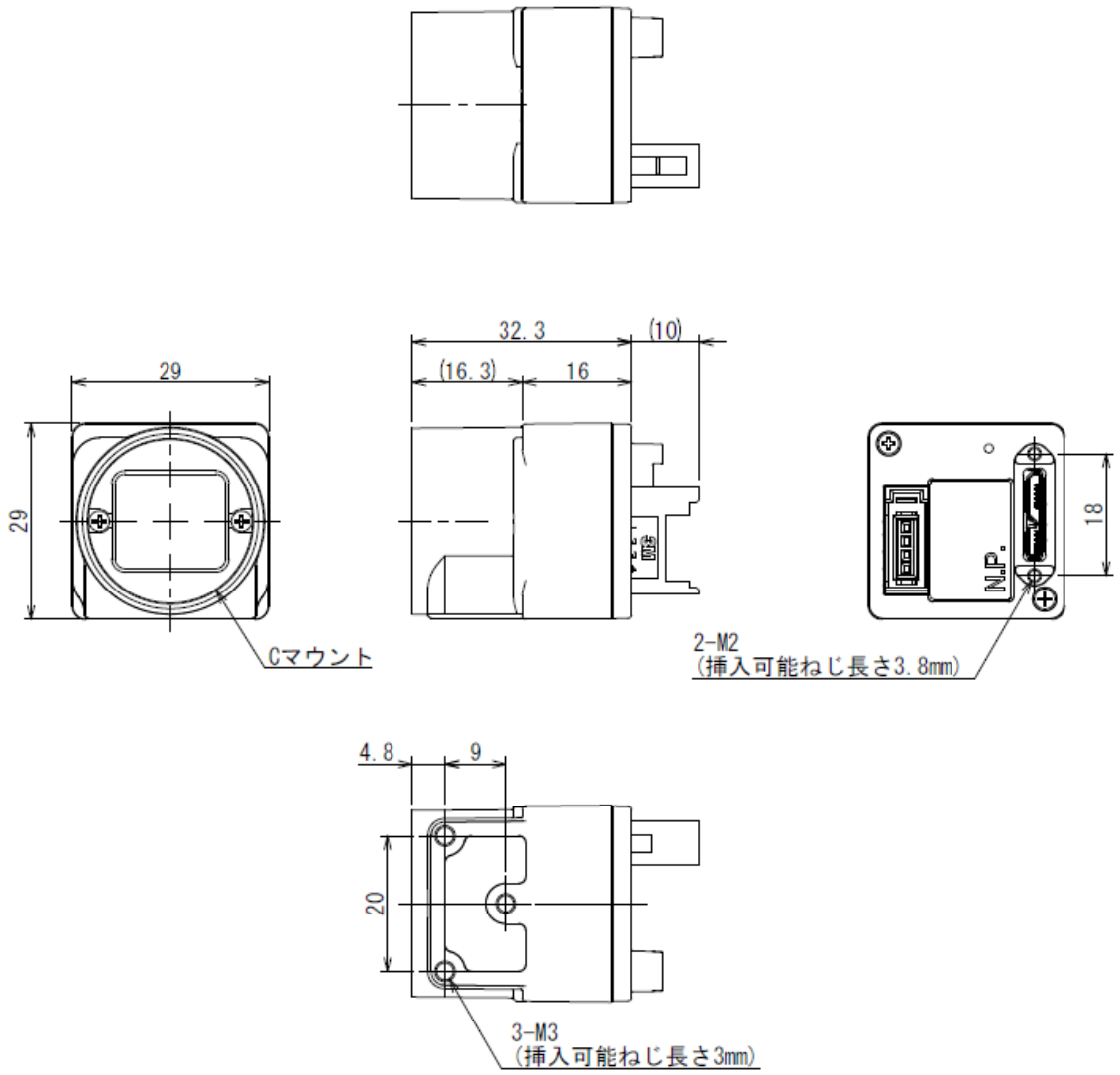
①
②
③
④

↑TOP

※　コネクタを嵌合側から見た図です。

Pin No.	I/O	信号名	機能
1	I/O	Line2	GPIO Input/Output
2	O	Line1	GPIO Output
3	-	GND	Ground
4	I	Line0	GPIO Input

外形仕様



仕様
主材質：アルミニウムダイカスト合金
処理：カチオン塗装（黒色）

主な仕様

● 白黒モデル

機種型名	BU2006MG
撮像素子	CMOS イメージセンサ
出力最大画素数(H)×(V)	5472×3648
光学サイズ	1 型相当
撮像面積(H) x (V)[mm]	13.13×8.76
画素サイズ(H) x (V)[μm]	2.4×2.4
走査方式	プログレッシブ
電子シャッタ方式	ローリングシャッタ
アスペクト比	3:2
感度	1210lx, F5.6, 1/19s
最低被写体照度(※1)	3lx
電源	DC+5V±5% (USB コネクタより給電)
消費電力(※1)	2.7W 以下
映像インターフェース方式	USB3.1 Gen1 (SuperSpeed のみサポート)
映像転送速度	5Gbps (最大)
プロトコル	USB3 Vision
映像出力フォーマット	Mono8, Mono10, Mono12
最大フレームレート(※2)	
Mono8	19fps
Mono10, Mono12	9.5fps
外形寸法	29 mm(W) x 29 mm (H) x 16 mm (D) (突起物を含まず)
質量	約 34g
レンズマウント	C マウント
フランジバック	17.526mm
フレーム接地 / 絶縁状況	回路 GND ～ 筐体間導通あり

(※1) F1.4, ゲイン：最大（+24dB），映像レベル：50%

(※2) 全画素読出し時

● カラーモデル

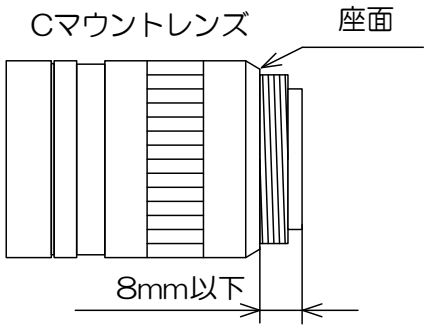
機種型名	BU2006MCF
撮像素子	CMOS イメージセンサ
出力最大画素数(H)×(V)	5472×3648
光学サイズ	1 型相当
撮像面積(H) x (V)[mm]	13.13×8.76
画素サイズ(H) x (V)[μm]	2.4×2.4
走査方式	プログレッシブ
電子シャッタ方式	ローリングシャッタ
アスペクト比	3:2
感度	1940x, F5.6, 1/19s
最低被写体照度(※1)	4lx
電源	DC+5V±5% (USB コネクタより給電)
消費電力(※1)	2.9W 以下
映像インターフェース方式	USB3.1 Gen1 (SuperSpeed のみサポート)
映像転送速度	5Gbps (最大)
プロトコル	USB3 Vision
映像出力フォーマット	RGB, BGR, YUV4:2:2, YUV4:1:1, Bayer8, Bayer10, Bayer12, Mono8
最大フレームレート(※2)	
Bayer8, Mono8	19fps
YUV4:1:1	12.7fps
YUV4:2:2	9.5fps
Bayer10, Bayer12	9.5fps
RGB, BGR	6.3fps
外形寸法	29 mm(W) x 29 mm (H) x 16 mm (D)　(突起物を含まず)
質量	約 34g
レンズマウント	C マウント
フランジバック	17.526mm
フレーム接地 / 絶縁状況	回路 GND ～ 筐体間導通あり

(※1) F1.4, ゲイン：最大（+24dB），映像レベル：50%

(※2) 全画素読出し時

お願い：組み合わせレンズについて

- ご使用になられるレンズ及び照明の組み合わせによっては、撮像エリアにゴーストとして映り込む場合がありますが、本製品の故障ではありません。また、レンズによっては周辺部の解像度及び明るさの低下、収差等、カメラの性能を充分に発揮できないことがあります。ご使用になられるレンズ及び照明で、本製品との組み合わせ確認を行って頂けるようお願いいたします。
- カメラにレンズ等を取付けるときは、傾きがないよう良く確かめてから取付けてください。またマウントのネジ部にキズやゴミ等がない物をご使用ください。カメラが外れなくなる場合があります。
- 本製品と組み合せて使用するレンズは、レンズが取り付かない場合がありますので座面からの突出寸法が 8mm 以下の C マウントレンズを使用してください。



LED 表示

カメラの状態	LED 表示
電源供給なし	消灯
リンク検出中	緑の高速点滅（ON:20ms, OFF:60ms）
接続エラー	赤と緑が交互に点滅
SuperSpeed 接続確立。転送なし。	緑の中速点滅（ON:200ms, OFF:800ms）
SuperSpeed 接続確立。トリガ待ち。	橙の中速点滅（ON:200ms, OFF:800ms）
データ転送中	緑の高速点灯（ON:60ms, OFF:20ms）
転送エラー	赤点灯（500ms 期間）
スタンバイ	橙の超低速点滅（ON:200ms, OFF:2800ms）

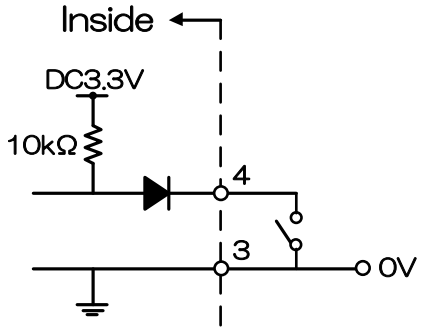
I/O 入出力信号仕様

● 信号仕様

・Line0（I/O コネクタ：4 ピン）

入出力仕様	：入力専用
入力回路	：LVTTL 入力
信号レベル	：Low 0 ～ 0.5V、High 2.0 ～ 24.0V
動作エッジ極性	：出荷設定 負極性(カメラ設定にて切り替え可能)
パルス幅	：最小５０μs

入力回路図



お願い：トリガ入力信号について

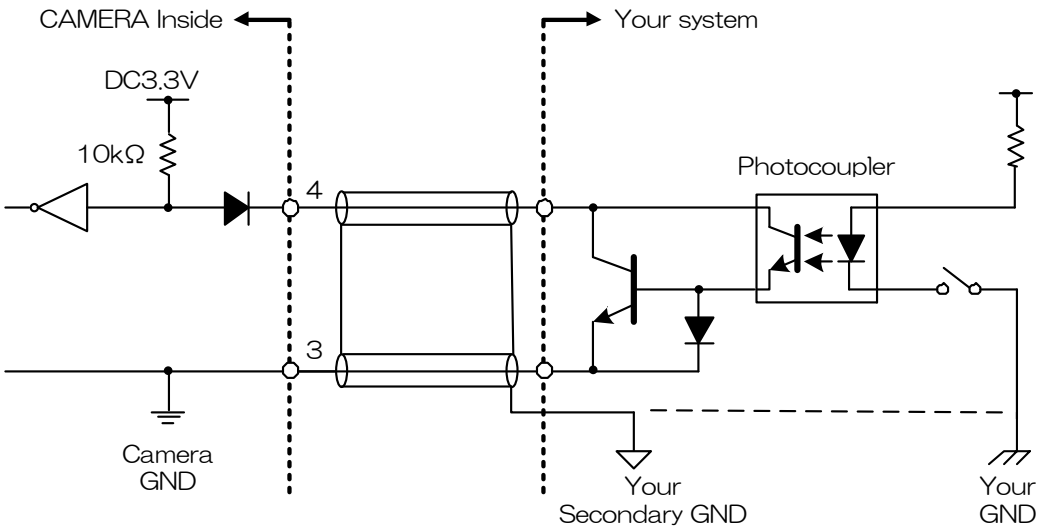
ケーブル長・線種、トリガライン入力電流値によっては、カメラ側にてトリガ信号を受けられない場合がありますので、ご確認の上ご使用ください。

お願い：トリガ入力レベルについて

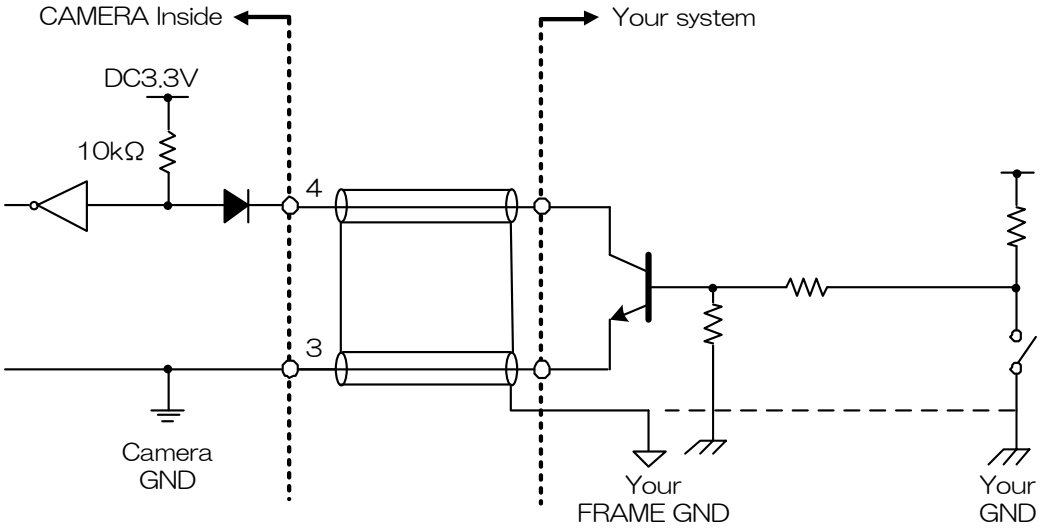
Line0 と Line2 の最大入力信号レベルは異なります。本取扱説明書に記載の電圧より高いレベルの信号を入力すると故障の原因となりますので、十分ご確認の上ご使用ください。

● 外部トリガ入力回路例

・Isolated I/F



・Non-Isolated I/F



お願い：トリガ入力ケーブルについて

・カメラの I/O コネクタの 3 ピンは、カメラ筐体（フレーム）と導通しています。
シールドケーブルを使用する場合、シールド編組はお客様自身のシステムフレーム GND に接続するか、システムシグナル GND へ接続をお願い致します。

・EMC 適合性の確認は、最終的にお客様のシステム全体で実施をして頂くようお願致します。

• Line2（I/O コネクタ：1 ピン）

入出力仕様	：入力／出力（LineMode により切替え可能）
出荷設定	：出力

入力信号仕様

入力回路	：5V CMOS 入力
信号レベル	：Low 0 ～ 0.5V、High 4.0 ～ 5.0V
動作エッジ極性	：出荷設定 負極性(カメラ設定にて切り替え可能)
パルス幅	：最小５０μs

お願い：トリガ入力信号について

ケーブル長・線種、トリガライン入力電流値によっては、カメラ側にてトリガ信号を受けられない場合がありますので、ご確認の上ご使用ください。

お願い：トリガ入力レベルについて

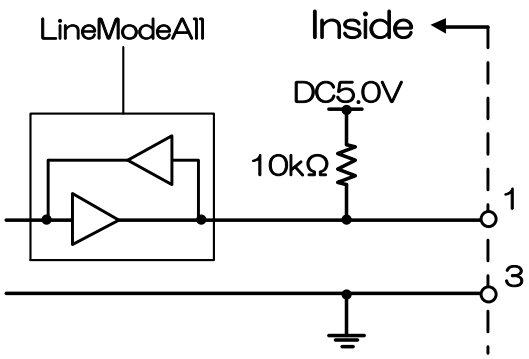
Line0 と Line2 の最大入力信号レベルは異なります。本取扱説明書に記載の電圧より高いレベルの信号を入力すると故障の原因となりますので、十分ご確認の上ご使用ください。

出力信号仕様

信号レベル	：5V CMOS
最大電流	：+/-32mA(駆動電流)
信号極性	：出荷設定 負極性(カメラ設定にて切り替え可能)
出力信号（LineSource）	：以下から選択 Off※ / UserOutput / Timer0Active / FrameTriggerWait FrameActive / FrameTransferActive / ExposureActive

※LineMode が出力に設定されている場合でも、LineSource に Off を設定している場合は Line2 への入力信号は有効となります。

入出力回路図



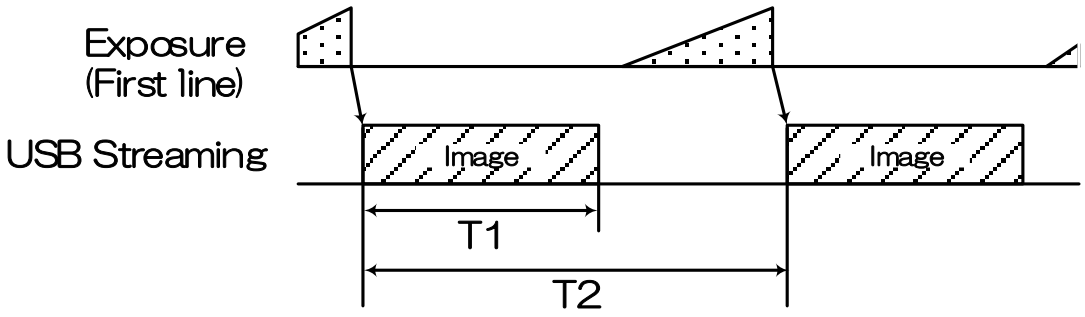
• Line1（I/O コネクタ：2 ピン）

入出力仕様	：出力専用
信号レベル	：5V CMOS
最大電流	：+/-32mA(駆動電流)
信号極性	：出荷設定 負極性(カメラ設定にて切り替え可能)
出力信号（LineSource）	：以下から選択 Off / UserOutput / Timer0Active / FrameTriggerWait FrameActive / FrameTransferActive / ExposureActive

タイミング仕様

本製品は映像データの転送プロトコルに USB のバルク転送を使用しています。以降に想定されたタイミング数値は伝送帯域を他ノードの制約無しに使用できることが絶対条件です。本製品と同時に転送を行っているノードがある場合は以降で規定した数値どおりではありません。

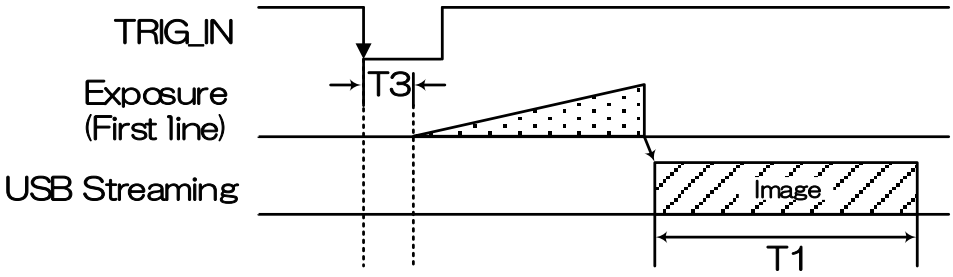
● ノーマルシャッタ動作



全画素読み出し時

型名	フォーマット	T1 [ms]	T2 [s]
BU2006MG	Mono8	51.8	1/(フレームレート設定値)
	Mono10, Mono12	97.8	
BU2006MCF	Bayer8, Mono8	51.8	
	YUV4:1:1	73.3	
	YUV4:2:2	97.8	
	Bayer10,Bayer12	97.8	
	RGB,BGR	147.0	

● ランダムトリガシャッタ動作



Edge モード/Bulk モード（全画素読み出し時）

型名	フォーマット	T3 [μs]
BU2006MG	Mono8	85.0
	Mono10, Mono12	
BU2006MCF	Bayer8, Mono8	85.0
	YUV4:1:1	
	YUV4:2:2	
	Bayer10,Bayer12	
	RGB,BGR	97.6

※ T1 は、ノーマルシャッタ動作時と同じです。
※ T3 は、Typical 値です。

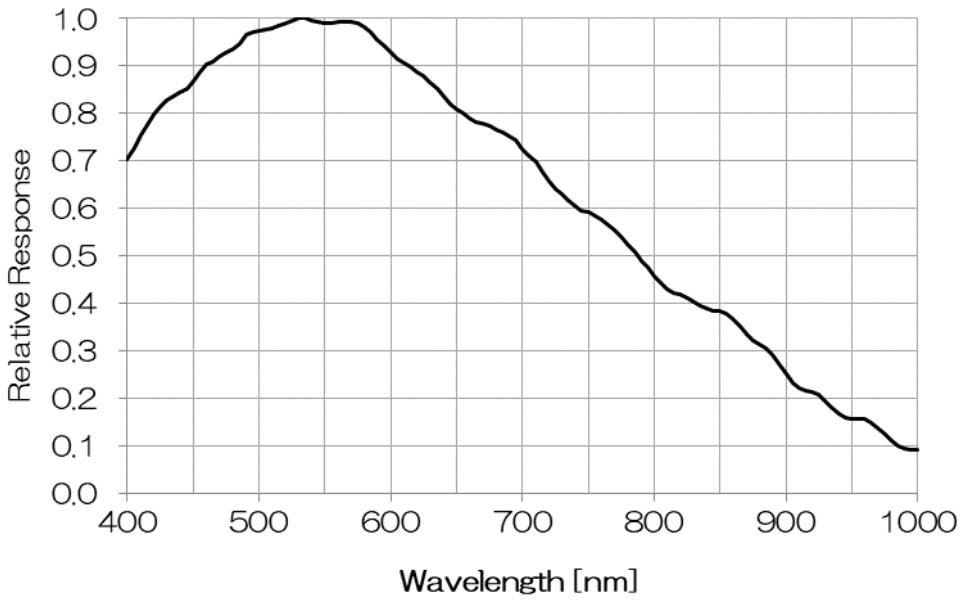
お願い：ランダムトリガシャッタにおける注意点

- FRAME_TRIGGER_WAIT(GPIO 出力信号参照)が inactive の期間は、トリガ信号を入力しないでください。
- 入力されるトリガ信号の周期が極端に短い場合、トリガ信号にノイズがのっている場合に誤動作を起こす可能性があります。トリガ信号生成回路において十分な配慮をお願いいたします。

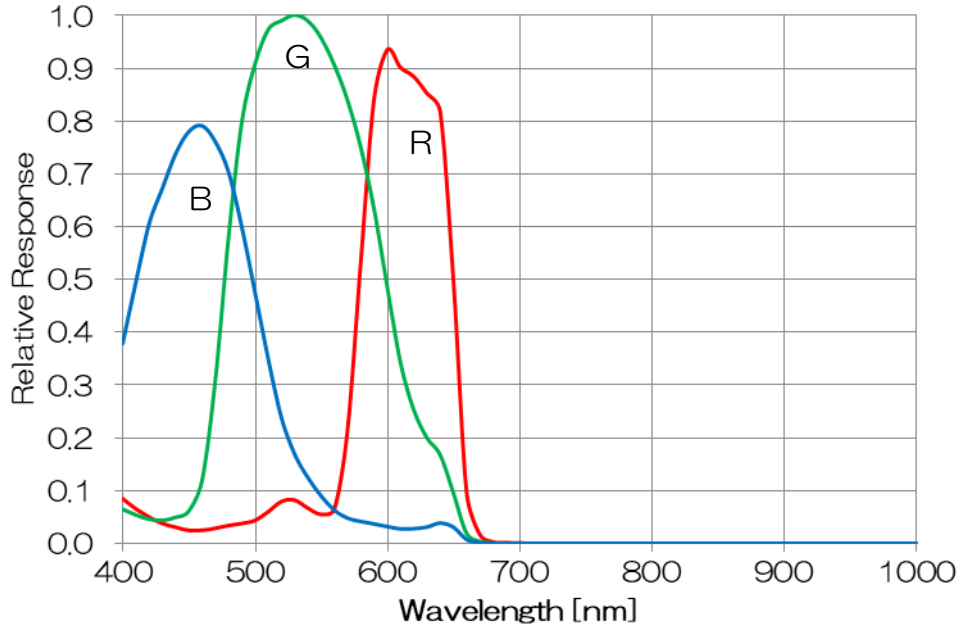
代表的分光感度特性

※ レンズ特性及び光源特性を除く

<BU2006MG>



<BU2006MCF>



使用環境条件

● 温湿度条件

・動作保証温度

周囲温度	：0℃ ～ 40℃、但し 筐体表面温度 60℃以下
湿度	：10% ～ 90% (非結露)

・保存温湿度

周囲温度	：-20℃ ～ 60℃
湿度	：90% 以下（非結露）

お願い：筐体の放熱について

本製品の筐体上面温度は原則 60℃以下でご使用ください。お客様の設置状況に応じて放熱対策を実施して頂くようお願い致します。

● EMC 条件

・EMI (電磁妨害)　　：EN61000-6-4
　　FCC Part 15 Subpart B Class A

・EMS (電磁感受性)　　：EN61000-6-2

お願い：EMC 規格の適合性について

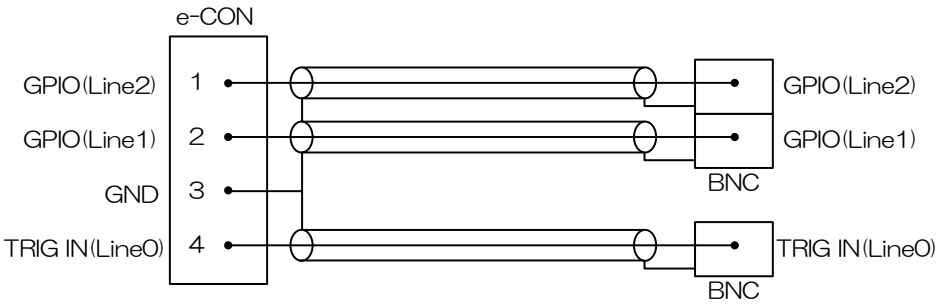
本製品の EMC 規格の適合性については、下記パーツと組み合わせた条件において確認しております。

・USB ケーブル	USB3-KR1-A-MBS-030	（沖電線製）
・e-CON ケーブル	シールドケーブル（3.0m）	（自社製作）

使用部品

・e-CON コネクタ	37104-3163-000 FL	（スリーエムジャパン製）
・シールド線	UL1533 (AWG28)	（日立電線製）

接続図



機械・装置全体で最終的な EMC 適合性の確認は、お客様にて実施して頂くようお願い致します。

機能

本カメラシリーズの主な機能について説明します。
BU2006M シリーズに実装されている機能は下記のとおりです。

機能一覧

カテゴリ	機能	
USB3Vision	Bootstrap Registers	USB3Vision 規格レジスタ
DeviceControl	DeviceControl	デバイス情報
ImageFormatControl	ImageFormatSelector	イメージフォーマット選択
	Scalable	スケーラブル
	Binning	ビニング
	Reverse	映像反転
	PixelFormat	ピクセルフォーマット
	TestPattern	テストパターン
AcquisitionControl	AcquisitionControl	映像取得 / 停止
	ImageBuffer	イメージバッファ
	TriggerControl	トリガモード
	ExposureControl	露光制御
DigitalIOControl	DigitalIOControl	GPIO 制御
	AntiGlitch	アンチグリッチ
	AntiChattering	アンチチャタリング
CounterAndTimerControl	TimerControl	Timer0Active 信号制御
AnalogControl	Gain	ゲイン
	BlackLevel	黒レベル
	Gamma	ガンマ補正
	Hue	色相補正
	Saturation	彩度補正
	BalanceRatio	カラーゲイン（R, B Gain）
	BalanceWhiteAuto	ワンプッシュホワイトバランス
	ColorCorrectionMatrix	カラーマトリクス補正
ALCControl	ALCControl	ALC 制御
LUTControl	LUTControl	LUT 制御
UserSetControl	UserSetControl	ユーザー設定の Load / Save
EventControl	EventControl	イベントパケット制御
VenderUniqueControl	LEDIndicatorLuminance	LED 輝度調整
DPCControl	DPCControl	画素欠陥補正
ChunkDataControl	Chunk	Chunk データ

各機種で対応している機能は下記のとおりです。

機能	BU2006MG	BU2006MCF
Bootstrap Registers	○	○
DeviceControl	○	○
ImageFormatSelector	○	○
Scalable	○	○
Reverse	○	○
PixelFormat	○	○
BayerProcessingMode	-	○
TestPattern	○	○
AcquisitionControl	○	○
ImageBuffer	○	○
TriggerControl	○	○
ExposureControl	○	○
DigitalIOControl	○	○
AntiGlitch	○	○
AntiChattering	○	○
TimerControl	○	○
Gain	○	○
BlackLevel	○	○
Gamma	○	○
Hue	-	○
Saturation	-	○
BalanceRatio	-	○
BalanceWhiteAuto	-	○
ColorCorrectionMatrix	-	○
ALCControl	○	○
LUTControl	○	○
UserSetControl	○	○
EventControl	○	○
LEDIndicatorLuminance	○	○
DPCControl	○	○
Chunk	○	○

Bootstrap Registers

本カメラは USB3 Vision を採用しています。
Bootstrap Registers の詳細については、USB3 Vision の規格を参照してください。
AIA (Automated Imaging Association) USB3 Vision ホームページ
http://www.visiononline.org/vision-standards-details.cfm?type=11
以下は、代表的なレジスタです。

- **UserDefinedName**
  カメラ内の不揮発性メモリに任意の文字列を保存できます。
- **StreamEnable**
  ストリームチャンネルのオープン/クローズを行います。
  ストリームチャンネルのオープン/クローズの際は、StreamEnable レジスタ設定の他にアプリケーション側で SDK のコントロールなどが必要になります。詳しくは、TeliCamSDK ライブラリマニュアルを参照してください。
- **EventEnable**
  イベント通知機能を有効にします。
  イベントチャンネルのオープン/クローズの際は、アプリケーション側で SDK のコントロールなどが必要になります。詳しくは、TeliCamSDK ライブラリマニュアルを参照してください。

● 使用するレジスタ

USB3 Vision ABRM					
レジスタ名	Address	GenICam Interface	Length Byte / [bit]	Access	説明
ManufactureName	0x00004	String	64	R	ベンダー名を返します
ModelName	0x00044	String	64	R	カメラモデル名を返します
FamilyName	0x00084	String	64	R	カメラシリーズ名を返します
DeviceVersion	0x000C4	String	64	R	デバイスバージョンを返します
ManufacturerInfo	0x00104	String	64	R	カメラ情報を返します
SerialNumber	0x00144	String	64	R	シリアル番号を返します
UserDefinedName	0x00184	String	64	R/W	UserDefinedName を返します
SBRM Address	0x001D8	Integer	8	R	SBRM の開始アドレスを返します

USB3 Vision SBRM					
レジスタ名	Address	GenICam Interface	Length Byte / [bit]	Access	説明
SIRMAddress	0x10020	Integer	8	R	SIRM の開始アドレスを返します
EIRMAddress	0x1002C	Integer	8	R	EIRM の開始アドレスを返します
IIDC2Address	0x10038	Integer	8	R	IIDC2 の開始アドレスを返します
CurrentSpeed	0x10040	Integer	[3..0]	R	[0]: Low-Speed 接続（非対応） [1]: Full-Speed 接続（非対応） [2]: High-Speed 接続 [3]: Super-Speed 接続

USB3 Vision SIRM					
レジスタ名	Address	GenICam Interface	Length Byte / [bit]	Access	説明
StreamEnable	0x20004	Integer	[0]	R/W	0: 画像データの転送を無効にします 1: 画像データの転送を有効にします
SIRequiredPayloadSize	0x20008	Integer	8	R	画像のペイロードサイズを返します。
SIRequiredLeaderSize	0x20010	Integer	4	R	Leader の最小サイズを返します
SIRequiredTrailerSize	0x20014	Integer	4	R	Trailer の最小サイズを返します
SIMaximumLeaderSize	0x20018	Integer	4	R	Leader の最大サイズを返します
SIPayloadTransferSize	0x2001C	Integer	4	R	1 パケットのペイロードサイズを返します
SIPayloadTransferCount	0x20020	Integer	4	R	1 画像に対する必要パケット数を返します
SIPayloadFinalTransfer1Size	0x20024	Integer	4	R	Final transfer1 のペイロードサイズを返します
SIPayloadFinalTransfer2Size	0x20028	Integer	4	R	Final transfer2 のペイロードサイズを返します
SIMaximumTrailerSize	0x2002C	Integer	4	R	Trailer の最大サイズを返します

USB3 Vision EIRM					
レジスタ名	Address	GenICam Interface	Length Byte / [bit]	Access	説明
EventEnable	0x30000	Integer	[0]	R/W	0: イベント機能を無効にします 1: イベント機能を有効にします

● **備考**

BU シリーズは High-Speed 接続での画像転送には対応していません。

CurrentSpeed レジスタは、カメラが USB2.0 で接続されたことをアプリケーション上で判別するために使用します。

DeviceControl

本カテゴリのレジスタから各種デバイス情報を読むことができます。
また任意のユーザーID の設定が可能です。

● 使用するレジスタ

レジスタ名	Address	GenICam Interface	Length Byte / [bit]	Access	説明
DeviceReset	0x20003C	Command	[0]	W	カメラをリセットします
DeviceVendorName	0x200070	String	16	R	ベンダー名を返します。
DeviceModelName	0x200090	String	16	R	カメラモデル名を返します。
DeviceManufactureInfo	0x2000B0	String	16	R	カメラ情報を返します。
DeviceVersion	0x2000D0	String	16	R	デバイスバージョンを返します。
DeviceID	0x200110	String	16	R	デバイス ID(製造番号) を返します。

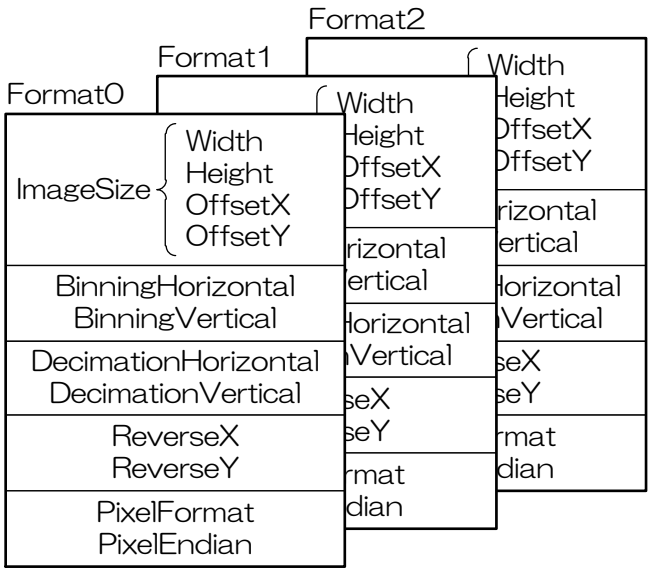
● 備考

- DeviceReset

  DeviceReset の実行によって USB バスのリセットを行います。
  - プラグアンドプレイが発生し、アプリケーションで割り付けられているカメラハンドルは無効にします。
  - カメラのクローズとオープンが必要です。

ImageFormatControl

本カテゴリのレジスタから映像フォーマットに関する制御を行うことができます。
カメラには、3つのイメージフォーマットがあります。ImageFormatSelector レジスタによってイメージフォーマットを選択することができます。



● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
ImageFormatSelector	IEnumeration	4	R/W	映像フォーマットの切り替えを行います。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
ImageFormatSelector	Implemented	0x202020	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x20202C	16	R	[0] : Format0 [1] : Format1 [2] : Format2
	Value	0x20203C	4	R/W	映像フォーマットの切り替えを行います。

ImageFormatSelector によって適用されるレジスタのリスト

ImageFormat 0 - 2
Width
Height
OffsetX
OffsetY
BinningHorizontal
BinningVertical
ReverseX
ReverseY
PixelFormat
PixelEndian

● TeliCamSDK 制御

Camera feature API

専用の API を使用して ImageFormat を制御します。

API 名	説明
GetCamImageFormatSelector	ImageFormatSelector の値を取得します。
SetCamImageFormatSelector	ImageFormatSelector に値を設定します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GenICam API を使用して ImageFormat を制御します。

◆ImageFormat

ImageFormatSelector によって映像フォーマットを選択します。

設定値は Enumeration 型で下記のとおりです。

Integer	String
0 (※)	Format0
1	Format1
2	Format2

※ 出荷設定

```
// GenICam node handle
CAM_NODE_HANDLE   hSelector = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, "ImageFormatSelector", &hSelector);

// ImageFormat = Format2
Nd_SetEnumStrValue(s_hCam, hSelector, "Format2");
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IEnumeration node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして映像フォーマットを制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆ImageFormat

ImageFormatSelector レジスタの Value フィールドに書き込みます。

```
// ImageFormat = Format2
uint32_t      uiSelector;
uiSelector = 2;
Cam_WriteReg(s_hCam, 0x20203C, 1, &uiSelector);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

● 備考

・映像ストリーム出力中は ImageFormatSelector レジスタ設定変更が無効となります。

Scalable

スケーラブル読み出しは、最大映像出力有効画素領域のうち任意の矩形領域のみを読み出し、出力する方法です。

選択できる形状は連続したユニット単位の矩形形状のみで、凸や凹のような選択はできません。また選択できるウィンド数は 1 個です。

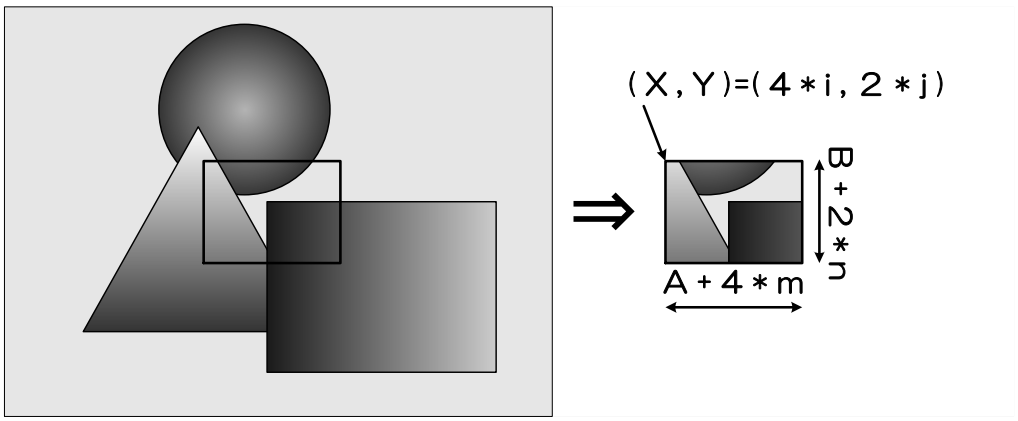
・ウィンドのサイズ　　：{A + 4 × m (H)} × {B + 2 × n (V)}

※　A, B はそれぞれの最小ユニットサイズ

※　m, n は整数、但しウィンドが最大ユニットサイズの全画面からはみ出さないこと

・ウィンドの開始位置　　：{4 x i (H)} x {2 x j (V)}

※　i, j は整数、但しウィンドが最大ユニットサイズの全画面からはみ出さないこと



スケーラブル

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
Width	IInteger	4	R/W	映像の幅を設定します。
Height	IInteger	4	R/W	映像の高さを設定します。
OffsetX	IInteger	4	R/W	映像の水平方向開始位置を設定します。
OffsetY	IInteger	4	R/W	映像の垂直方向開始位置を設定します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
ImageSize	Implemented	0x202060	[31]	R	この機能が有効かどうかを返します。
	OffsetXMin	0x20206C	4	R	水平方向開始位置の最小値を返します。
	OffsetXInc	0x202070	4	R	水平方向開始位置の設定できる単位を返します。
	WidthMin	0x202074	4	R	幅の最小値を返します。
	WidthInc	0x202078	4	R	ユニットサイズの幅を返します。
	SensorWidth	0x20207C	4	R	センサの有効画素幅を返します。
	OffsetYMin	0x202080	4	R	垂直方向開始位置の最小値を返します。
	OffsetYInc	0x202084	4	R	垂直方向開始位置の設定できる単位を返します。
	HeightMin	0x202088	4	R	高さの最小値を返します。
	HeightInc	0x20208C	4	R	ユニットサイズの高さを返します。
	SensorHeight	0x202090	4	R	センサの有効画素高さを返します。
	OffsetX	0x202094	4	RW	映像の水平方向開始位置を設定します。
	Width	0x202098	4	RW	映像の幅を設定します。
	OffsetY	0x20209C	4	RW	映像の垂直方向開始位置を設定します。
	Height	0x2020A0	4	RW	映像の高さを設定します。

● TeliCamSDK 制御

Camera feature API

専用 API を使用してスケーラブルを制御します。

API 名	説明
GetCamSensorWidth	センサの有効画素幅を取得します。
GetCamSensorHeight	センサの有効画素高さを取得します。
GetCamRoi	カメラの ROI を取得します。
SetCamRoi	カメラの ROI を設定します。
GetCamWidthMinMax	映像の幅の最小値と最大値を取得します。
GetCamWidth	映像の幅を取得します。
SetCamWidth	映像の幅を設定します。
GetCamHeightMinMax	映像の高さの最小値と最大値を取得します。
GetCamHeight	映像の高さを取得します。
SetCamHeight	映像の高さを設定します。
GetCamOffsetXMinMax	映像の水平開始位置の最小値と最大値を取得します。
GetCamOffsetX	映像の水平開始位置を取得します。
SetCamOffsetX	映像の水平開始位置を設定します。
GetCamOffsetYMinMax	映像の垂直開始位置の最小値と最大値を取得します。
GetCamOffsetY	映像の垂直開始位置を取得します。
SetCamOffsetY	映像の垂直開始位置を設定します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用してスケーラブルを制御します。

◆Scalable

```
// GenICam node handle
CAM_NODE_HANDLE   hWidth = NULL;
CAM_NODE_HANDLE   hHeight = NULL;
CAM_NODE_HANDLE   hOffsetX = NULL;
CAM_NODE_HANDLE   hOffsetY = NULL;

// ROI = {OffsetX, Width, OffsetY, Height};
uint64_t      ROI[] = {612,1224, 512,1024};

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “Width”, &hWidth);
Nd_GetNode(s_hCam, “Height”, &hHeight);
Nd_GetNode(s_hCam, “OffsetX”, &hOffsetX);
Nd_GetNode(s_hCam, “OffsetY”, &hOffsetY);

// Set ROI
Nd_SetIntValue(s_hCam, hWidth, ROI[1]);
Nd_SetIntValue(s_hCam, hOffsetX, ROI[0]);
Nd_SetIntValue(s_hCam, hHeight, ROI[3]);
Nd_SetIntValue(s_hCam, hOffsetY, ROI[2]);
```

映像の幅を小さくする場合は、最初に Width を設定し、その後、OffsetX を設定します。

映像の幅を大きくする場合は、最初に OffsetX を設定し、その後、Width を設定します。

映像の高さを小さくする場合は、最初に Height を設定し、その後、OffsetY を設定します。

映像の高さを大きくする場合は、最初に OffsetY を設定し、その後、Height を設定します

詳細は[TeliCamAPI Library manual]の[INode functions], [IInteger node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスしてスケーラブルを制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆Scalable

OffsetX, Width, OffsetY, Height レジスタの Value フィールドに書き込みます。

```
// ROI = {OffsetX, Width, OffsetY, Height};
uint32_t      ROI[] = {612,1224, 512,1024};

// Set ROI (in one by one)
Cam_WriteReg(s_hCam, 0x202094, 1, &ROI[0]);
Cam_WriteReg(s_hCam, 0x202098, 1, &ROI[1]);
Cam_WriteReg(s_hCam, 0x20209C, 1, &ROI[2]);
Cam_WriteReg(s_hCam, 0x2020A0, 1, &ROI[3]);

// Set ROI (in block)
Cam_WriteReg(s_hCam, 0x202094, 4, &ROI[0]);
```

IIDC2 レジスタアクセスでは
OffsetX, Width, OffsetY, Height を任意の順番で設定できます。(one by one access)
OffsetX, Width, OffsetY, Height を 1 回のアクセスで設定することも可能です。(block access)

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

◆最小値／最大値

Scalable	BU2006MG	BU2006MCF
Width/OffsetX 設定単位	4	4
Height/OffsetY 設定単位	2	2
最小ユニットサイズ	64 x 64	64 x 64
最大ユニットサイズ (※)	5472 x 3648	5472 x 3648

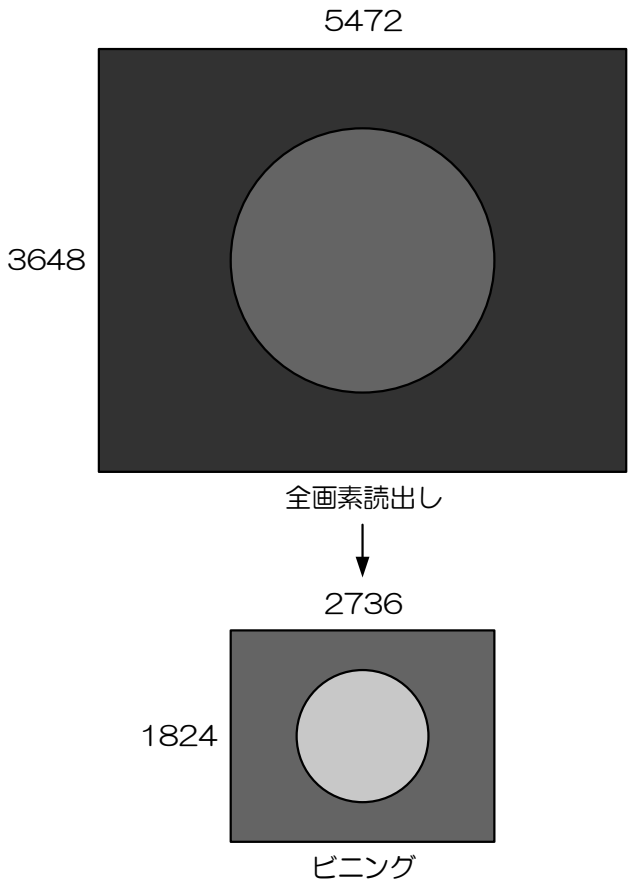
※ 出荷設定

● **備考**

・映像ストリーム出力中は、Width, Height, OffsetX, OffsetY レジスタ設定変更が無効となります。

Binning

CMOS センサによるビニング機能をサポートしています。水平・垂直方向の隣り合う画素を加重加算後、平均して読み出します。ビニング読み出しではさらに高速なフレームレートで出力することができ、S/N を向上させることができます。



ビニング動作のイメージ（2x2 ビニング）

お願い：ビニング使用時の欠陥画素補正について
本製品では工場出荷時に全画素出力モードにてあらかじめ CMOS センサの欠陥画素を登録し、補正を行っています。ビニングモード使用時は CMOS センサの加重加算により、想定されない座標に欠陥画素が現れる場合があります。運用上問題がある場合は、欠陥画素を追加で登録するなどの措置をお願い致します。

● ビニング時の各出力フォーマットにおけるフレームレート(fps)

型名	フォーマット	最大フレームレート
BU2006MG	Mono8	53.6 fps
	Mono10、Mono12	38.0 fps
BU2006MCF	Bayer8、Mono8	53.6 fps
	YUV 4:1:1	50.7 fps
	YUV 4:2:2	38.0 fps
	Bayer10、Bayer12	38.0 fps
	RGB、BGR	25.1 fps

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
BinningHorizontal	IInteger	4	R/W	水平方向のビニングライン数を設定します。
BinningVertical	IInteger	4	R/W	垂直方向のビニングライン数を設定します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
BinningHorizontal	Implemented	0x202120	[31]	R	この機能が有効かどうかを返します。
	Min	0x202134	4	R	水平方向のビニングライン数の最小値を返します。
	Max	0x202138	4	R	水平方向のビニングライン数の最大値を返します。
	Value	0x20213C	4	R/W	水平方向のビニングライン数を設定します。
BinningVertical	Implemented	0x202140	[31]	R	この機能が有効かどうかを返します。
	Min	0x202154	4	R	垂直方向のビニングライン数の最小値を返します。
	Max	0x202158	4	R	垂直方向のビニングライン数の最大値を返します。
	Value	0x20215C	4	R/W	垂直方向のビニングライン数を設定します。

● TeliCamSDK 制御

Camera feature API

専用 API を使用して Binning を制御します。

API 名	説明
GetCamBinningHorizontalMinMax	水平方向のビニングラインの最小値と最大値を取得します。
GetCamBinningHorizontal	水平方向のビニングライン数を取得します。
SetCamBinningHorizontal	水平方向のビニングライン数を設定します。
GetCamBinningVerticalMinMax	垂直方向のビニングラインの最小値と最大値を取得します。
GetCamBinningVertical	垂直方向のビニングライン数を取得します。
SetCamBinningVertical	垂直方向のビニングライン数を設定します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用して Binning を制御します。

◆Binning

```
// GenICam node handle
CAM_NODE_HANDLE   hBinning = NULL;

// Binning = 2x2
uint64_t      Binning = 2;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “BinningHorizontal”, &hBinning);
// Nd_GetNode(s_hCam, “BinningVertical”, &hBinning); // either will do

// Set Binning
Nd_SetIntValue(s_hCam, hBinning, Binning);
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IInteger node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして Binning を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆Binning

BinningHorizontal レジスタ または BinningVertical レジスタの Value フィールドに書き込みます。

```
// Binning = 2x2
uint32_t      Binning = 2;

// Set Binning
Cam_WriteReg(s_hCam, 0x20213C, 1, &Binning);
// Cam_WriteReg(s_hCam, 0x20215C, 1, &Binning); // either will do
```

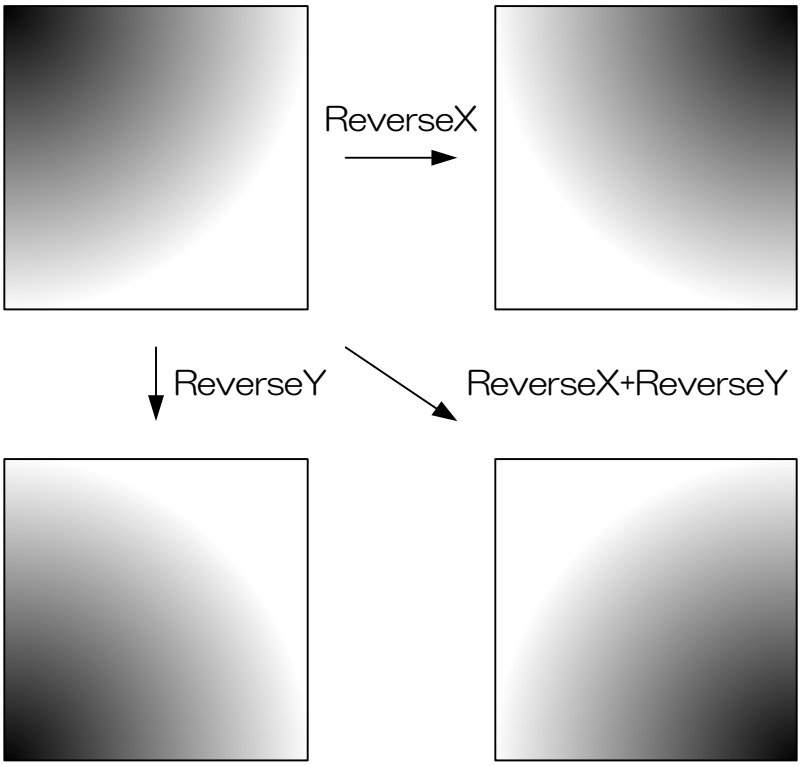
詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

● **備考**

- 映像ストリーム出力中は BinningHorizontal および BinningVertical レジスタ設定変更が無効となります。
- BU2006MG/BU2006MCF では 2×2 のみ対応しています。

Reverse

映像出力を水平方向、垂直方向に反転することができます。



● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
ReverseX	IBoolean	4	R/W	水平方向の映像反転を行います。
ReverseY	IBoolean	4	R/W	垂直方向の映像反転を行います。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
ReverseX	Implemented	0x2021A0	[31]	R	この機能が有効かどうかを返します。
	Value	0x2021B0	4	R/W	水平方向の映像反転を行います。. [0] : Off [1] : On
ReverseY	Implemented	0x2021C0	[31]	R	この機能が有効かどうかを返します。
	Value	0x2021D0	4	R/W	垂直方向の映像反転を行います。 [0] : Off [1] : On

● TeliCamSDK 制御

Camera feature API

専用の API を使用して Reverse を制御します。

API 名	説明
GetCamReverseX	ReverseX の値を取得します。
SetCamReverseX	ReverseX に値を設定します。
GetCamReverseY	ReverseY の値を取得します。
SetCamReverseY	ReverseY に値を設定します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用して Reverse を制御します。

◆Reverse

ReverseX で水平方向の映像反転を設定します。

ReverseY で垂直方向の映像反転を設定します。

```
// GenICam node handle
CAM_NODE_HANDLE   hReverseX = NULL;
CAM_NODE_HANDLE   hReverseY = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “ReverseX”, &hReverseX);
Nd_GetNode(s_hCam, “ReverseY”, &hReverseY);

// Set Reverse (flip horizontal and vertical direction)
Nd_SetBoolValue(s_hCam, hReverseX, true);
Nd_SetBoolValue(s_hCam, hReverseY, true);
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IBoolean node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして Reverse を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆Reverse

ReverseX レジスタの Value フィールドに書き込みます。

ReverseY レジスタの Value フィールドに書き込みます。

```
// Set Reverse (flip horizontal and vertical direction)
uint32_t      dat = 1;
Cam_WriteReg(s_hCam, 0x2021B0, 1, &dat);
Cam_WriteReg(s_hCam, 0x2021D0, 1, &dat);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

● **備考**

• 映像ストリーム出力中は ReverseX および ReverseY レジスタ設定変更が無効となります。

PixelFormat

映像ストリームのピクセルフォーマットを選択します。

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
PixelCoding	IEnumeration	4	R/W	ピクセルコーディングを選択します。
PixelSize	IEnumeration	4	R/W	映像画素のビットサイズを選択します。
PixelFormat	IEnumeration	4	R/W	ピクセルフォーマットを選択します。 ピクセルフォーマットは AIA の Pixel Format Naming Convention に準拠します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
PixelCoding	Implemented	0x2020C0	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x2020CC ～0x2020D8	16	R	[0] : Mono [32] : RGB [34] : RGBPacked (obsolete) [40] : BGR [42] : BGRPacked (obsolete) [66] : YUV411Packed [74] : YUV422Packed [96] : BayerGR [99] : BayerRG [102] : BayerGB [105] : BayerBG
	Value	0x2020DC	4	R/W	ピクセルコーディングを選択します。
PixelSize	Implemented	0x2020E0	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x2020EC ～0x2020F8	16	R	[8] : Bpp8 [10] : Bpp10 [12] : Bpp12 [16] : Bpp16 [24] : Bpp24
	Value	0x2020FC	4	R/W	映像画素のビットサイズを選択します。

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
PixelFormat	Implemented	0x202400	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x20240C ～0x202418	16	R	[1] : Mono8 [3] : Mono10 [5] : Mono12 [8] : BayerGR8 [9] : BayerRG8 [10] : BayerGB8 [11] : BayerBG8 [12] : BayerGR10 [13] : BayerRG10 [14] : BayerGB10 [15] : BayerBG10 [16] : BayerGR12 [17] : BayerRG12 [18] : BayerGB12 [19] : BayerBG12 [20] : RGB8 [21] : BGR8 [30] : YUV411Packed [31] : YUV422Packed
	Value	0x20241C	4	R/W	ピクセルフォーマットを選択します。

● ピクセルフォーマット一覧

◆白黒モデル

PixelCoding \ PixelSize	Bpp8	Bpp10	Bpp12
Mono	Mono8 (※)	Mono10	Mono12
PixelFormat ID	0x01080001	0x01100003	0x01100005

※ 出荷設定

◆カラーモデル

PixelCoding \ PixelSize	Bpp8	Bpp10	Bpp12	Bpp16	Bpp24
Mono	Mono8	-	-	-	-
PixelFormat ID	0x01080001				
BayerBG(※2)	BayerBG8	BayerBG10	BayerBG12	-	-
PixelFormat ID	0x0108000B	0x0110000F	0x01100013		
YUV411	-	-	YUV411Packed	-	-
PixelFormat ID			0x020C001E		
YUV422	-	-	-	YUV422Packed	-
PixelFormat ID				0x0210001F	
RGB	-	-	-	-	RGB8 (※1)
PixelFormat ID					0x02180014
BGR	-	-	-	-	BGR8
PixelFormat ID					0x02180015

※1 出荷設定

※2 ReverseX、ReverseY 設定は False

◆Reverse 時の Bayer フォーマット

ReverseY \ ReverseX	FALSE		TRUE	
FALSE	BayerBG		BayerGB	
	Bpp8	0x0108000B	Bpp8	0x0108000A
	Bpp10	0x0110000F	Bpp10	0x0110000E
	Bpp12	0x01100013	Bpp12	0x01100012
TRUE	BayerGR		BayerRG	
	Bpp8	0x01080008	Bpp8	0x01080009
	Bpp10	0x0110000C	Bpp10	0x0110000D
	Bpp12	0x01100010	Bpp12	0x01100011

● TeliCamSDK 制御

Camera feature API

専用の API を使用して PixelFormat を制御します。

API 名	説明
GetCamPixelFormat	PixelFormat の値を取得します。
SetCamPixelFormat	PixelFormat に値を設定します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用して PixelFormat を制御します。

◆PixelCoding/PixelSize

PixelCoding レジスタと PixelSize レジスタの組み合せで PixelFormat を決定します。

1.PixelCoding にてピクセルコーディングを選択します。

設定値は Enumeration 型で下記のとおりです。

Integer	String
0	Mono
32	RGB
34	RGBPacked
40	BGR
42	BGRPacked
66	YUV411Packed
74	YUV422Packed
96	BayerGR
99	BayerRG
102	BayerGB
105	BayerBG

2.PixelSize にて映像画素のビットサイズを選択します。

設定値は Enumeration 型で下記のとおりです。

Integer	String
8	Bpp8
10	Bpp10
12	Bpp12
16	Bpp16
24	Bpp24

```
// GenICam node handle
CAM_NODE_HANDLE   hCoding = NULL;
CAM_NODE_HANDLE   hSize = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, "PixelCoding", &hCoding);
Nd_GetNode(s_hCam, "PixelSize", &hSize);

// 1.Select a pixel coding.
Nd_SetEnumStrValue(s_hCam, hCoding, "Mono");
// 2.Select a pixel size.
Nd_SetEnumStrValue(s_hCam, hSize, "Bpp10");
```

◆PixelFormat

PixelFormat にてピクセルフォーマットを選択します。

設定値は Enumeration 型で下記のとおりです。

Integer	String	Integer	String
1	Mono8	15	BayerBG10
3	Mono10	16	BayerGR12
5	Mono12	17	BayerRG12
8	BayerGR8	18	BayerGB12
9	BayerRG8	19	BayerBG12
10	BayerGB8	20	RGB8
11	BayerBG8	21	BGR8
12	BayerGR10	30	YUV411Packed
13	BayerRG10	31	YUV422Packed
14	BayerGB10		

```
// GenICam node handle
CAM_NODE_HANDLE   hFormat = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, "PixelFormat", &hFormat);

// 1.Select a pixel format.
Nd_SetEnumStrValue(s_hCam, hFormat, "Mono10");
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IEnumeration node functions]を参照してください。.

Register access API

IIDC2 レジスタに直接アクセスして PixelFormat を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆PixelCoding/PixelSize

PixelCoding レジスタと PixelSize レジスタの組み合せで PixelFormat を決定します。

1.PixelCoding レジスタの Value フィールドに書き込みます。

2.PixelSize レジスタの Value フィールドに書き込みます。

```
uint32_t coding = 0; // Mono
uint32_t size = 10; // Bpp10

// 1.Select a pixel coding.
Cam_WriteReg(s_hCam, 0x2020DC, 1, &coding);
// 2.Select a pixel size.
Cam_WriteReg(s_hCam, 0x2020FC, 1, &size);
```

◆PixelFormat

PixelFormat レジスタの Value フィールドに書き込みます。

```
uint32_t format = 3; // Mono10

// 1.Select a pixel format.
Cam_WriteReg(s_hCam, 0x20241C, 1, &format);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

● **備考**

• 映像ストリーム中は PixelCoding および PixelSize レジスタ設定変更が無効となります。

BayerProcessingMode

カラーモデルでは BayerProcessingMode を切替可能です。
カラープロセス機能の対応表は下記になります。

BayerProcessingMode	説明
Full	全機能が使用可能です
Partial	Partial 機能が使用可能です
Raw	Gain 設定のみ使用可能です

Function	Full (※)	Partial	Raw
Gain	○	○	○
BlackLevel	○	○	-
Gamma	○	○	-
Hue	○	-	-
Saturation	○	-	-
BalanceRatio	○	○	-
ColorCorrectionMatrix	○	-	-
LUTControl	○	○	-
DPCControl	○	○	-

※ 出荷設定

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
BayerProcessingMode	IEnumeration	4	R/W	Bayer Processing Mode を選択します.

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
BayerProcessingMode	Implemented	0x21F420	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x21F42C ～0x21F438	16	R	[0] : Raw [8] : Partial [16] : Full
	Value	0x21F43C	4	R/W	Bayer Processing Mode を選択します.

● TeliCamSDK 制御

GenICam function API

GeniCam API を使用して BayerProcessingMode を制御します。

◆BayerProcessingMode

BayerProcessingMode で BayerProcessingMode を選択します。

設定値は Integer 型または String 型で下記のとおりです。

Integer	String
0	Raw
8	Partial
16	Full

```
// GenICam node handle
CAM_NODE_HANDLE   hMode = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, "BayerProcessingMode", &hMode);

// BayerProcessingMode = Raw
Nd_SetEnumStrValue(s_hCam, hMode, "Raw");
```

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして PixelFormat を制御します。.

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆BayerProcessingMode

BayerProcessingMode レジスタの Value フィールドに書き込みます。

```
// BayerProcessingMode = Raw
uint32_t      uiMode;
uiMode = 0;
Cam_WriteReg(s_hCam, 0x21F43C, 1, &uiMode);
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IEnumeration node functions]を参照してください。

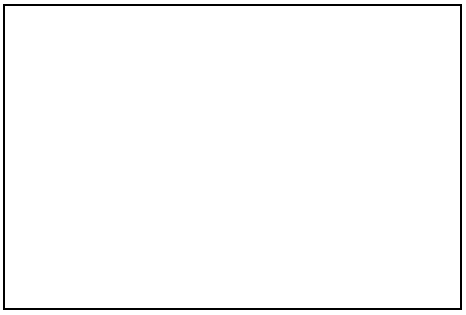
● 備考

・映像ストリーム中は BayerProcessingMode レジスタ設定変更が無効となります。

TestPattern

本カメラはテストパターン出力をサポートしています。サポートしているパターンは以下のとおりです。

Black = 全画面 0 LSB @ 8bit

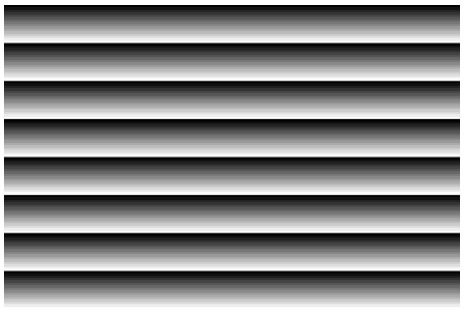


White = 全画面 255LSB @ 8bit

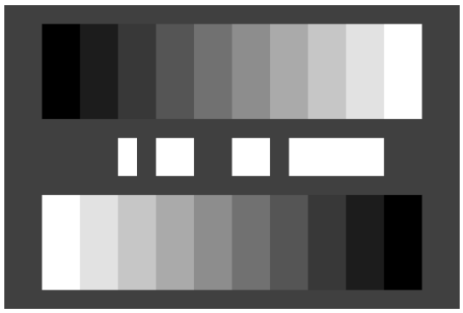
GreyA = 全画面 170LSB @ 8bit

GreyB = 85LSB @ 8bit

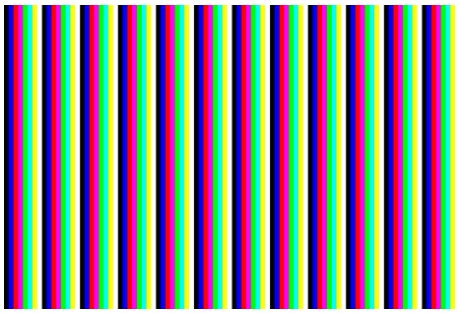
GreyHorizontalRamp = 水平ランプ



GreyVerticalRamp = 垂直ランプ



GreyScale = グレースケール
（白黒モデルのみ）



ColorBar = カラーバー
（カラーモデルのみ）

テストパターン （BU2006MG/BU2006MCF）

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
TestPattern	IEnumeration	4	R/W	テストパターンを選択します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
TestPattern	Implemented	0x21F120	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x21F12C ～0x21F138	16	R	[0] : Off [1] : Black [2] : White [3] : GrayA [4] : GrayB [5] : GreyHorizontalRamp [6] : GrayScale [7] : ColorBar [8] : GreyVerticalRamp
	Value	0x21F13C	4	R/W	テストパターンを選択します。

● TeliCamSDK 制御

Camera feature API

専用の API を使用して TestPattern を制御します。

API 名	説明
GetCamTestPattern	TestPattern の値を取得します。
SetCamTestPattern	TestPattern に値を設定します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用して TestPatterm を制御します。

◆TestPattern

テストパターンを選択します。

設定値は Enumeration 型で下記のとおりです。

Integer	String	説明
0 (※)	Off	テストパターン Off、通常映像
1	Black	全てのピクセルが 0
2	White	全てのピクセルが 255 @Mono8
3	GreyA	全てのピクセルが 170 @Mono8
4	GreyB	全てのピクセルが 85 @Mono8
5	GreyHorizontalRamp	水平方向ランプ
6	GreyVerticalRamp	垂直方向ランプ
7	GreyScale	グレースケール
8	ColorBar	カラーバー

※ 出荷設定

```
// GenICam node handle
CAM_NODE_HANDLE   hNode = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, "TestPattern", & hNode);

// 1.Select a test pattern.
Nd_SetEnumStrValue(s_hCam, hNode, "GreyHorizontalRamp");
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IEnumeration node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして TestPattern を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆TestPattern

TestPattern レジスタの Value フィールドに書き込みます。

```
uint32_t dat = 5; // Horizontal Ramp

// 1.Select a test pattern.
Cam_WriteReg(s_hCam, 0x21F13C, 1, &dat);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

ImageBuffer

ImageBuffer はイメージバッファに画像を取り込んでおき、任意のタイミングで読み出しを行うことができます。

この機能はノーマルシャッタモードでも動作しますが、通常ランダムトリガモードにて使用します。

TriggerControl の項目も参照ください。

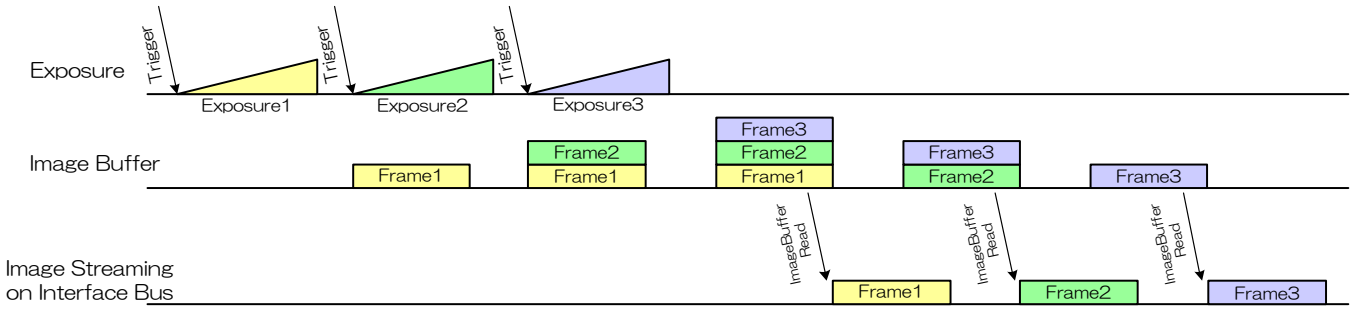


Image Buffer

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
ImageBufferMode	IEnumeration	4	R/W	イメージバッファモードを有効にします。
ImageBufferFrameCount	IInteger	4	R/W	イメージバッファに取り込まれた画像枚数を返します。
ImageBufferRead	ICommand	4	W	イメージバッファから画像を読み出します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
ImageBufferMode	Implemented	0x203060	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x20306C ~0x203078	16	R	[0] : Off [1] : On
	Value	0x20307C	4	R/W	イメージバッファモードを有効にします。
ImageBufferFrameCount	Implemented	0x203080	[31]	R	この機能が有効かどうかを返します。
	Min	0x203094	4	R	イメージバッファモードの最小画像枚数を返します。
	Max	0x203098	4	R	イメージバッファモードの最大画像枚数を返します。
	Value	0x20309C	4	R	イメージバッファに取り込まれた画像枚数を返します。

● TeliCamSDK 制御

Camera feature API

専用の API を使用して ImageBuffer を制御します。

API 名	説明
GetCamImageBufferMode	ImageBuffer の値を取得します。
SetCamImageBufferMode	ImageBuffer モードを設定します。
GetCamImageBufferFrameCount	ImageBufferFrameCount の値を取得します。
ExecuteCamImageBufferRead	Image Buffer から画像を読み出します。

◆ImageBuffer

1.イメージバッファモードを有効にします。

SetCamImageBufferMode 関数によって ImageBufferMode を設定します。

2.映像ストリームをキャプチャします。

画像のキャプチャ開始／停止の方法は AcquisitionControl と同じです。

AcquisitionControl の項目も参照してください。

3.イメージバッファに取り込まれた画像枚数を読み出します。

GetCamImageBufferFrameCount 関数によってイメージバッファ内の画像枚数を読み出します。

4.画像を読み出します。

ExecuteCamImageBufferRead 関数によってイメージバッファから画像を読み出します。

5.イメージバッファから画像を受信します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用して ImageBuffer を制御します。

◆ImageBuffer

1.ImageBufferMode によってイメージバッファモードを有効にします。

設定値は Enumeration 型で下記のとおりです。

Integer	String
0 (※)	Off
1	On

※ 出荷設定

2.ストリームチャンネルのオープン／クローズ。

画像のキャプチャ開始／停止の方法は AcquisitionControl と同じです。

AcquisitionControl の項目も参照してください。

3.ImageBufferFrameCount によってバッファに取り込まれた画像枚数を読み出します。

4.ImageBufferRead によってバッファ内の画像を読み出します。

5.イメージバッファから画像を受信します。

6.イメージストリームのストップ／クローズ。

```
// GenICam node handle
CAM_NODE_HANDLE   hMode = NULL;
CAM_NODE_HANDLE   hCount = NULL;
CAM_NODE_HANDLE   hRead = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, "ImageBufferMode", &hMode);
Nd_GetNode(s_hCam, "ImageBufferFrameCount", &hCount);
Nd_GetNode(s_hCam, "ImageBufferRead", &hRead);

// 1.Select an Image Buffer mode
Nd_SetEnumStrValue(s_hCam, hMode, "On");

// 2.Open and Start image stream.
//   2.1.Set Trigger mode
        SetCamTriggerMode(s_hCam, true);
        SetCamTriggerSource(s_hCam, CAM_TRIGGER_SOFTWARE);
//   2.2.Open Stream
        s_hStrmEvt = CreateEvent(NULL, FALSE, FALSE, NULL);
        Strm_OpenSimple(s_hCam, &s_hStrm, &s_uiImgBufSize,
s_hStrmEvt);
        s_pucImgBuf = (uint8_t *)VirtualAlloc(NULL, s_uiImgBufSize,
MEM_RESERVE | MEM_COMMIT, PAGE_EXECUTE_READWRITE);
//   2.3.Stream Start
        Strm_Start(s_hStrm);
//   2.4.Execute Software Trigger
        ExecuteCamSoftwareTrigger(s_hCam);

// 3.Read the number of frames in Image Buffer by
'ImageBufferFrameCount'.
int64_t count = 0;
while(count==0)
{
        Nd_GetIntValue(s_hCam, hCount, &count);
}

// 4.Read Image from Image Buffer by 'ImageBufferRead'.
Nd_CmdExecute(s_hCam, hRead);

// 5.Receive Image from Image Buffer
        uint32_t uiSize   = s_uiImgBufSize;
        WaitForSingleObject(s_hStrmEvt, 1000);
        Strm_ReadCurrentImage(s_hStrm, s_pucImgBuf, &uiSize, NULL);

// 6.Stop and Close image stream.
//   6.1.Stream Stop
        Strm_Stop(s_hStrm);
//   6.2.Close Stream
        Strm_Close(s_hStrm);
        CloseHandle(s_hStrmEvt);
        VirtualFree(s_pucImgBuf, 0, MEM_RELEASE);
```

Register access API

IIDC2 レジスタに直接アクセスして ImageBuffer を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆ImageBuffer

1.イメージバッファモードを有効にします。

ImageBufferMode を制御するために ImageBufferMode レジスタの Value フィールドに書き込みます。

2.ストリームチャンネルのオープン／クローズ。

画像のキャプチャ開始／停止の方法は AcquisitionControl と同じです。

AcquisitionControl の項目も参照してください。

3.イメージバッファに取り込まれた画像枚数を読み出します。

ImageBufferFrameCount レジスタのレジスタの Value フィールドを読み出します。

4.イメージバッファに取り込まれた画像を読み出します

ImageBufferRead を実行すために AcquisitionCommand レジスタの Value フィールドに[10]を書き込みます。

AcquisitionControl の AcquisitionCommand の項目も参照してください。

5.イメージバッファから画像を受信します。

6.イメージストリームのストップ／クローズ。

```
// 1.Select an Image Buffer mode
uint32_t      dat = 1;
Cam_WriteReg(s_hCam, 0x20307C, 1, &dat);

// 2.Open and Start image stream.
//   2.1.Set Trigger mode
            SetCamTriggerMode(s_hCam, true);
            SetCamTriggerSource(s_hCam, CAM_TRIGGER_SOFTWARE);
//   2.2.Open Stream
            s_hStrmEvt = CreateEvent(NULL, FALSE, FALSE, NULL);
            Strm_OpenSimple(s_hCam, &s_hStrm, &s_uiImgBufSize,
s_hStrmEvt);
            s_pucImgBuf = (uint8_t *)VirtualAlloc(NULL, s_uiImgBufSize,
MEM_RESERVE | MEM_COMMIT, PAGE_EXECUTE_READWRITE);
//   2.3.Stream Start
            Strm_Start(s_hStrm);
//   2.4.Execute Software Trigger
            ExecuteCamSoftwareTrigger(s_hCam);

// 3.Read the number of frames in Image Buffer.
uint32_t count = 0;
while(count==0)
{
            Cam_ReadReg(s_hCam, 0x20309C, 1, &count);
}

// 4.Read Image from Image Buffer. AcquisitionCommand = 10 : Image
Buffer Read
dat = 10;
Cam_WriteReg(s_hCam, 0x20303C, 1, &dat);

// 5.Receive Image from Image Buffer
            uint32_t uiSize    = s_uiImgBufSize;
            WaitForSingleObject(s_hStrmEvt, 1000);
            Strm_ReadCurrentImage(s_hStrm, s_pucImgBuf, &uiSize, NULL);

// 6.Stop and Close image stream.
//   6.1.Stream Stop
            Strm_Stop(s_hStrm);
//   6.2.Close Stream
            Strm_Close(s_hStrm);
            CloseHandle(s_hStrmEvt);
            VirtualFree(s_pucImgBuf, 0, MEM_RELEASE);
```

詳細[TeliCamAPI Library manual]の[Camera functions]を参照してください。

● **備考**

- バッファに取り込める画像枚数は画像サイズによって異なります。（最大 256MByte）
- ImageBufferRead コマンドで転送するフレーム数は、AcquisitionFrameCount で決まります。
- 映像ストリーム出力中は ImageBufferMode レジスタ設定変更が無効となります。

TriggerControl

露光動作には、フリーランで動作するノーマルシャッタモードと外部からのトリガにより任意のタイミングで動作するランダムトリガシャッタモードの２種類があります。
ランダムトリガシャッタモードは次の２とおりのトリガ入力で動作します。

・カメラ背面の I/O コネクタから入力されるトリガ（ハードウェアトリガ）
・USB3.1 Gen1 インターフェースを経由して入力されるトリガ（ソフトウェアトリガ）

カメラの動作モードをまとめると以下のようになります。

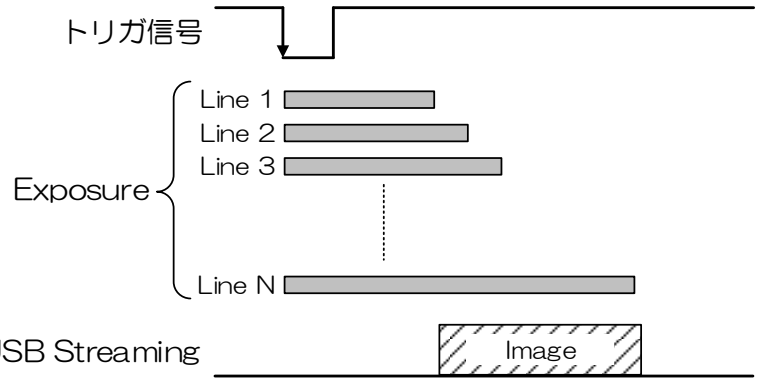
動作モード

トリガ動作モード	同期	露光制御
ノーマルシャッタ	フリーラン(内部同期)	ExposureTime レジスタ制御
ランダムトリガシャッタ	ハードウェアトリガ	・Edge モード:TriggerSequence0 ・Bulk モード:TriggerSequence6 ExposureTime レジスタ制御
	ソフトウェアトリガ	・Edge モード:TriggerSequence0 ・Bulk モード:TriggerSequence6 ExposureTime レジスタ制御

※上記以外の動作モードの組み合わせについては保証いたしません。

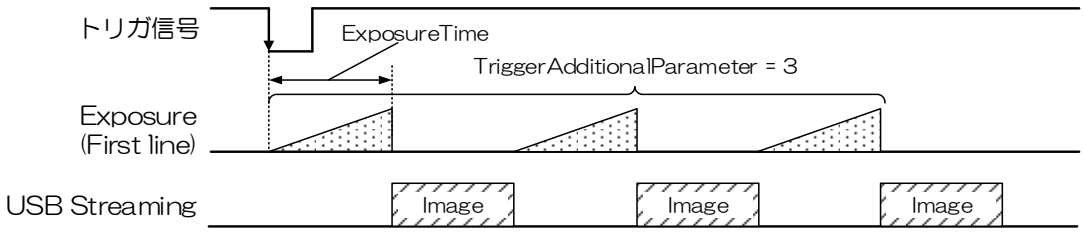
• Edge モード：TriggerSequence0

露光時間は電子シャッタの設定値

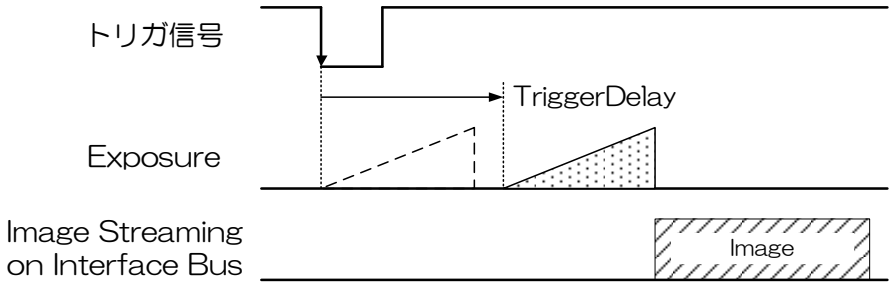


• Bulk モード（FrameBurstTrigger）：TriggerSequence6

1 回の外部トリガ信号入力で、連続して複数回の露光と映像出力を行います。



ハードウェアトリガは入力信号のエッジで動作し、その極性はレジスタ設定によって変更できます。また有効エッジから露光開始するまでの時間に任意の遅延時間を付加することが可能です。



トリガディレイ

なお、ランダムトリガシャッタで動作させた場合、外部トリガを入力してから露光を開始するまでに遅延時間が発生します。“仕様”の“タイミング”をご参照ください。

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
TriggerMode	IEnumeration	4	R/W	カメラのトリガ動作モードを設定します。
TriggerSoftware	ICommand	4	W	ソフトウェアトリガを実行します。
TriggerSource	IEnumeration	4	R/W	ランダムトリガシャッタのトリガソースを選択します。
TriggerActivation	IEnumeration	4	R/W	ハードウェアトリガの有効エッジを選択します。
TriggerDelay	IFloat	4	R/W	トリガ信号検出から露光開始までの遅延時間を設定します。
TriggerSequence	IEnumeration	4	R/W	露光時間の制御モードを選択します。
TriggerAdditionalParameter	IInteger	4	R/W	Bulk モード時の露光回数を設定します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
TriggerMode	Implemented	0x207020	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x20702C ~0x207038	16	R	[0] : OFF (ノーマルシャッタ) [1] : ON (ランダムトリガシャッタ)
	Value	0x20703C	4	R/W	カメラのトリガ動作モードを設定します。
TriggerSequence	Implemented	0x207040	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x20704C ~0x207058	16	R	[0] : TriggerSequence0 (Edge モード) [6] : TriggerSequence6 (Bulk モード)
	Value	0x20705C	4	R/W	露光時間の制御モードを選択します。
TriggerSource	Implemented	0x207060	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x20706C ~0x207078	16	R	[0] : Line0 (ハードウェアトリガ) [2] : Line2 (ハードウェアトリガ) [64] : Software (ソフトウェアトリガ)
	Value	0x20707C	4	R/W	ランダムトリガシャッタのトリガソースを選択します。
TriggerAdditionalParameter	Implemented	0x207080	[31]	R	この機能が有効かどうかを返します。
	Min	0x207094	4	R	Bulk モード時の最小露光回数を返します。
	Max	0x207098	4	R	Bulk モード時の最大露光回数を返します。
	Value	0x20709C	4	R/W	Bulk モード時の露光回数を設定します。

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
TriggerDelay	Implemented	0x2070A0	[31]	R	この機能が有効かどうかを返します。
	Mult	0x2070AC	4	R	絶対値［sec］= Raw * (Mult / Div)
	Div	0x2070B0	4	R	
	Min	0x2070B4	4	R	遅延時間の最小値を返します。
	Max	0x2070B8	4	R	遅延時間の最大値を返します。
	Value	0x2070BC	4	R/W	トリガ信号検出から露光開始までの遅延時間を設定します。
SoftwareTrigger	Implemented	0x207040	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x20702C ～0x207038	16	R	[0]：Inactive [8]：Impulse
	Value	0x20705C	4	R/W	ソフトウェアトリガを実行します。

● TeliCamSDK 制御

Camera feature API

専用の API を使用してトリガ動作を制御します。

API 名	説明
GetCamTriggerMode	TriggerMode を取得します。
SetCamTriggerMode	TriggerMode を設定します。
GetCamTriggerSequence	TriggerSequence を取得します。
SetCamTriggerSequence	TriggerSequence を設定します。
GetCamTriggerSource	TriggerSource を取得します。
SetCamTriggerSource	TriggerSource を設定します。
GetCamTriggerAdditionalParameterMinMax	Bulk モード時の露光回数の最小値と最大値を取得します。
GetCamTriggerAdditionalParameter	Bulk モード時の露光回数を取得します。
SetCamTriggerAdditionalParameter	Bulk モード時の露光回数を設定します。
GetCamTriggerDelayMinMax	遅延時間の最小値と最大値を取得します。
GetCamTriggerDelay	遅延時間を取得します。
SetCamTriggerDelay	遅延時間を設定します。
ExecuteCamSoftwareTrigger	Software Trigger を実行します。

1.トリガ動作モードを切り替えます。
SetCamTriggerMode 関数によって TriggerMode を設定します。
2.露光制御モードを切り替えます。
SetCamTriggerSequence 関数によって TriggerSequence を設定します。
3.トリガソースを切り替えます。
SetCamTriggerSource 関数によって TriggerSource を設定します。
4.ハードウェアトリガの有効エッジを選択します。
ハードトリガの極性は SetCamLineInverter で決定されます。
DigitalIOControl’の SetCamLineInverter’の項目も参照してください。
5.露光回数を設定します。（Bulk モード時）
SetCamTriggerAdditionalParameter 関数によって TriggerAdditionalParameter を設定します。
6.トリガディレイを設定します。
SetCamTriggerDelay 関数によって TriggerDelay を設定します。
7.映像ストリームをキャプチャします。
画像のキャプチャ開始／停止の方法は AcquisitionControl と同じです。
AcquisitionControl の項目も参照してください。
TeliCamSDK インストールフォルダ内の[TeliCamAPI Library manual]の
[Camera streaming functions]と[GrabStreamSimple]サンプルコードを参照してください。
8.ソフトウェアトリガによって映像を取得します。
ExecuteCamSoftwareTrigger 関数によって SoftwareTrigger モードでソフトウェアトリガを実行します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用してトリガ動作を制御します。

1.TriggerMode によってトリガ動作モードを有効にします。

設定値は Enumeration 型で下記のとおりです。

Integer	String
0 (※)	Off
1	On

※ 出荷設定

2.TriggerSequence によって露光時間の制御モードを選択します。

設定値は Enumeration 型で下記のとおりです。

Integer	String
0 (※)	TriggerSequence0
6	TriggerSequence6

※ 出荷設定

3.TriggerSource によってトリガソースを選択します。

設定値は Enumeration 型で下記のとおりです。

Integer	String
0 (※)	Line0
2	Line2
64	Software

※ 出荷設定

4.TriggerActivation によってハードウェアトリガの有効エッジを選択します。

ハードトリガの極性は SetCamLineInverter で決定されます。

DigitalIOControl’の SetCamLineInverter’の項目も参照してください。

5.TriggerAdditionalParameter によって露光回数を設定します。（Bulk モード時）

6.TriggerDelay によってトリガディレイを設定します。

7.映像ストリームのキャプチャします

画像のキャプチャ開始／停止の方法は AcquisitionControl と同じです。

AcquisitionControl の項目も参照してください。

TeliCamSDK インストールフォルダ内の[TeliCamAPI Library manual]の[Camera streaming functions]と[GrabStreamSimple]サンプルコードを参照してください。

8.TriggerSoftware によって SoftwareTrigger モードでソフトウェアトリガを実行します。

9.映像を受信します。

10.映像ストリームのストップ／クローズ。

```
// GenICam node handle
CAM_NODE_HANDLE   hMode = NULL;
CAM_NODE_HANDLE   hSequence = NULL;
CAM_NODE_HANDLE   hSource = NULL;
CAM_NODE_HANDLE   hAdditionalParameter = NULL;
CAM_NODE_HANDLE   hDelay = NULL;
CAM_NODE_HANDLE   hTriggerSoftware = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, "TriggerMode", &hMode);
Nd_GetNode(s_hCam, "TriggerSequence", &hSequence);
Nd_GetNode(s_hCam, "TriggerSource", &hSource);
Nd_GetNode(s_hCam, "TriggerAdditionalParameter", &hAdditionalParameter);
Nd_GetNode(s_hCam, "TriggerDelay", &hDelay);
Nd_GetNode(s_hCam, "TriggerSoftware", &hTriggerSoftware);

// 1. Select a trigger mode by 'TriggerMode'.
Nd_SetEnumStrValue(s_hCam, hMode, "On");

// 2. Select a trigger sequence of random trigger shutter by 'TriggerSequence'.
Nd_SetEnumStrValue(s_hCam, hSequence, "TriggerSequence6");          // Bulk mode

// 3. Select a trigger source of random trigger shutter by 'TriggerSource'.
Nd_SetEnumStrValue(s_hCam, hSource, "Software");                    // Software

// 4.Select a trigger activation of hardware trigger by 'LineInverterAll'.
//   -- not applicable for Software Trigger

// 5.Set the number of frames to exposure in Bulk mode by 'TriggerAdditionalParameter'.
Nd_SetIntValue(s_hCam, hAdditionalParameter, 3);                    // 3 frames

// 6.Set a trigger delay by 'TriggerDelay'.
Nd_SetFloatValue(s_hCam, hDelay, 1000.0);                           // 1ms

// 7.Open and Start image stream.
//   7.1.Open Stream
            s_hStrmEvt = CreateEvent(NULL, FALSE, FALSE, NULL);
            Strm_OpenSimple(s_hCam, &s_hStrm, &s_uiImgBufSize, s_hStrmEvt);
            s_pucImgBuf = (uint8_t *)VirtualAlloc(NULL, s_uiImgBufSize, MEM_RESERVE |
MEM_COMMIT, PAGE_EXECUTE_READWRITE);
//   7.2.Stream Start
            Strm_Start(s_hStrm);

// 8.Execute software trigger in SoftwareTrigger mode by 'TriggerSoftware'.
Nd_CmdExecute(s_hCam,hTriggerSoftware);

// 9.Receive Image
uint32_t uiSize   = s_uiImgBufSize;
CAM_IMAGE_INFO sImageInfo;
for(int i=0; i<3; i++)
{
            WaitForSingleObject(s_hStrmEvt, 1000);
            Strm_ReadCurrentImage(s_hStrm, s_pucImgBuf, &uiSize, &sImageInfo);
}
// 10.Stop and Close image stream.
//   10.1.Stream Stop
            Strm_Stop(s_hStrm);
//   10.2.Close Stream
            Strm_Close(s_hStrm);
            CloseHandle(s_hStrmEvt);
            VirtualFree(s_pucImgBuf, 0, MEM_RELEASE);
```


詳細は[TeliCamAPI Library manual]の[INode functions], [IEnumeration node functions], [ICommand node functions], [IInteger node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスしてトリガ動作を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

1.トリガ動作モードを切り替えます。
　TriggerMode を制御するために TriggerMode レジスタの Value フィールドに書き込みます。
2.露光制御モードを切り替えます。
　TriggerSequence を制御するために TriggerSequence レジスタの Value フィールドに書き込みます。
3.トリガソースを切り替えます。
　TriggerSource を制御するために TriggerSource レジスタの Value フィールドに書き込みます。
4.ハードウェアトリガの有効エッジを選択します。
　ハードトリガの極性は SetCamLineInverter で決定されます。
　DigitalIOControl’の SetCamLineInverter’の項目も参照してください。
5.露光回数を設定します。（Bulk モード時）
　TriggerAdditionalParameter を制御するために TriggerAdditionalParameter レジスタの Value フィールドに書き込みます。
6.トリガディレイを設定します。
　TriggerDelay を制御するために TriggerDelay レジスタの Value フィールドに書き込みます。
7.映像ストリームのオープン／スタート。
　画像のキャプチャ開始／停止の方法は AcquisitionControl と同じです。
　AcquisitionControl の項目も参照してください。
　TeliCamSDK インストールフォルダ内の[TeliCamAPI Library manual]の[Camera streaming functions]と[GrabStreamSimple]サンプルコードを参照してください。
8.ソフトウェアトリガによって映像を取得します。
　SoftwareTrigger を制御するために SoftwareTrigger レジスタの Value フィールドに[8]を書き込みます。
9.映像を受信します。
10.映像ストリームのストップ／クローズ。

```
uint32_t      dat;

// 1. Select a trigger mode by 'TriggerMode'.
dat = 1;
Cam_WriteReg(s_hCam, 0x20703C, 1, &dat); // TriggerMode = On

// 2. Select a trigger sequence of random trigger shutter by
'TriggerSequence'.
dat = 6;
Cam_WriteReg(s_hCam, 0x20705C, 1, &dat); // TriggerSequence6 (Bulk
mode)

// 3. Select a trigger source of random trigger shutter by 'TriggerSource'.
dat = 64;
Cam_WriteReg(s_hCam, 0x20707C, 1, &dat); // Software

// 4.Select a trigger activation of hardware trigger by 'LineInverterAll'.
//   -- not applicable for Software Trigger

// 5.Set the number of frames to exposure in Bulk mode by
'TriggerAdditionalParameter'.
dat = 3;
Cam_WriteReg(s_hCam, 0x20709C, 1, &dat); // 3 frames

// 6.Set a trigger delay by 'TriggerDelay'.
// TriggerDelay = 1000.0us (Raw value = 60000)
dat = 60000;
Cam_WriteReg(s_hCam, 0x2070BC, 1, &dat); // 1ms

// 7.Open and Start image stream.
//   7.1.Open Stream
            s_hStrmEvt = CreateEvent(NULL, FALSE, FALSE, NULL);
            Strm_OpenSimple(s_hCam, &s_hStrm, &s_uiImgBufSize,
s_hStrmEvt);
            s_pucImgBuf = (uint8_t *)VirtualAlloc(NULL, s_uiImgBufSize,
MEM_RESERVE | MEM_COMMIT, PAGE_EXECUTE_READWRITE);
//   7.2.Stream Start
            Strm_Start(s_hStrm);

// 8.Execute software trigger in SoftwareTrigger mode by
'SoftwareTrigger'.
dat = 8;
Cam_WriteReg(s_hCam, 0x2070DC, 1, &dat); // SoftwareTrigger

// 9.Receive Image
uint32_t uiSize   = s_uiImgBufSize;
CAM_IMAGE_INFO sImageInfo;
for(int i=0; i<3; i++)
{
            WaitForSingleObject(s_hStrmEvt, 1000);
            Strm_ReadCurrentImage(s_hStrm, s_pucImgBuf, &uiSize,
&sImageInfo);
}

// 10.Stop and Close image stream.
//   10.1.Stream Stop
            Strm_Stop(s_hStrm);
//   10.2.Close Stream
            Strm_Close(s_hStrm);
            CloseHandle(s_hStrmEvt);
            VirtualFree(s_pucImgBuf, 0, MEM_RELEASE);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

◆最小値／最大値

TriggerAdditionalParameter	Raw 値 = 絶対値(Float)
最小値	0
最大値	255
初期値	0

TriggerDelay	Raw 値	絶対値(Float)
最小値	0	0.0 [us]
最大値	120000000	2000000.0 [us]
初期値	0	0.0 [us]
式	絶対値 [us] = Raw 値 / 60	

● 備考

• ソフトウェアトリガ動作時の TriggerSoftware 実行～映像取得の遅延時間は不定となります。

• BU シリーズには 2 つのハードウェアトリガのトリガソースがあります。
I/O 入出力信号仕様の項目も参照してください。

TriggerSource	説明
Line0 (※)	I/O コネクタ：4 pin, LVTTL High Level 2.0 to 24.0V
Line2	I/O コネクタ：1 pin, 5V CMOS High Level 4.0V to 5.0V

※ 出荷設定

• TriggerDelay はハードウェアトリガとソフトウェアトリガの両方に適用されます。

• TriggerAdditionalParameter レジスタ設定は、Bulk モード時のみ有効です。

ExposureTime

ExposureTime はイメージセンサが光にさらされる（露出する）時間を制御します。

制御方式として、任意の露光時間を設定するマニュアル露光時間制御(MANUAL)、被写体の明るさに合わせて露光時間を自動で調整する自動露光時間制御(AE)、電子シャッタ機能を OFF する NoSpecify モードがあります。

- NoSpecify　：AcquisitionFrameRate によるフレームレート設定の露光時間で動作します。
- Manual　：レジスタに設定した任意の露光時間で動作します。
- Auto　：被写体の明るさに合わせて露光時間を自動で調整し動作します。

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
ExposureTime	IFloat	4	R/W	Manual 動作時の露光時間を設定します。
ExposureAuto	IEnumeration	4	R/W	露光時間の制御モードを選択します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
ExposureTime	Implemented	0x204020	[31]	R	この機能が有効かどうかを返します。
	Control	0x204028	4	R/W	[0]: NoSpecify [1]: Manual [2]: Auto
	Mult	0x20402C	4	R	絶対値［sec］= Raw * (Mult / Div)
	Div	0x204030	4	R	
	Min	0x204034	4	R	露光時間の最小値を返します。
	Max	0x204038	4	R	露光時間の最大値を返します。
	Value	0x20403C	4	R/W	Manual 動作時の露光時間を設定します。

● TeliCamSDK 制御

Camera feature API

専用の API を使用して ExposureTime を制御します。

API 名	説明
GetCamExposureTimeMinMax	露光時間の最小値と最大値を取得します。
GetCamExposureTime	露光時間を取得します。
SetCamExposureTime	露光時間を設定します。
GetCamExposureTimeControl	露光時間の制御モードを取得します。
SetCamExposureTimeControl	露光時間の制御モードを設定します。

ExposureTimeControl パラメータ	説明
CAM_EXPOSURE_TIME_CONTROL_NO_SPECIFY	NoSpecify
CAM_EXPOSURE_TIME_CONTROL_MANUAL	Manual
CAM_EXPOSURE_TIME_CONTROL_AUTO	Auto

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用して ExposureTime を制御します。

◆ExposureTime

ExposureTime によって露光時間を設定します。

```
// GenICam node handle
CAM_NODE_HANDLE   hMode = NULL;
CAM_NODE_HANDLE   hSelector = NULL;
CAM_NODE_HANDLE   hSource = NULL;

// GenICam node handle
CAM_NODE_HANDLE   hNode = NULL;

// ExposureTime = 1000us
float64_t      dExposureTime = 1000.0;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, "ExposureTime", &hNode);

// Set ExposureTime Value
Nd_SetFloatValue(s_hCam, hNode, dExposureTime);
```

◆ExposureAuto

ExposureAuto によって露光時間の制御モードを選択します。

設定値は Enumeration 型で下記のとおりです。

Integer	String	説明
0	DeviceSpecific	NoSpecify モード
1 (※)	Off	Manual モード
2	Continuous	Auto モード

※ 出荷設定

```
// GenICam node handle
CAM_NODE_HANDLE   hNode = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, "ExposureAuto", &hNode);

// Set ExposureAuto = "Continuous"
Nd_SetEnumStrValue(s_hCam, hNode, "Continuous");
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IFloat node functions], [IEnumeration node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして ExposureTime を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆ExposureTime

ExposureTime レジスタの Value フィールドに書き込みます。

```
// ExposureTime = 1000us (Raw value = 60000)
uint32_t      uiExposureTimeRaw = 60000;

// Set ExposureTime Value
Cam_WriteReg(s_hCam, 0x20403C, 1, &uiExposureTimeRaw);
```

◆ExposureAuto

ExposureTime レジスタの Control フィールドに書き込みます。

```
// ExposureAuto = " Continuous "
uint32_t      uiExposureAuto = 2;

// Set ExposureAuto = " Continuous "
Cam_WriteReg(s_hCam, 0x204028, 1, &uiExposureAuto);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。.

◆最小値／最大値

ExposureTime				BU2006MG
最小値	全画素読み出し時		Raw 値	4
			絶対値	56.444 [us]
	ビニング時		Raw 値	4
			絶対値	20.111[us]
最大値	Manual	全画素読み出し時	Raw 値	1133859
			絶対値	16000010.333[us]
		ビニング時	Raw 値	3182321
			絶対値	16000002.805[us]
	Auto	全画素読み出し時	Raw 値	70867
			絶対値	1000012.111 [us]
		ビニング時	Raw 値	198896
			絶対値	1000004.899[us]
初期値			Raw 値	3685
			絶対値	51999.444 [us]
式				絶対値 [us] = Raw 値 x 14.1111 ビニング時：絶対値 [us] = Raw 値 x 5.0277

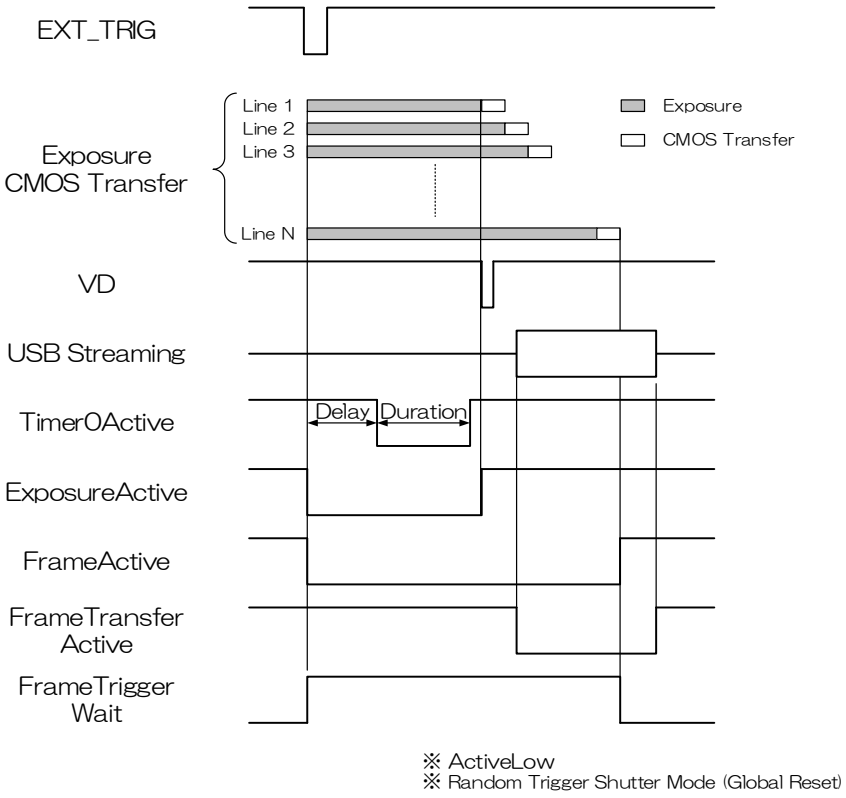
ExposureTime					BU2006MCF
最小値	全画素読み出し時		Bayer8、Mono8 YUV 4:1:1 YUV 4:2:2 Bayer10、Bayer12	Raw 値	4
				絶対値	56.444 [us]
			RGB、BGR	Raw 値	4
				絶対値	64.722 [us]
	ビニング時			Raw 値	4
				絶対値	20.111[us]
最大値	Manual	全画素読み出し時	Bayer8、Mono8 YUV 4:1:1 YUV 4:2:2 Bayer10、Bayer12	Raw 値	1133859
				絶対値	16000010.333[us]
			RGB、BGR	Raw 値	988842
				絶対値	16000012.917 [us]
		ビニング時		Raw 値	3182321
				絶対値	16000002.805[us]
	Auto	全画素読み出し時	Bayer8、Mono8 YUV 4:1:1 YUV 4:2:2 Bayer10、Bayer12	Raw 値	70867
				絶対値	1000012.111 [us]
			RGB、BGR	Raw 値	61803
				絶対値	1000006.875[us]
		ビニング時		Raw 値	198896
				絶対値	1000004.899[us]
初期値				Raw 値	3213
				絶対値	51988.125 [us]
式					絶対値 [us] = Raw 値 x 14.1111 RGB、BGR 時：絶対値 [us] = Raw 値 x 16.1806 ビニング時：絶対値 [us] = Raw 値 x 5.0277

● 備考

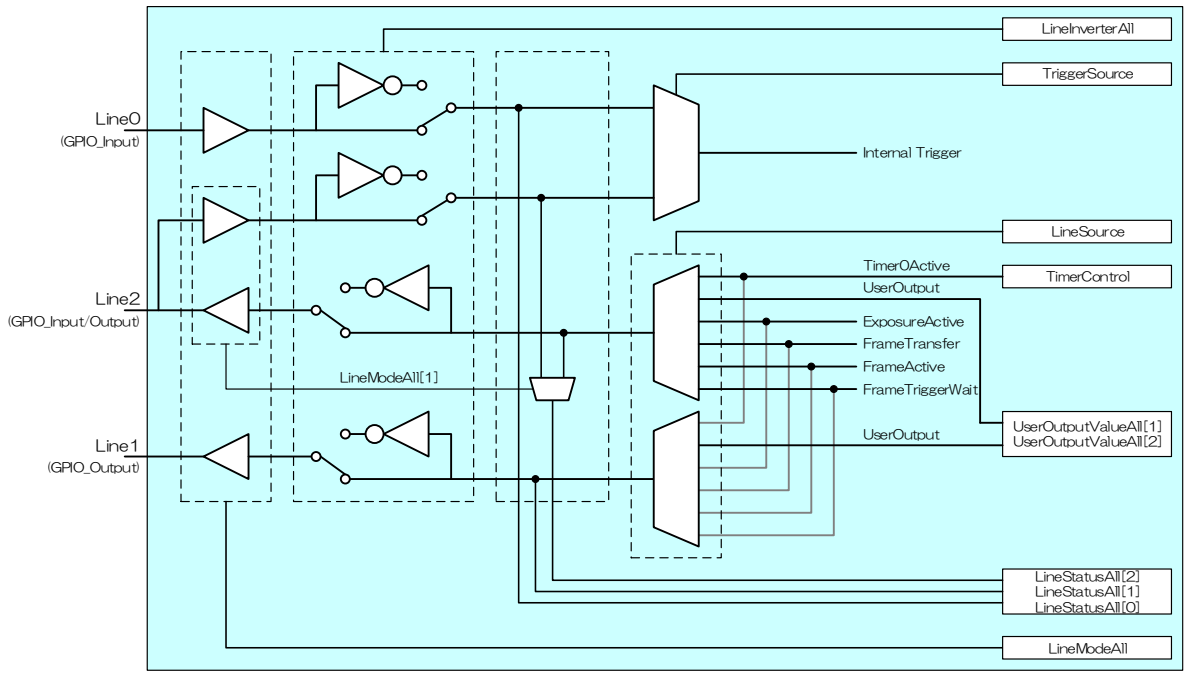
・ExposureAuto モードでは、現在の露光時間の値が ExposureTime レジスタに設定されます。

DigitalIOControl

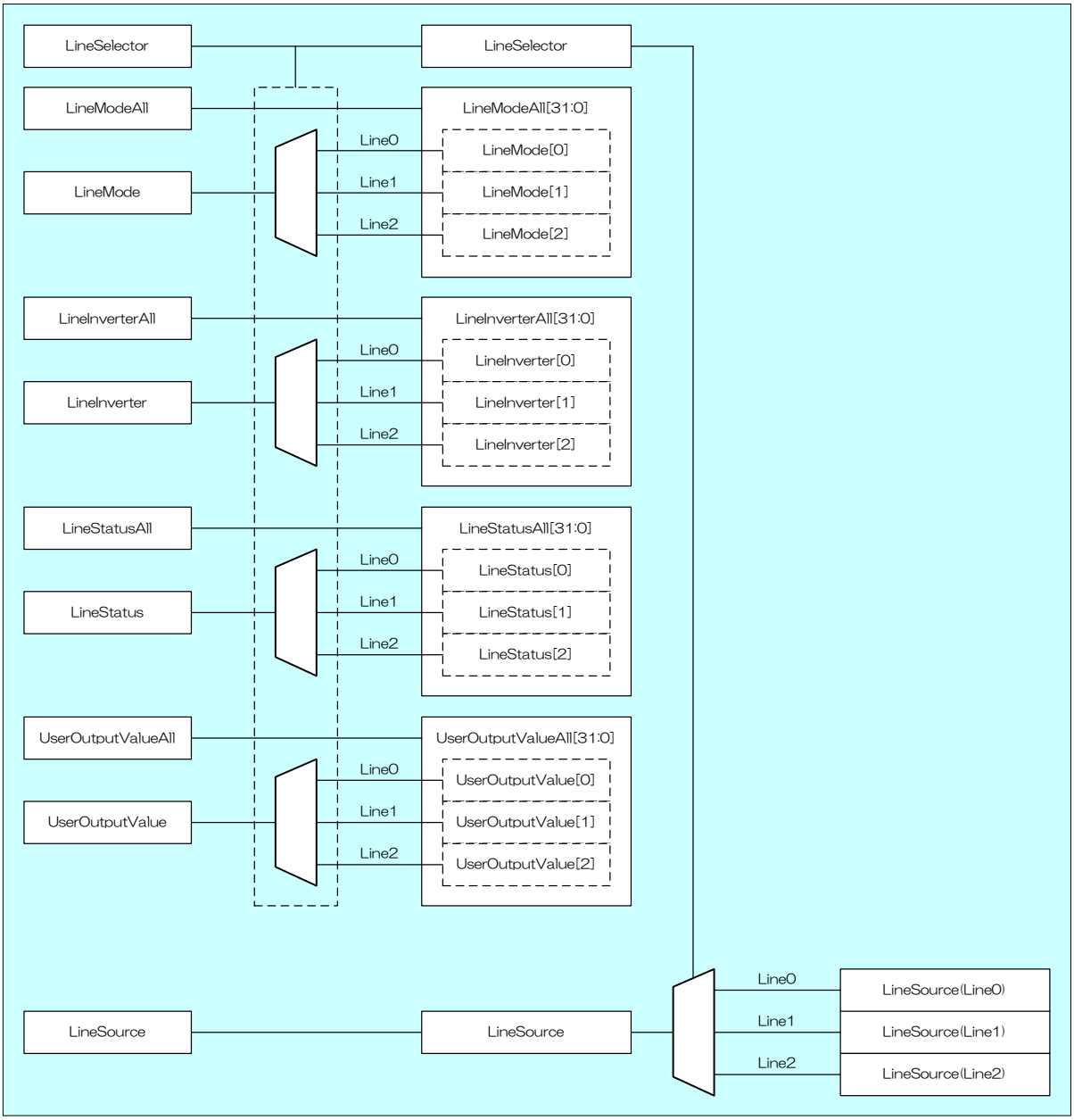
I/O コネクタ汎用出力ピンから出力する信号を選択することができます。また出力信号の極性を切り替えることができます。出力信号の仕様は下図のとおりです。



汎用出力信号仕様



GPIO 内部回路構成



各信号の概念(全体図)

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
LineMode	IEnumeration	4	R/W	LineSelector で選択した LINE の入出力を選択します。
LineModeAll	IInteger	4	R/W	LINE の入出力を選択します。
LineInverter	IBoolean	4	R/W	LineSelector で選択した LINE の極性を選択します。
LineInverterAll	IInteger	4	R/W	LINE の極性を選択します。
LineStatus	IBoolean	4	R	LineSelector で選択した LINE の状態を返します。
LineStatusAll	IInteger	4	R	LINE の状態を返します。
UserOutputValue	IBoolean	4	R/W	LineSelector で選択した LINE のユーザー設定を選択します。
UserOutputValueAll	IInteger	4	R/W	LINE 出力のユーザー設定値を設定します。
LineSelector	IEnumeration	4	R/W	LINE を選択します。
LineSource	IEnumeration	4	R/W	LINE の信号種類を選択します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
LineModeAll	Implemented	0x209020	[31]	R	この機能が有効かどうかを返します。
	BitWritable	0x20902C	4	R	[0]: Line0 [1]: Line1 [2]: Line2
	Value	0x209030	4	R/W	LINE の入出力を選択します。
LineInverterAll	Implemented	0x209040	[31]	R	この機能が有効かどうかを返します。
	Value	0x209050	4	R/W	LINE の極性を選択します。入出力両方に反映します。
LineStatusAll	Implemented	0x209060	[31]	R	この機能が有効かどうかを返します。
	Value	0x209070	4	R	LINE の状態を返します。
UserOutputValueAll	Implemented	0x209080	[31]	R	この機能が有効かどうかを返します。
	Value	0x209090	4	R/W	LINE 出力のユーザー設定値を設定します。
LineSelector	Implemented	0x2090A0	[31]	R	この機能が有効かどうかを返します。
	Value	0x2090BC	4	R/W	LINE を選択します。
LineSource	Implemented	0x2090C0	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x2090CC	4	R	[0] : Off [32] : UserOutput [64] : Timer0Active [99] : AcquisitionActive [106] : FrameTriggerWait [107] : FrameActive [115] : FrameTransferActive [123] : ExposureActive
	Value	0x2090DC	4	R/W	LINE の信号種類を選択します。

● TeliCamSDK 制御

Camera feature API

専用の API を使用して DigitalIOControl を制御します。

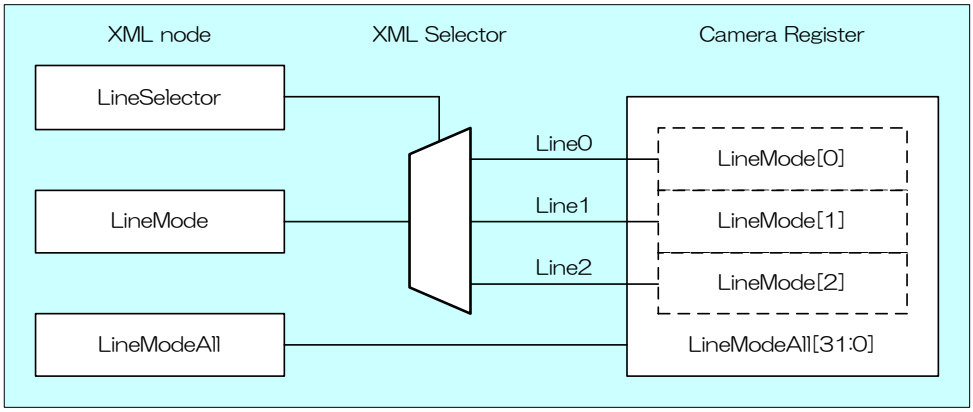
API 名	説明
GetCamLineModeAll	LineMode の値を取得します。
GetCamLineInverterAll	LineInverter の値を取得します。
SetCamLineInverterAll	LineInverter に値を設定します。
GetCamLineStatusAll	LineStatus の値を取得します。
GetCamUserOutputValueAll	UserOutput の値を取得します。
SetCamUserOutputValueAll	UserOutput に値を設定します。
GetCamLineSource	LineSource の値を取得します。
SetCamLineSource	LineSource に値を設定します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用して DigitalIOControl を制御します。

◆LineModeAll



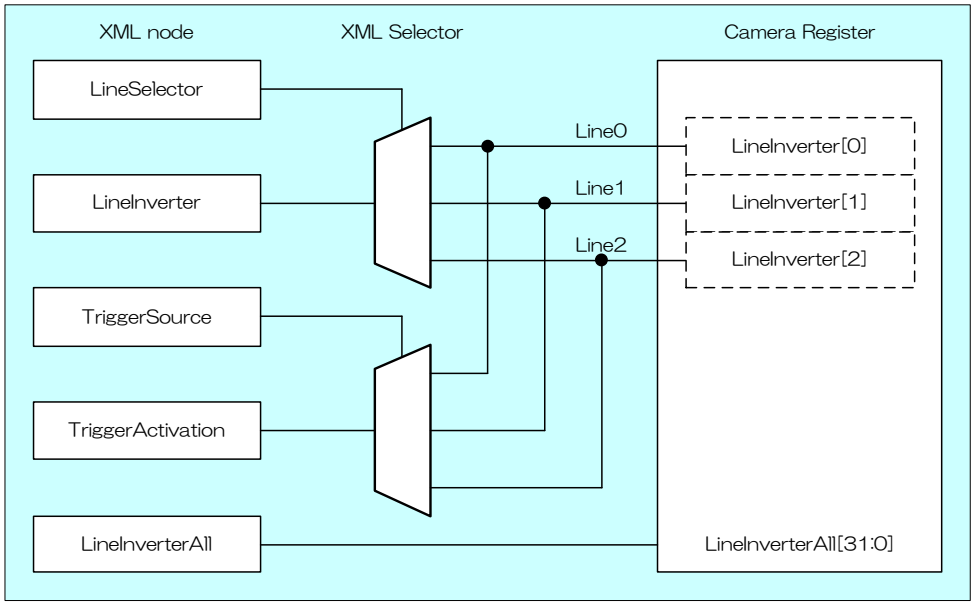
LineModeAll によって各 Line の入出力を選択します。
設定値は Integer 型で各 bit が各 Line に対応しています（bit0=Line0(不変), bit1=Line1(不変 bit2=Line2）。

bit value	I/O Direction
0	入力
1	出力

LineModeAll Value	bit value [Line2] [Line1] [Line0]	I/O Direction [Line2] [Line1] [Line0]
0	使用不可	
1		
2	[0] [1] [0]	[in] [out] [in]
3	使用不可	
4		
5		
6(※)	[1] [1] [0]	[out] [out] [in]
7	使用不可	

※ 出荷設定

◆LineInverterAll



LineInverterAll によって各 Line の極性を選択します。

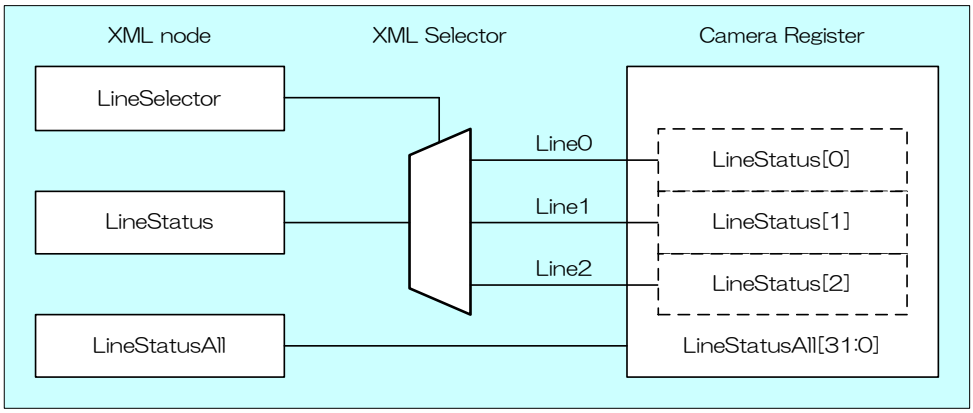
設定値は Integer 型で各 bit が各 Line に対応しています(bit0=Line0, bit1=Line1, bit2=Line2)。

bit value	Inverter
0	False (invert なし)
1	True (invert あり)

LineInverterAll Value	bit value [Line2] [Line1] [Line0]	Inverter [Line2] [Line1] [Line0]
0 (※)	[0] [0] [0]	[off] [off] [off]
1	[0] [0] [1]	[off] [off] [on]
2	[0] [1] [0]	[off] [on] [off]
3	[0] [1] [1]	[off] [on] [on]
4	[1] [0] [0]	[on] [off] [off]
5	[1] [0] [1]	[out [off] [on]
6	[1] [1] [0]	[on] [on] [off]
7	[1] [1] [1]	[on] [on] [on]

※ 出荷設定

◆LineStatusAll

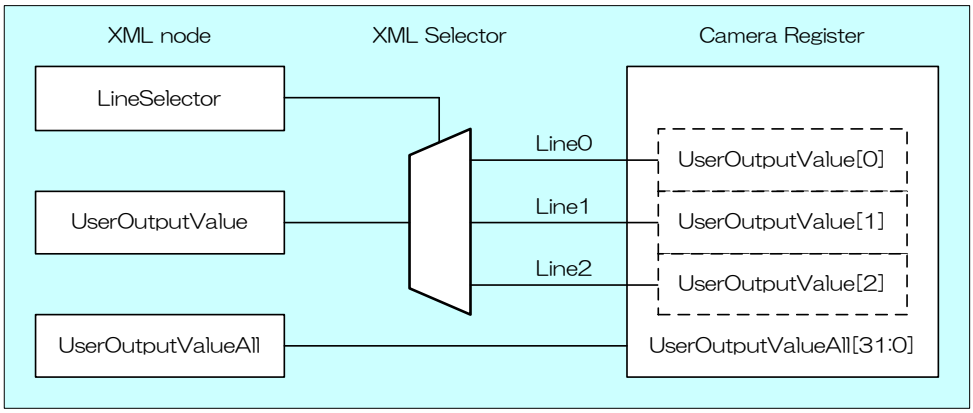


LineStatusAll によって各信号ラインの状態を取得します。
設定値は Integer 型で各 bit が各 Line に対応しています(bit0=Line0, bit1=Line1, bit2=Line2)。

bit value	Line Status
0	False (Low)
1	True (High)

LineStatusAll Value	bit value [Line2] [Line1] [Line0]	Line Status [Line2] [Line1] [Line0]
0	[0] [0] [0]	[low] [low] [low]
1	[0] [0] [1]	[low] [low] [high]
2	[0] [1] [0]	[low] [high] [low]
3	[0] [1] [1]	[low] [high] [high]
4	[1] [0] [0]	[high] [low] [low]
5	[1] [0] [1]	[high] [low] [high]
6	[1] [1] [0]	[high] [high] [low]
7	[1] [1] [1]	[high] [high] [high]

◆UserOutputValueAll



UserOutputValueAll によって Line 出力のユーザー設定値を設定します。

設定値は Integer 型で各 bit が各 Line に対応しています（bit0=Line0(不変), bit1=Line1, bit2=Line2）。

bit value	Output
0	False（Low）
1	True（High）

UserOutputValueAll	bit value [Line2] [Line1]	Output [Line2] [Line1]
0 (※)	[0] [0]	[low] [low]
1		
2	[0] [1]	[low] [high]
3		
4	[1] [0]	[high] [low]
5		
6	[1] [1]	[high] [high]
7		

※ 出荷設定

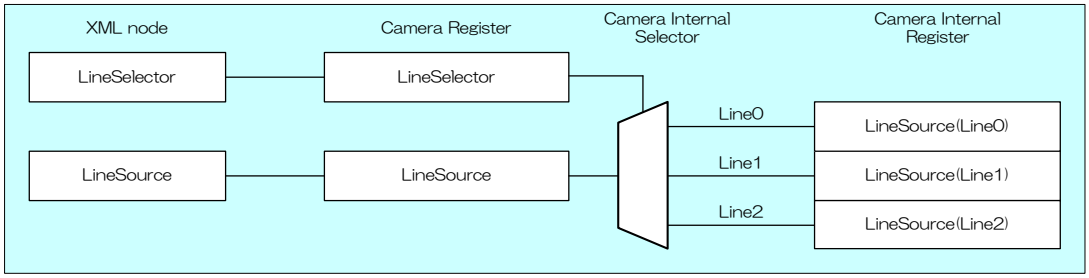
◆LineSelector

LineSelector によって出力する I/O ラインを選択します。

設定値は Enumeration 型で下記のとおりです。

Integer	String
1	Line1
2	Line2

◆LineSource



LineSource によって出力信号の種類を選択します。
設定値は Enumeration 型で下記のとおりです。

Integer	String	説明
0	Off	汎用出力は無効です。
32	UserOutput	UserOutputValue にて設定した値を出力します。
64	Timer0Active	ストロボ制御用信号として使用できます。トリガ入力からの遅延量と幅を設定できます。
99	AcquisitionActive	AcquisitionStart 状態であることを示す信号です。
106	FrameTriggerWait	ランダムトリガシャッタ時に、トリガ待ち受け期間であることを示す信号です。
107	FrameActive	露光開始から CMOS 転送完了までの期間です。
115	FrameTransferActive	映像を USB3.0 バスに転送している期間です。
123	ExposureActive	露光を行っている期間です。

```
// GenICam node handle
CAM_NODE_HANDLE   hMode = NULL;
CAM_NODE_HANDLE   hSelector = NULL;
CAM_NODE_HANDLE   hSource = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, "LineModeAll", &hMode);
Nd_GetNode(s_hCam, "LineSelector", &hSelector);
Nd_GetNode(s_hCam, "LineSource", &hSource);

// Line2/Line1 = output, Line0 = input
int64_t      Mode = 6;
Nd_SetIntValue(s_hCam, hMode, Mode);

// Line1 = ExposureActive
Nd_SetEnumStrValue(s_hCam, hSelector, "Line1");
Nd_SetEnumStrValue(s_hCam, hSource, "ExposureActive");

// Line2 = FrameTransferActive
Nd_SetEnumStrValue(s_hCam, hSelector, "Line2");
Nd_SetEnumStrValue(s_hCam, hSource, "FrameTransferActive");
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IInteger node functions], [IEnumeration node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして DigitalIOControl を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆LineModeAll

LineModeAll レジスタの Value フィールドに書き込みます。

◆LineInverterAll

LineInverterAll レジスタの Value フィールドに書き込みます。

◆LineStatusAll

LineStatusAll レジスタの Value フィールドに書き込みます。

◆UserOutputValueAll

UserOutputAll レジスタの Value フィールドに書き込みます。

◆LineSelector

LineSelector レジスタの Value フィールドに書き込みます。

◆LineSource

LineSource レジスタの Value フィールドに書き込みます。

```
// Line2/Line1 = output, Line0 = input
uint32_t      uiMode = 6;
Cam_WriteReg(s_hCam, 0x209030, 1, &uiMode);

// Set Value
uint32_t      uiSelector;
uint32_t      uiSource;
uiSelector = 1;       // Line1
uiSource = 123;    // ExposureActive
Cam_WriteReg(s_hCam, 0x2090BC, 1, &uiSelector);
Cam_WriteReg(s_hCam, 0x2090DC, 1, &uiSource);

uiSelector = 2;       // Line2
uiSource = 115;    // FrameTransferActive
Cam_WriteReg(s_hCam, 0x2090BC, 1, &uiSelector);
Cam_WriteReg(s_hCam, 0x2090DC, 1, &uiSource);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

● Note

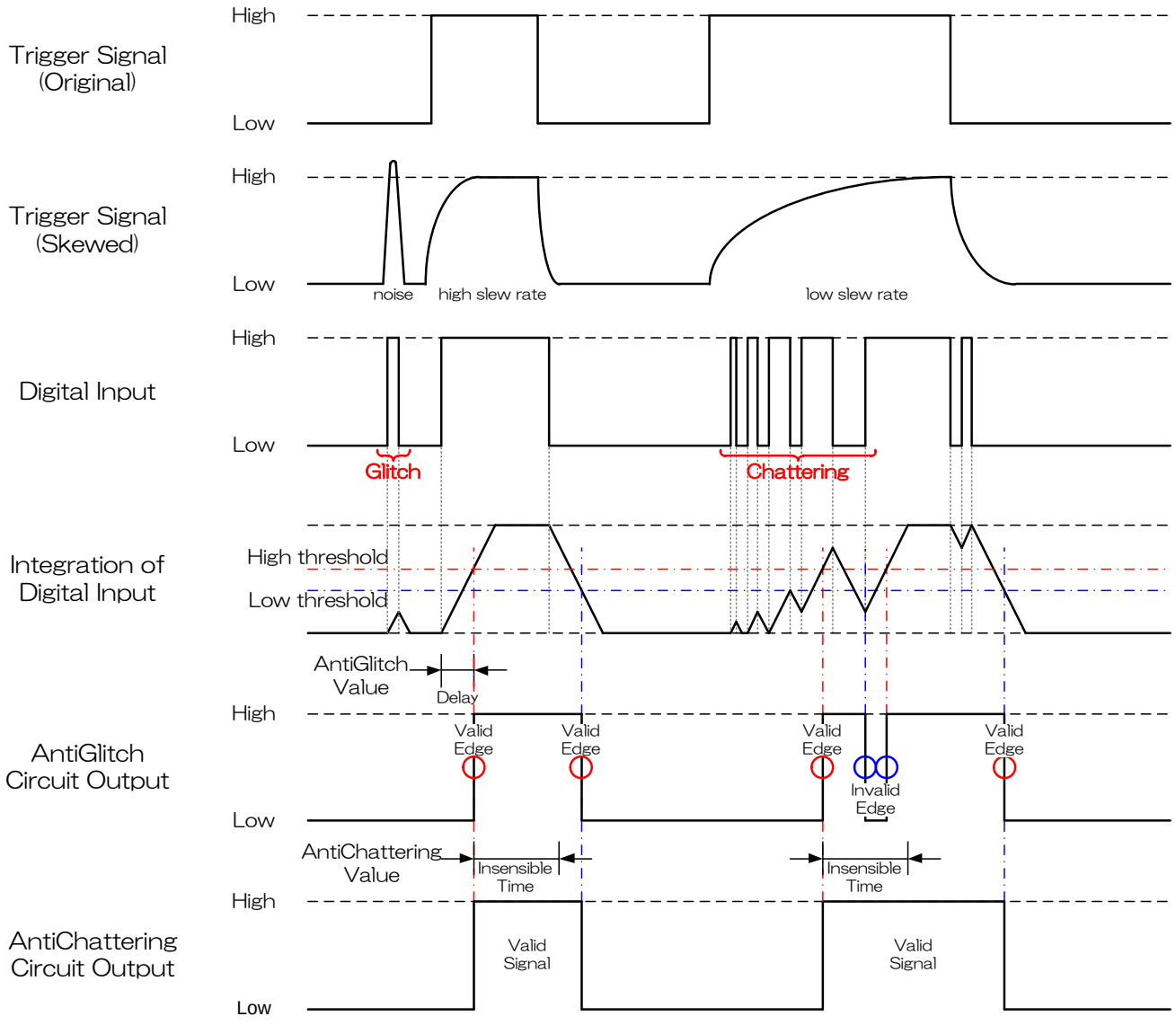
- Line0 は入力専用です。
- Line1 は出力専用です。
- Line2 は入出力変更可能です。出荷設定は入力です。
- LineSource：Timer0Active 信号の動作設定については TimerControl の項目を参照してください。

AntiGlitch/AntiChattering

アンチグリッチとアンチチャタリングはノイズや不安定なデジタル入力（トリガ信号）にフィルタをかける機能です。

アンチグリッチ回路は、トリガ信号のデジタル積分を行います。インパルス性ノイズを取り除くことに有効です。

アンチチャタリング回路は、トリガの誤動作を防止するためにエッジを受け付けない時間を設定します。不安定な論理状態やスイッチチャタリングを取り除くことに有効です。



アンチグリッチとアンチチャタリング

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
AntiGlitch	IFloat	4	R/W	デジタル入力信号の積分時間（絶対値）を設定します。
AntiChattering	IFloat	4	R/W	デジタル入力信号のエッジを受け付けない時間（絶対値）を設定します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
AntiGlitch	Implemented	0x21F3C0	[31]	R	この機能が有効かどうかを返します。
	Mult	0x21F3CC	4	R	絶対値［sec］= Raw * (Mult / Div)
	Div	0x21F3D0	4	R	
	Min	0x21F3D4	4	R	デジタル入力信号積分時間の最小値を設定します。
	Max	0x21F3D8	4	R	デジタル入力信号積分時間の最大値を設定します。
	Value	0x21F3DC	4	R/W	デジタル入力信号の積分時間（絶対値）を設定します。
AntiChattering	Implemented	0x21F3E0	[31]	R	この機能が有効かどうかを返します。
	Mult	0x21F3EC	4	R	絶対値［sec］= Raw * (Mult / Div)
	Div	0x21F3F0	4	R	
	Min	0x21F3F4	4	R	デジタル入力信号のエッジを受け付けない時間の最小値を設定します。
	Max	0x21F3F8	4	R	デジタル入力信号のエッジを受け付けない時間の最大値を設定します。
	Value	0x21F3FC	4	R/W	デジタル入力信号のエッジを受け付けない時間（絶対値）を設定します。

● TeliCamSDK 制御

GenICam function API

専用の API を使用して AntiGlitch/AntiChattering を制御します。

◆AntiGlitch/AntiChattering

1.AntiGlitch によってデジタル入力信号の積分時間（絶対値）を設定します。

2.AntiChattering によってデジタル入力信号のエッジを受け付けない時間（絶対値）を設定します。

```
// GenICam node handle
CAM_NODE_HANDLE   hGlitch = NULL;
CAM_NODE_HANDLE   hChattering = NULL;

// AntiGlitch = 1.0[us], AntiChattering = 10.0[us]
float64_t      dGlitch = 0.000001;
float64_t      dChattering = 0.000010;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “AntiGlitch”, &hGlitch);
Nd_GetNode(s_hCam, “AntiChattering”, &hChattering);

// Set Value
Nd_SetFloatValue(s_hCam, hGlitch, dGlitch);
Nd_SetFloatValue(s_hCam, hChattering, dChattering);
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IFloat node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして AntiGlitch/AntiChattering を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆AntiGlitch/AntiChattering

1.AntiGlitch レジスタの Value フィールドに書き込みます。

2.WAntiChattering レジスタの Value フィールドに書き込みます。

```
// AntiGlitch = 1.0[us] (Raw value = 100)
// AntiChattering = 10.0[us] (Raw value = 1250)
uint32_t      uiAntiGlitchRaw = 100;
uint32_t      uiAntiChatteringRaw = 1250;

// Set Value
Cam_WriteReg(s_hCam, 0x21F3DC, 1, &uiAntiGlitchRaw);
Cam_WriteReg(s_hCam, 0x21F3FC, 1, &uiAntiChatteringRaw);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

◆最小値／最大値

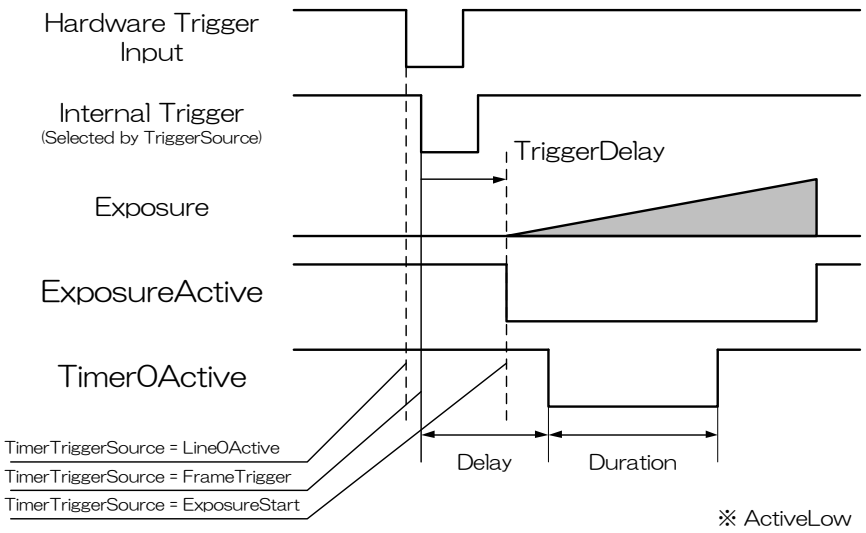
AntiGlitch	Raw 値	絶対値(Float)
最小値	9	0.00000009 [s]
最大値	200000	0.002 [s]
初期値	9	0.00000009 [s]
式	絶対値 [s] = Raw 値 / 100000000	

AntiChattering	Raw 値	絶対値(Float)
最小値	250	0.000002 [s]
最大値	250000	0.002 [s]
初期値	250	0.000002 [s]
式	絶対値 [s] = Raw 値 / 125000000	

TimerControl

Timer0Active 信号は露光タイミングを基準にレジスタ設定にて生成することがきでます。

ExposureActive 信号とタイミングを合わせることで、ストロボなどの照明機器の制御用信号として使用できます。



Timer0Active

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
TimerSelector	IEnumeration	4	R	選択されているタイマー名を返します。
TimerDelay	IFloat	4	R/W	Timer0Active 信号の遅延量を設定します。
TimerDuration	IFloat	4	R/W	Timer0Active 信号の幅を設定します。
TimerTriggerSource	IEnumeration	4	R/W	Timer0Active 信号の基準信号を選択します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
TimerSelector	Implemented	0x20A020	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x20A02C	4	R	[0] : Timer0
	Value	0x20A03C	4	R/W	選択されているタイマー名を返します。
TimerDelay	Implemented	0x20A040	[31]	R	この機能が有効かどうかを返します。
	Mult	0x20A04C	4	R	絶対値［sec］= Raw * (Mult / Div)
	Div	0x20A050	4	R	
	Min	0x20A054	4	R	Timer0Active 信号遅延量の最小値を返します
	Max	0x20A058	4	R	Timer0Active 信号遅延量の最大値を返します
	Value	0x20A05C	4	R/W	Timer0Active 信号の遅延量を設定します。
TimerDuration	Implemented	0x20A060	[31]	R	この機能が有効かどうかを返します。
	Mult	0x20A06C	4	R	絶対値［sec］= Raw * (Mult / Div)
	Div	0x20A070	4	R	
	Min	0x20A074	4	R	Timer0Active 信号幅の最小値を返します。
	Max	0x20A078	4	R	Timer0Active 信号幅の最大値を返します。
	Value	0x20A07C	4	R/W	Timer0Active 信号の幅を設定します。
TimerTriggerSource	Implemented	0x20A080	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x20A08C	4	R/W	[0] : Off [32] : Line0 [104] : FrameTrigger [124] : ExposureStart
	Value	0x20A09C	4	R/W	Timer0Active 信号の基準信号を選択します。

● TeliCamSDK 制御

Camera feature API

専用の API を使用して TimerControl を制御します。

API 名	説明
GetCamTimerDelayMinMax	TimerDelay の最小値と最大値の値を取得します
GetCamTimerDelay	TimerDelay の値を取得します。
SetCamTimerDelay	TimerDelay に値を設定します。
GetCamTimerDurationMinMax	TimerDuration の最小値と最大値の値を取得します
GetCamTimerDuration	TimerDuration の値を取得します。
SetCamTimerDuration	TimerDuration に値を設定します。
GetCamTimerTriggerSource	TimerTriggerSource の値を取得します。
SetCamTimerTriggerSource	TimerTriggerSource に値を設定します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions を参照してください。

GenICam function API

GeniCam API を使用して TimerControl を制御します。

◆TimerDelay/TimerDuration/TimerTriggerSource

1. TimerDelay によって Timer0Active 信号の遅延量を設定します。
2. TimerDuration によって Timer0Active 信号の幅を設定します。
3. TimerTriggerSource によって Timer0Active 信号の基準信号を選択します。
   設定値は Enumeration 型で下記のとおりです。

Integer	String	説明
0	Off	Timer 出力は無効です。
32	Line0	Line0 入力より Timer がスタートします。
104	FrameTrigger	トリガ受付より Timer がスタートします。
124	ExposureStart	露光開始より Timer がスタートします。

```
// GenICam node handle
CAM_NODE_HANDLE   hDelay = NULL;
CAM_NODE_HANDLE   hDuration = NULL;
CAM_NODE_HANDLE   hSource = NULL;

// TimerDelay = 1000.0[us], TimerDuration = 2000.0[us]
float64_t      dDelay = 1000.0;
float64_t      dDuration = 2000.0;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, "TimerDelay", &hDelay);
Nd_GetNode(s_hCam, "TimerDuration", &hDuration);
Nd_GetNode(s_hCam, "TimerTriggerSource", &hSource);

// 1.Sets the delay of Timer0Active signal.
Nd_SetFloatValue(s_hCam, hDelay, dDelay);

// 2.Sets the dulation of Timer0Active signal.
Nd_SetFloatValue(s_hCam, hDuration, dDuration);

// 3.Selects the source of Timer0Active pulse to start.
Nd_SetEnumStrValue(s_hCam, hSource, "ExposureStart");
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IFloat node functions], [IEnumeration node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして TimerControl を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆TimerDelay / TimerDuration / TimerTriggerSource

1.TimerDelay レジスタの Value フィールドに書き込みます。

2.TimerDuration レジスタの Value フィールドに書き込みます。

3.TimerTriggerSource レジスタの Value フィールドに書き込みます。

```
// TimerDelay = 1000.0[us] (Raw value = 125000)
// TimerDuration = 2000.0[us] (Raw value = 250000)
// TimerTriggersource = "ExposureStart"
uint32_t      uiTimerDelayRaw = 125000;
uint32_t      uiTimerDurationRaw = 250000;
uint32_t      uiTimerTriggerSource = 124;

// Set Value
Cam_WriteReg(s_hCam, 0x20A05C, 1, &uiTimerDelayRaw);
Cam_WriteReg(s_hCam, 0x20A07C, 1, &uiTimerDurationRaw);
Cam_WriteReg(s_hCam, 0x21F27C, 1, &uiTimerTriggerSource);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

◆最小値／最大値

TimerDelay TimerDuration	Raw 値	絶対値(Float)
最小値	0	0 [us]
最大値	250000000	2000000 [us]
初期値	0	0.00 [us]
式	絶対値 [us] = Raw 値 / 125	

● 備考

TimerTriggerSource の動作は以下のとおりとなっております。

・Line0Active はハードウェアトリガ入力のみ有効です。FrameTriggerError 時にも反応します。

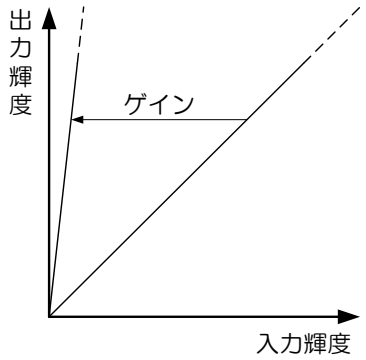
・FrameTrigger はハードウェアトリガ、ソフトウェアトリガ入力ともに有効です。FrameTriggerError 時には反応しません。

・ExposureStart はハードウェアトリガ、ソフトウェアトリガ入力ともに有効です。FrameTriggerError 時には反応しません。

・TriggerDelay が設定されている場合、TriggerDelay+TimerDelay[us]の遅延が発生します。

Gain

ゲインを設定することで、映像輝度の倍率を変更することができます。制御方式としてマニュアルゲイン（MANUAL）と自動ゲイン制御（AGC）が利用可能です。AGC では被写体の明るさに応じてゲインを自動で調整します。



ゲイン設定時の入力輝度と出力輝度の関係は下記のとおりです。

$$出力輝度 = 入力輝度 \times 10^{\frac{Gain}{20}}$$

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
Gain	IFloat	4	R/W	ゲイン（絶対値）を設定します。
GainAuto	IEnumeration	4	R/W	ゲイン動作モードを設定します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
Gain	Implemented	0x204060	[31]	R	この機能が有効かどうかを返します。
	Control	0x204068	4	R/W	[1]: Manual [2]: Auto
	Mult	0x20406C	4	R	絶対値［dB］= Raw * (Mult / Div)
	Div	0x204070	4	R	
	Min	0x204074	4	R	ゲインの最小値を返します。
	Max	0x204078	4	R	ゲインの最大値を返します。
	Value	0x20407C	4	R/W	ゲイン（絶対値）を設定します。

● TeliCamSDK 制御

Camera feature API

専用の API を使用して Gain を制御します。

API 名	説明
GetCamGainMinMax	Gain の最小値と最大値を取得します。
GetCamGain	Gain の値を取得します。
SetCamGain	Gain に値を設定します。
GetCamGainAuto	GainAuto モードの値を取得します。
SetCamGainAuto	GainAuto モードに値を設定します。

GainAuto パラメータ	説明
CAM_GAIN_AUTO_OFF	Off
CAM_GAIN_AUTO_AUTO	Adjust continuously

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用して Gain を制御します。.

◆Gain

Gain を制御するには IFloat インターフェースを使用します。

```
// GenICam node handle
CAM_NODE_HANDLE   hNode = NULL;

// Gain = 6.0dB
float64_t      dGain = 6.0;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “Gain”, &hNode);

// Set Gain Value
Nd_SetFloatValue(s_hCam, hNode, dGain);
```

◆GainAuto

GainAuto を制御するには IEnumertion インターフェースを使用します。

設定値は Enumeration 型で下記のとおりです。

Integer	String	説明
1	Off	マニュアルゲイン制御（MANUAL）
2	Continuous	自動ゲイン制御（AGC）

```
// GenICam node handle
CAM_NODE_HANDLE   hNode = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “GainAuto”, &hNode);

// Set GainAuto = “Continuous”
Nd_SetEnumStrValue(s_hCam, hNode, “Continuous”);
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IFloat node functions],

[IEnumeration node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして Gain を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆Gain

Gain レジスタの Value フィールドに書き込みます。

```
// Gain = 6.0dB (Raw value = 60)
uint32_t      uiGainRaw = 60;

// Set Gain Value
Cam_WriteReg(s_hCam, 0x20407C, 1, &uiGainRaw);
```

◆GainAuto

Gain レジスタの Control フィールドに書き込みます。

```
// GainAuto = “Continuous”
uint32_t      uiGainAuto = 2;

// Set GainAuto = “Continuous”
Cam_WriteReg(s_hCam, 0x204068, 1, &uiGainAuto);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

◆最小値／最大値

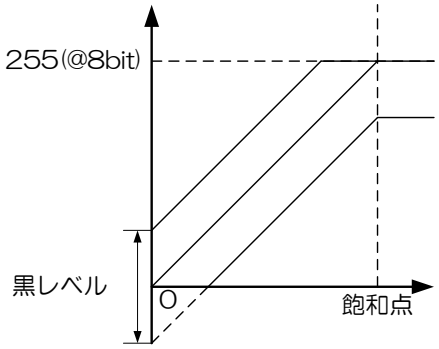
Gain	Raw 値	絶対値(Float)
最小値	0	0.00 [dB]
最大値	240	24.00 [dB]
初期値	0	0.00 [dB]
式	絶対値 [dB] = Raw 値 / 10	

● 備考

・GainAuto モードでは、現在のゲインの値が Gain レジスタに設定されます。

BlackLevel

映像の黒レベルを設定します。映像の飽和レベルを 100%として、黒レベル(画像レベルの取りうる最小値)を-25.0%～+25.0%の範囲で設定可能です。但し黒レベルを 0%以下にすると、映像輝度が飽和しない場合があります。



黒レベル

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
BlackLevel	IFloat	4	R/W	黒レベル(絶対値)を設定します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
BlackLevel	Implemented	0x204040	[31]	R	この機能が有効かどうかを返します。
	Mult	0x20404C	4	R	絶対値 [%] = Raw * (Mult / Div)
	Div	0x204050	4	R	
	Min	0x204054	4	R	黒レベルの最小値を返します。
	Max	0x204058	4	R	黒レベルの最大値を返します。
	Value	0x20405C	4	R/W	黒レベル(絶対値)を設定します。

● TeliCamSDK 制御

Camera feature API

専用の API を使用して BlackLevel を制御します。

API 名	説明
GetCamBlackLevelMinMax	BlackLevel の最小値と最大値を取得します。
GetCamBlackLevel	BlackLevel の値を取得します。
SetCamBlackLevel	BlackLevel に値を設定します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用して BlackLevel を制御します。

◆BlackLevel

BlackLevel を制御するには IFloat インターフェースを使用します。

```
// GenICam node handle
CAM_NODE_HANDLE   hNode = NULL;

// BlackLevel = 25%
float64_t      dBlackLevel = 25.0;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “BlackLevel”, &hNode);

// Set BlackLevel Value
Nd_SetFloatValue(s_hCam, hNode, dBlackLevel);
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IFloat node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして BlackLevel を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆BlackLevel

BlackLevel レジスタの Value フィールドに書き込みます。

```
// BlackLevel = 25% (Raw value = 256)
int32_t      iBlackLevelRaw = 256; // signed

// Set BlackLevel Value
Cam_WriteReg(s_hCam, 0x20405C, 1, &iBlackLevelRaw);
```

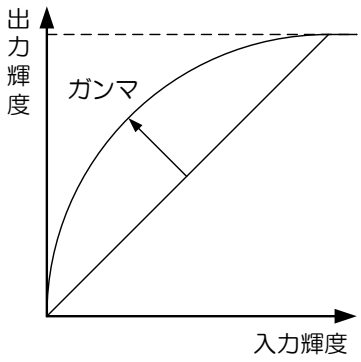
詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

◆最小値／最大値

BlackLevel	Raw 値	絶対値(Float)
最小値	-256	-25.00[%]
最大値	256	+25.00[%]
初期値	0	0.00[%]
式	絶対値［%］= Raw 値 ＊100 / 1024	

Gamma

出力映像に対しガンマ補正を適用します。



ガンマ

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
Gamma	IFloat	4	R/W	ガンマ補正値の設定をします。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
Gamma	Implemented	0x204080	[31]	R	この機能が有効かどうかを返します。
	Mult	0x20408C	4	R	絶対値 = Raw * (Mult / Div)
	Div	0x204090	4	R	
	Min	0x204094	4	R	ガンマ補正値の最小値を返します。
	Max	0x204098	4	R	ガンマ補正値の最大値を返します。
	Value	0x20409C	4	R/W	ガンマ補正値の設定をします。

● Control with TeliCamSDK

Camera feature API

専用の API を使用して Gamma を制御します。

API 名	説明
GetCamGammaMinMax	Gamma の最小値と最大値を取得します。
GetCamGamma	Gamma の値を取得します。
SetCamGamma	Gamma に値を設定します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用して Gamma を制御します。.

◆Gamma

Gamma を制御するには IFloat インターフェースを使用します。

```
// GenICam node handle
CAM_NODE_HANDLE   hNode = NULL;

// Gamma = 0.45
float64_t      dGamma = 0.45;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “Gamma”, &hNode);

// Set Gamma Value
Nd_SetFloatValue(s_hCam, hNode, dGamma);
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IFloat node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして Gamma を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆Gamma

Gamma レジスタの Value フィールドに書き込みます。

```
// Gamma = 0.45 (Raw value = 45)
uint32_t      uiGammaRaw = 45;

// Set Gamma Value
Cam_WriteReg(s_hCam, 0x20409C, 1, &uiGammaRaw);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。.

◆最小値／最大値

Gamma	Raw 値	絶対値(Float)
最小値	45	0.45
最大値	100	1.00
初期値	100	1.00
式	絶対値 = Raw 値 / 100	

Hue / Saturation

Hue 設定により色相を調整することができます。また、Saturation 設定により彩度を調整することができます。

本機能はカラーモデルのみで使用可能です。

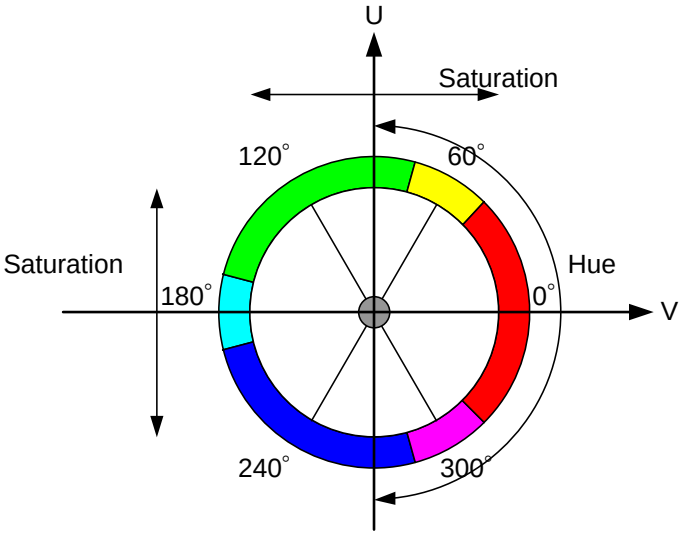


図. Hue/Saturation

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
Hue	IFloat	4	R/W	色相を設定します
Saturation	IFloat	4	R/W	彩度を設定します

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
Hue	Implemented	0x205020	[31]	R	この機能が有効かどうかを返します。
	Mult	0x20502C	4	R	絶対値 [deg] = Raw * (Mult / Div)
	Div	0x205030	4	R	
	Min	0x205034	4	R	Hue の最小値を返します。
	Max	0x205038	4	R	Hue の最大値を返します。
	Value	0x20503C	4	R/W	Hue の値を設定します。
Saturation	Implemented	0x205040	[31]	R	この機能が有効かどうかを返します。
	Mult	0x20504C	4	R	絶対値 [deg] = Raw * (Mult / Div)
	Div	0x205050	4	R	
	Min	0x205054	4	R	Saturation の最小値を返します。
	Max	0x205058	4	R	Saturation の最大値を返します。
	Value	0x20505C	4	R/W	Saturation の値を設定します

● TeliCamSDK 制御

Camera feature API

専用の API を使用して ExposureTime を制御します。

API 名	説明
GetCamHueMinMax	Hue の最小値と最大値を取得する。
GetCamHue	Hue の値を取得する。
SetCamHue	Hue の値に設定する。
GetCamSaturationMinMax	Saturation の最小値と最大値を取得する。
GetCamSaturation	Saturation の値を取得する。
SetCamSaturation	Saturation に値を設定する。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください

GenICam function API

GeniCam API を使用して Hue と Saturation を制御します。

◆Hue

Hue を制御するには Float インターフェースを使用します。

```
// GenICam node handle
CAM_NODE_HANDLE   hNode = NULL;

// Hue = 0.0 [deg]
float64_t      dHue = 0.0;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “Hue”, &hNode);

// Set Gain Value
Nd_SetFloatValue(s_hCam, hNode, dHue);
```

◆Saturation

Saturation を制御するには Float インターフェースを使用します。

```
// GenICam node handle
CAM_NODE_HANDLE   hNode = NULL;

// Saturation = 100.0 [%]
float64_t      dSaturation = 100.0;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “Saturation”, &hNode);

// Set Saturation Value
Nd_SetFloatValue(s_hCam, hNode, dSaturation);
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IFloat node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして ExposureTime を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆Hue

Hue レジスタの Value フィールドに書き込みます。

```
// Hue = 0.0 [deg] (Raw value = 0)
uint32_t      uiHueRaw = 0;

// Set Hue Value
Cam_WriteReg(s_hCam, 0x20503C, 1, &uiHueRaw);
```

◆Saturation

Saturation レジスタの Value フィールドに書き込みます。

```
// Saturation = 100.0 [%] (Raw value = 65536)
uint32_t      uiSaturationRaw = 65536;

// Set Saturation Value
Cam_WriteReg(s_hCam, 0x20505C, 1, &uiSaturationRaw);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

◆最小値／最大値

Hue	Raw 値	絶対値(Float)
最小値	-180	-180[deg]
最大値	180	+180[deg]
初期値	0	0[deg]
式	絶対値 [deg] = Raw 値	

Saturation	Raw 値	絶対値(Float)
最小値	0	0[%]
最大値	131072	200[%]
初期値	65536	100[%]
式	絶対値 [%] = Raw 値 * 100 / 65536	

● 備考

Hue/Saturation は以下の PixelFormat 時に有効です。

Bayer8/10/12(BayerProcessingMode = Full), RGB8, BGR8, YUV411, YUV422

BalanceRatio

BalanceRatio の設定によりホワイトバランスゲインを調整します。
本機能はカラーモデルのみで使用可能です。

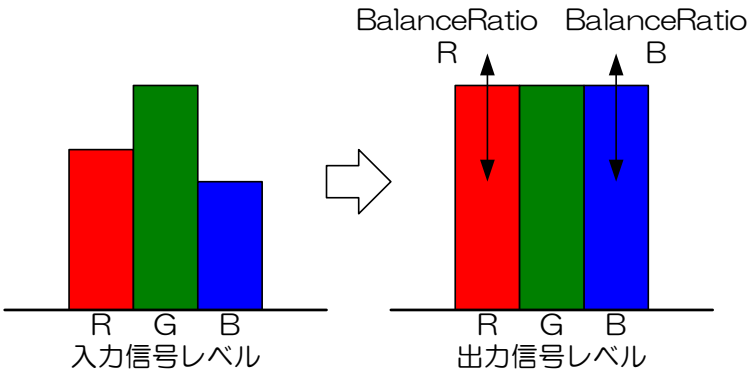


図. BalanceRatio

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
BalanceRatioSelector	IEnumeration	4	R/W	ホワイトバランスゲイン設定の対象となる要素を選択します。
BalanceRatio	IFloat	4	R/W	ホワイトバランスゲイン(倍率)を設定します。
BalanceWhiteAuto	IEnumeration	4	R/W	ホワイトバランスゲイン(倍率)を自動で設定します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
WhiteBalanceR	Implemented	0x205060	[31]	R	この機能が有効かどうかを返します。
	Control	0x205068	4	R/W	[1]: Off [2]: Continuous [3]: Once
	Mult	0x20506C	4	R	絶対値 [times] = Raw * (Mult / Div)
	Div	0x205070	4	R	
	Min	0x205074	4	R	WhiteBalanceR の最小値を返します。
	Max	0x205078	4	R	WhiteBalanceR の最大値を返します。
	Value	0x20507C	4	R/W	WhiteBalanceR を設定します。
WhiteBalanceB	Implemented	0x205080	[31]	R	この機能が有効かどうかを返します。
	Control	0x205088	4	R/W	[1]: Off [2]: Continuous [3]: Once
	Mult	0x20508C	4	R	絶対値 [times] = Raw * (Mult / Div)
	Div	0x205090	4	R	
	Min	0x205094	4	R	WhiteBalanceB の最小値を返します。
	Max	0x205098	4	R	WhiteBalanceB の最大値を返します。
	Value	0x20509C	4	R/W	WhiteBalanceB を設定します。

● TeliCamSDK 制御

Camera feature API

専用の API を使用して Control BalanceRatio and BalanceWhiteAuto を制御します。

API 名	説明
GetCamBalanceRatioMinMax	BalanceRatio の最小値と最大値を取得する。
GetCamBalanceRatio	BalanceRatio を取得する。
SetCamBalanceRatio	BalanceRatio を設定する。
GetCamBalanceWhiteAuto	BalanceWhiteAuto の設定を取得する。
SetCamBalanceWhiteAuto	BalanceWhiteAuto を設定する。

BalanceWhiteAuto パラメータ	説明
CAM_BALANCE_WHITE_AUTO_OFF	Off
CAM_BALANCE_WHITE_AUTO_ONCE	Adjust once

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用して Control BalanceRatio and BalanceWhiteAuto を制御します。

◆BalanceRatio

1. BalanceRatioSelector によって設定する色成分を選択します。

   設定値は Enumeration 型と String 型で下記のとおりです。

Integer	String
1	Red
2	Blue

2. BalanceRatio によってホワイトバランスゲインを設定します。

```
// GenICam node handle
CAM_NODE_HANDLE   hNode = NULL;

// 1.Select a color component
// Retrieve GenICam node.
Nd_GetNode(s_hCam, “BalanceRatioSelector”, &hNode);

// Select “Red”
Nd_SetEnumStrValue(s_hCam, hNode, “Red”);

// 2.Set a white balance gain
// white balance R gain = x 1.0
float64_t      dBalanceRatioR = 1.0;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “BalanceRatio”, &hNode);

// Set Gain Value
Nd_SetFloatValue(s_hCam, hNode, dBalanceRatioR);
```

◆BalanceWhiteAuto

Sharpness を制御するには Enumeration インターフェースを使用します。

設定値は Enumeration 型と String 型で下記のとおりです。

Integer	String
1	Off
3	Once

```
// GenICam node handle
CAM_NODE_HANDLE   hNode = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “BalanceWhiteAuto”, &hNode);

// Set BalanceWhiteAuto = “Continuous”
Nd_SetEnumStrValue(s_hCam, hNode, “Continuous”);
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IFloat node functions], [IEnumeration node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして Control BalanceRatio and BalanceWhiteAuto を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆BalanceRatio

WhiteBalanceR または WhiteBalanceB レジスタの Value フィールドに書き込みます。.

```
// WhiteBalance R Gain = x 1 (Raw value = 65536)
// WhiteBalance B Gain = x 1 (Raw value = 65536)
uint32_t      uiWhiteBalanceRRaw = 65536;
uint32_t      uiWhiteBalanceBRaw = 65536;

// Set WhiteBalance Gain Value
Cam_WriteReg(s_hCam, 0x20507C, 1, &uiWhiteBalanceRRaw);
Cam_WriteReg(s_hCam, 0x20509C, 1, &uiWhiteBalanceBRaw);
```

◆BalanceRatioAuto

WhiteBalanceR または WhiteBalanceB レジスタの Control フィールドに書き込みます。.

```
// BalanceWhiteAuto = “Once”
uint32_t      uiBalanceWhiteAuto = 3;

// Set BalanceWhiteAuto = “Once”
Cam_WriteReg(s_hCam, 0x205068, 1, &uiBalanceWhiteAuto);
// Cam_WriteReg(s_hCam, 0x205088, 1, &uiBalanceWhiteAuto); // either will do
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。.

◆最小値／最大値

BalanceRatio	Raw 値	絶対値(Float)
最小値	65536	x 1 [times]
最大値	524287	x 8 [times]
初期値	機器による	機器による
式	絶対値 [times] = Raw 値 / 65536	

● 備考

BalanceWhiteAuto で選択された要素のホワイトバランスゲインは BalanceRatioR、BalanceRatioB レジスタに設定します。

BalanceRatio and BalanceRatioAuto controls は以下の PixelFormat 時に有効です。

Bayer8/10/12(BayerProcessingMode = Full、Partial)、RGB8、BGR8、YUV411、YUV422

お願い：ホワイトバランスゲイン可変時の画質について

ホワイトバランスゲイン設定値を上げすぎるとノイズが増加する場合があります。撮影画像の明るさを調整する場合は、機械・装置全体で最終的な画質の確認をお客様にて実施して頂くようお願い致します。

ColorCorrectionMatrix

色補正マトリクスを利用して RGB レベルを補正することができます。
本機能はカラーモデルのみで使用可能です。

補正前データ R、G、B と補正後データ R’、G’、B’の関係は下記の式で表されます。

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} 1 & -mask\_rg & -mask\_rb \\ -mask\_gr & 1 & -mask\_gb \\ -mask\_br & -mask\_bg & 1 \end{bmatrix} \begin{bmatrix} R & (G-R) & (B-R) \\ (R-G) & G & (B-G) \\ (R-B) & (G-B) & B \end{bmatrix}$$

$$R' = (1 - mask\_rg - mask\_rb) \cdot R + mask\_rg \cdot G + mask\_rb \cdot B$$
$$G' = mask\_gr \cdot R + (1 - mask\_gr - mask\_gb) \cdot G + mask\_gb \cdot B$$
$$B' = mask\_br \cdot R + mask\_bg \cdot G + (1 - mask\_br - mask\_bg) \cdot B$$

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
ColorCorrectionMatrixSelectorI	IEnumeration	4	R/W	色補正マトリクスの行要素を選択します。
ColorCorrectionMatrixSelectorJ	IEnumeration	4	R/W	色補正マトリクスの列要素を選択します。
ColorCorrectionMatrix	IFloat	4	R/W	色補正マトリクスの係数を設定します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
MaskingRG	Implemented	0x21F040	[31]	R	この機能が有効かどうかを返します。
	Mult	0x21F04C	4	R	絶対値 = Raw * (Mult / Div)
	Div	0x21F050	4	R	
	Min	0x21F054	4	R	マスキング(R-G) の最小値を返します。
	Max	0x21F058	4	R	マスキング(R-G) の最大値を返します。
	Value	0x21F05C	4	R/W	マスキング(R-G) を設定します。
MaskingRB	MaskingRG と同じ構造です。				
	Value	0x21F07C	4	R/W	マスキング(R-B) を設定します。
MaskingGR	MaskingRG と同じ構造です。				
	Value	0x21F09C	4	R/W	マスキング(G-R) を設定します。
MaskingGB	MaskingRG と同じ構造です。				
	Value	0x21F0BC	4	R/W	マスキング(G-B) を設定します。
MaskingBR	MaskingRG と同じ構造です。				
	Value	0x21F0DC	4	R/W	マスキング(B-R) を設定します。
MaskingBG	MaskingRG と同じ構造です。				
	Value	0x21F0FC	4	R/W	マスキング(B-G) を設定します。

● TeliCamSDK 制御

Camera feature API

専用の API を使用して ColorCorrectionMatrix を制御します。

API 名	説明
GetCamColorCorrectionMatrixMinMax	ColorCorrectionMatrix の最小値と最大値の値を取得します。
GetCamColorCorrectionMatrix	ColorCorrectionMatrix の値を取得します。
SetCamColorCorrectionMatrix	ColorCorrectionMatrix に値を設定します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions を参照してください。

GenICam function API

GeniCam API を使用して TimerCo ColorCorrectionMatrix ntrol を制御します。

◆ColorCorrectionMatrix

1. ColorCorrectionMatrixSelectorI によって色補正マトリクスの行要素を選択します。
   ColorCorrectionMatrixSelectorJ によって色補正マトリクスの列要素を選択します。
   設定値は Integer 型と Enumeration 型で下記のとおりです。

Integer	String
0	R
1	G
2	B

ColorCorrectionMatrixSelectorI と ColorCorrectionMatrixSelectorJ の関係は下記の通りです。

	SelectorJ=R	SelectorJ=G	SelectorJ=B
SelectorI=R		mask_rg	mask_rb
SelectorI=G	mask_gr		mask_gb
SelectorI=B	mask_br	mask_bg	

color correction matrix 対応表

2. ColorCorrectionMatrix によって color correction matrix の係数を設定します。

```
// GenICam node handle
CAM_NODE_HANDLE   hSelectorI = NULL;
CAM_NODE_HANDLE   hSelectorJ = NULL;
CAM_NODE_HANDLE   hNode = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “ColorCorrectionMatrixSelectorI”, &hSelectorI);
Nd_GetNode(s_hCam, “ColorCorrectionMatrixSelectorJ”, &hSelectorJ);
Nd_GetNode(s_hCam, “ColorCorrectionMatrix”, &hNode);

float64_t      dCoefficient[] = { 0.0,-0.2, 0.2,
                                  -0.3,0.0,-0.4,
                                  -0.1,-0.5,0.0 };       // 3x3 matrix

for(int64_t i=0; i<3; i++)
{
 for(int64_t j=0; j<3; j++)
 {
  if(i != j)
  {
   // 1.Select a color correction matrix element
   Nd_SetEnumIntValue(s_hCam, hSelectorI, i);
   Nd_SetEnumIntValue(s_hCam, hSelectorJ, j);

   // 2.Set a coefficient of color correction matrix.
   Nd_SetFloatValue(s_hCam, hNode, dCoefficient);
  }
 }
}
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IFloat node functions], [IEnumeration node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして ColorCorrectionMatrix を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆ColorCorrectionMatrix

MaskingRG、MaskingRB、MaskingGR、MaskingGB、MaskingBR、MaskingBG の Value フィールドに書き込みます。

```
// Masking value (Raw value)
int32_t      uiMasking[] = {-13108, 13107,-19661,-26215,
-6554,-32768};

// Set Masking Value
uint64_t      addr = 0x21F05C;
for(int i=0; i<6; i++, addr+= 0x20)
{
  Cam_WriteReg(s_hCam, addr, 1, &uiMasking[i]);
}
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

◆最小値／最大値

Masking	Raw 値	絶対値(Float)
最小値	-65536	-1.00
最大値	65535	0.99
式	絶対値 = Raw 値 / 65536	

初期値

		SelectorJ		
		R	G	B
SelectorI	R		0	0
	G	0		0
	B	0	0	

● Note

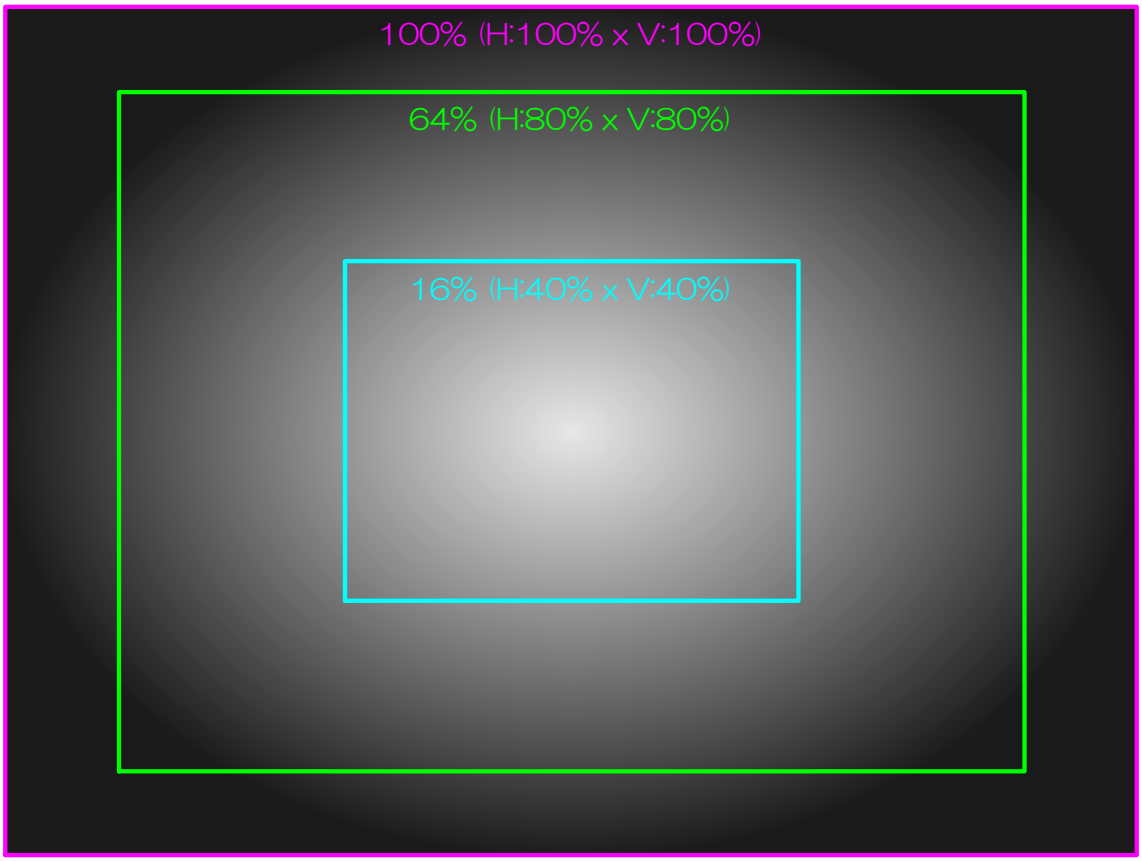
ColorCorrectionMatrix control は以下の PixelFormat 時に有効です。

Bayer8/10/12(BayerProcessingMode = Full), RGB8, BGR8, YUV411, YUV422

ALCControl

ALC 動作は測光範囲の設定と収束値に対して補正値を設定することができます。

・ALCPhotometricAreaSize は輝度を測定するための測光エリアサイズを定義します。



測光エリアサイズのイメージ（それぞれ 100%、64%、16%で設定した場合）

・ALCExposureValue は収束値の補正値を定義します。

ALC 動作収束補正値設定による最終的な収束値は下記の式により求められます。

$$\text{最終収束値} = 84\text{（基準輝度）} \times 2^{\text{ALCExposureValue}}$$

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
ALCPhotometricAreaSize	IFloat	4	R/W	映像輝度を測定するエリアサイズを選択します。
ALCExposureValue	IFloat	4	R/W	ALC 動作の映像輝度収束補正値を設定します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
ALCPhotometricAreaSize	Implemented	0x21F360	[31]	R	この機能が有効かどうかを返します。
	Mult	0x21F36C	4	R	絶対値 = Raw * (Mult / Div)
	Div	0x21F370	4	R	
	Min	0x21F374	4	R	映像輝度を測定するエリアサイズの最小値を返します。
	Max	0x21F378	4	R	映像輝度を測定するエリアサイズの最大値を返します。
	Value	0x21F37C	4	R/W	映像輝度を測定するエリアサイズを選択します。
ALCExposureValue	Implemented	0x2040C0	[31]	R	この機能が有効かどうかを返します。
	Mult	0x2040CC	4	R	絶対値 = Raw * (Mult / Div)
	Div	0x2040D0	4	R	
	Min	0x2040D4	4	R	ALC 動作の映像輝度収束補正値の最小値を返します。
	Max	0x2040D8	4	R	ALC 動作の映像輝度収束補正値の最大値を返します。
	Value	0x2040DC	4	R/W	ALC 動作の映像輝度収束補正値を設定します。

● TeliCamSDK 制御

GenICam function API

GeniCam API を使用して ALC パラメータを制御します。

◆ALCPhotometricAreaSize/ALCExposureValue

ALCPhotometricAreaSize と ALCExposureValue を制御するには IFloat インターフェースを使用します。

```
// GenICam node handle
CAM_NODE_HANDLE   hSize = NULL;
CAM_NODE_HANDLE   hEV = NULL;

// ALCPhotometricAreaSize = 25%
float64_t      dSize = 25.0;
// ALCExposureValue = +1.0EV
float64_t      dEV = 1.0;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “ALCPhotometricAreaSize”, &hSize);
Nd_GetNode(s_hCam, “ALCExposureValue”, &hEV);

// Set ALCPhotometricAreaSize
Nd_SetFloatValue(s_hCam, hSize, dSize);
// Set ALCExposureValue
Nd_SetFloatValue(s_hCam, hEV, dEV);
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IFloat node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして ALC パラメータを制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆ALCPhotometricAreaSize/ALCExposureValue

ALCPhotometricAreaSize レジスタの Value フィールドに書き込みます。

ALCExposureValue レジスタの Value フィールドに書き込みます。

```
// ALCPhotometricAreaSize = 25% (Raw value = 25)
uint32_t      uiSizeRaw = 25;
// ALCExposureValue = +1.0EV (Raw value = 10)
int32_t      iEVRaw = 10; // signed

// Set ALCPhotometricAreaSize
Cam_WriteReg(s_hCam, 0x21F37C, 1, &uiSizeRaw);
// Set ALCExposureValue
Cam_WriteReg(s_hCam, 0x2040DC, 1, &iEVRaw);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

◆最小値／最大値

ALCPhotometricAreaSize	Raw 値	絶対値(Float)
最小値	1	1.00[%]
最大値	100	100.00[%]
初期値	100	100.00[%]
式	最大値［%］= Raw 値	

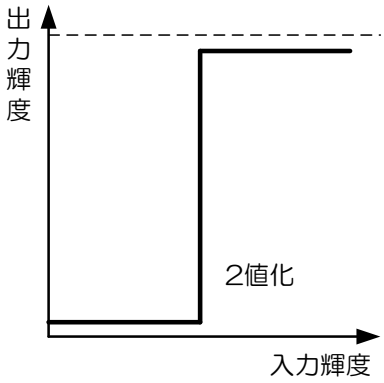
ALCExposureValue	Raw 値	絶対値(Float)
最小値	-20	-2.0[EV]
最大値	15	+1.5[EV]
初期値	100	0.0[EV]
式	最大値［EV］= Raw 値 ／ 10	

● 備考

・ランダムトリガシャッタモード時の ALC 動作は保証いたしません。

LUTControl

映像に対して入力：12bit，出力：12bit の任意の LUT を適用することが可能です。



LUT の設定例

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
LUTEnable	IBoolean	4	R/W	LUT の有効 / 無効を切り替えます。
LUTIndex	IInteger	4	R/W	LUT の入力値を設定します。
LUTValue	IInteger	4	R/W	LUT の出力値を設定します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
LUTEnable	Implemented	0x206020	[31]	R	この機能が有効かどうかを返します。
	Value	0x20603C	4	R/W	LUT の有効 / 無効を切り替えます。 [0] : Off [1] : On
LUTValueAll	Implemented	0x2FFFDC	[31]	R	この機能が有効かどうかを返します。
	Value[0]	0x300000	4	R/W	LUT の出力値を設定します。
	Value[1]	0x300004	4	R/W	LUT の出力値を設定します。
	Value[2]	0x300008	4	R/W	LUT の出力値を設定します。
	…	…	…	…	…
	Value[4095]	0x303FFC	4	R/W	LUT の出力値を設定します。

● TeliCamSDK 制御

Camera feature API

専用の API を使用して LUT を制御します。

API 名	説明
GetCamLUTEnable	LUT の有効/無効を取得します。
SetCamLUTEnable	LUT の有効/無効を設定します。
GetCamLUTValue	LUT の値を取得します。
SetCamLUTValue	LUT の値を設定します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用して LUT を制御します。

◆LUT

1.LUTIndex によって LUT の入力値を設定します。

2.LUTValue によって LUT の出力値を設定します。

3.LUTEnable によって LUT を有効にします。

```
// GenICam node handle
CAM_NODE_HANDLE   hIndex = NULL;
CAM_NODE_HANDLE   hValue = NULL;
CAM_NODE_HANDLE   hEnable = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “LUTIndex”, &hIndex);
Nd_GetNode(s_hCam, “LUTValue”, &hValue);
Nd_GetNode(s_hCam, “LUTEnable”, &hEnable);

for(int64_t i=0; i<4096; i++)
{
    // 1.Set the input level of LUT to 'LUTIndex'.
    Nd_SetIntValue(s_hCam, hIndex, i);
    // 2.Set the output level of LUT to 'LUTValue'
    Nd_SetIntValue(s_hCam, hValue, 4095 - i); // invert
}

// 3.Set the activation of LUT function by 'LUTEnable'
Nd_SetBoolValue(s_hCam, hEnable, true);
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IInteger node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして LUT を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆LUT

1.LUT の出力レベルを設定します。

LUTValueAll レジスタの Value[0]～Value[4095]フィールドに書き込みます。

Value レジスタのインデックスは LUT の入力レベルを意味します。

2.LUT を有効にします。

LUTEnable レジスタの Value フィールドに書き込みます。

```
// 1.Set the output level of LUT.
uint64_t addr = 0x300000;
uint32_t dat;
for(int i=0; i<4096; i++, addr+= 0x4)
{
        dat = 4095 - i; // invert
        Cam_WriteReg(s_hCam, addr, 1, &dat);
}

// 2.Set the activation of LUT function.
dat = 1;
Cam_WriteReg(s_hCam, 0x20603C, 1, &dat);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

◆最小値／最大値

LUTIndex/LUTValue	値
最小値	0
最大値	4095

UserSetControl

カメラに実装されている不揮発性メモリまたは揮発性メモリに、ユーザー設定を Save することができます。不揮発性メモリと揮発性メモリには、ユーザーメモリとして 7 のチャンネルが用意されています。よく使用する設定を Save しておき、使用時に Load することで各々の設定をする手間を省くことができます。Load と Save が適用されるユーザー設定は下記のとおりです。

UserSet 適用レジスタ

カテゴリ	レジスタ名	カテゴリ	レジスタ名
ImageFormatControl	ImageFormatSelector	DigitalIOControl	AntiGlitch
	Width		AntiChattering
	Height	TimerControl	TimieTriggerSource
	OffsetX		TimerDuration
	OffsetY		TimerDelay
	Binning	AnalogControl	Gain
	Reverse		GainControl
	PixelFormat		BlackLevel
	TestPattern		Gamma
TriggerControl	TriggerMode		Hue
	TriggerSequence		Saturation
	TriggerSource		BalanceRatio
	TriggerAdditionalParameter		ColorCorrectionMatrix
	TriggerDelay	ALCControl	ALCExposureValue
ExposureControl	ExposureTime		ALCPhotometricAreaSize
	ExposureControl	LUTControl	LUTEnable
DigitalIOControl	LineModeAll	DPCControl	DPCEnable(※)
	LineInverterAll		DPCNumber(※)
	UserOutputValueAll		DPCEntryX(※)
	LineSelector		DPCEntryY(※)
	LineSource	EventControl	EventNotification
		VendorUniqueControl	LEDIndicatiorLuminance

※ 保存される Entry は 1 チャンネル分で、Entry は全てのチャンネルで共有されます。

この表は白黒/カラーすべての機能を記載しています。P.31「機能一覧」も参照してください

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
UserSetSelector	IEnumeration	4	R/W	ユーザー設定チャンネルを選択します。
UserSetLoad	ICommand	4	W	ユーザー設定の Load を実行します。
UserSetSave	ICommand	4	W	不揮発性メモリにユーザー設定の Save を実行します。
UserSetQuickSave	ICommand	4	W	揮発性メモリにユーザー設定の Save を実行します。
UserSetDefault	IEnumeration	4	R/W	カメラ起動時に Load するユーザー設定チャンネルを選択します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
UserSetSelector	Implemented	0x208060	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x20806C	4	R	[0] : Default [1] : UserSet1 … [15] : UserSet15
	Value	0x20807C	4	R/W	ユーザー設定チャンネルを選択します。
UserSetCommand	Implemented	0x208080	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x20808C	4	R	[0] : Done [8] : Load [9] : Save [120] : QuickSave
	Value	0x20809C	4	R/W	ユーザー設定コマンドを選択します。
UserSetDefault	Implemented	0x2080A0	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x2080AC	4	R	[0] : Default [1] : UserSet1 … [15] : UserSet15
	Value	0x2080BC	4	R/W	カメラ起動時に Load するユーザー設定チャンネルを選択します。

● TeliCamSDK 制御

Camera feature API

専用の API を使用して UserSetControl を制御します。

API 名	説明
ExecuteCamUserSetLoad	UserSetLoad を実行します。
ExecuteCamUserSetSave	UserSetSave を実行します。
ExecuteCamUserSetSaveAndSetDefault	UserSetDefault を実行します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用して UserSetControl を制御します。

◆UserSetLoad/UserSetSave/UserSetQuickSave

1.UserSetSelector によってユーザー設定チャンネルを選択します。

設定値は Enumeration 型で下記のとおりです。

Integer	String	説明	セーブ	ロード
0	Default	・ホワイトバランス未調整 ・DPC 機能未設定 ・その他機能は工場出荷設定と同じ	-	○
1	UserSet1（※）	ユーザー設定チャンネル 1 出荷時に下記機能調整済み ・ホワイトバランス ・欠陥画素座標設定、DPC 機能 ON	○	○
2～15	UserSet2～7	ユーザー設定チャンネル 2～7	○	○

※ 出荷設定

2.UserSetLoad, UserSetSave, UserSetQuickSave を実行します。

UserSetLoad を実行することで、UserSetSelector レジスタにて選択されているチャンネルからユーザー設定を Load します。

UserSetSave または UserSetQuickSave を実行することで、UserSetSelector レジスタに選択されているチャンネルにユーザー設定を Save します。

UserSetSave 実行後、Nd_GetCmdIsDone で UserSetSave の完了を待ちます。

```
// GenICam node handle
CAM_NODE_HANDLE   hSelector = NULL;
CAM_NODE_HANDLE   hSave = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “UserSetSelector”, &hSelector);
Nd_GetNode(s_hCam, “UserSetSave”, &hSave);

// 1.Select a channel of user setting by ‘UserSetSelector’.
uint64_t dat = 1; // UserSet1
Nd_SetEnumIntValue(s_hCam, hSelector, dat);

// 2. Execute UserSetSave
Nd_CmdExecute(s_hCam, hSave);

bool8_t bDone;
while(1) {
 Nd_GetCmdIsDone(s_hCam, hSave, &bDone);
 if(bDone == true)
  break;
 Sleep(0);
}
```

◆UserSetDefault

1.UserSetDefault によってカメラ起動時に Load するユーザー設定チャンネルを選択します。
設定値は Enumeration 型で下記のとおりです。

Integer	String	説明
0	Default	工場出荷設定と同じ
1～7	UserSet1～7	ユーザー設定チャンネル 1～7

```
// GenICam node handle
CAM_NODE_HANDLE   hSelector = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, “UserSetDefault”, &hSelector);

// 1.Select a channel of user setting when camera powers up by
 ‘UserSetDefault’.
uint64_t dat = 1; // UserSet1
Nd_SetEnumIntValue(s_hCam, hSelector, dat);
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IEnumeration node functions], [ICommand node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして UserSetControl を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆UserSetLoad/UserSetSave/UserSetQuickSave

1.UserSetSelector によってユーザー設定チャンネルを選択します。

UserSetSelector レジスタの Value フィールドに書き込みます。

2.UserSetLoad, UserSetSave, UserSetQuickSave を実行します。

UserSetLoad を実行するために、UserSetCommand レジスタの Value フィールドに[8]を書き込みます。

UserSetSave を実行するために、UserSetCommand レジスタの Value フィールドに[9]を書き込みます。

UserSetQuickSave を実行するために、UserSetCommand レジスタの Value フィールドに[120]を書き込みます

UserSetSave 実行後、UserSetCommand が Done（= 0）になるまで待ちます。

```
uint32_t dat;

// 1.Select a channel of user setting by ‘UserSetSelector’.
dat = 1; // UserSet1
Cam_WriteReg(s_hCam, 0x20807C, 1, &dat);

// 2. Execute UserSetSave
dat = 9; // UserSetSave
Cam_WriteReg(s_hCam, 0x20809C, 1, &dat);

while(1) {
  Cam_ReadReg(s_hCam, 0x20809C, 1, &dat);
  if(dat == 0)
    break;
  Sleep(0);
}
```

◆UserSetDefault

UserSetDefault によってカメラ起動時に Load するユーザー設定チャンネルを選択します。

```
// 1.Select a channel of user setting when camera powers up
uint32_t dat = 1; // UserSet1
Cam_WriteReg(s_hCam, 0x2080BC, 1, &dat);
```

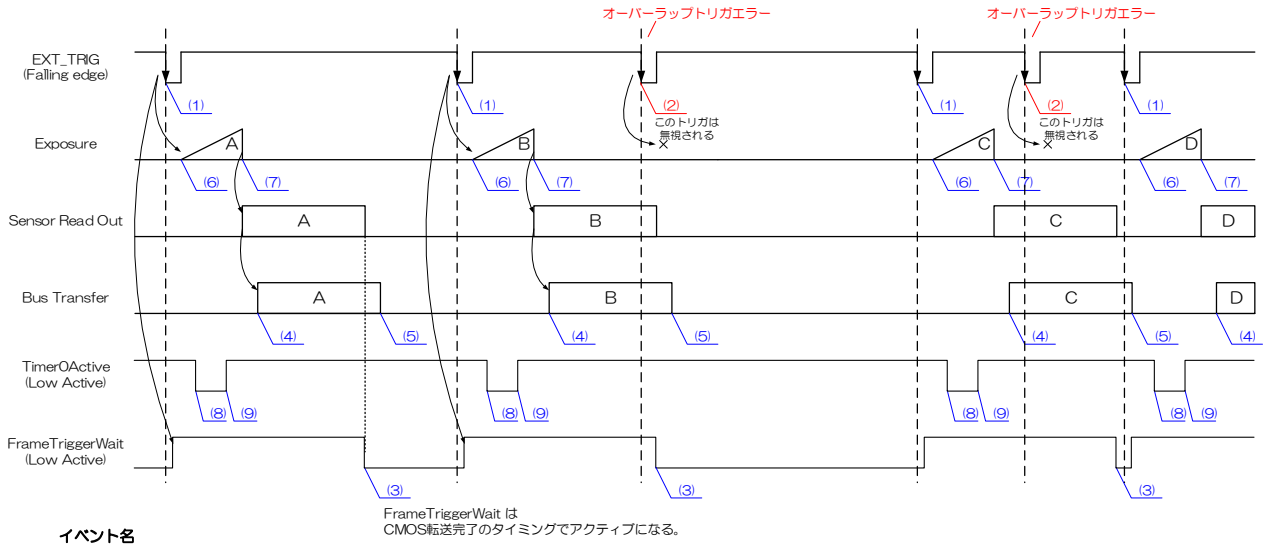
詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

EventControl

Event Packet 機能を用いて、トリガ受付状態などを取得することができます。

- FrameTrigger　　：トリガ受付
- FrameTriggerError　　：トリガエラー
- FrameTriggerWait　　：トリガ受付待ち開始
- FrameTransferStart　　：映像転送開始
- FrameTransferEnd　　：映像転送終了
- ExposureStart　　：露光開始
- ExposureEnd　　：露光終了
- Timer0Start　　：Timer0 開始
- Timer0End　　：Timer0 終了

イベントの発行タイミングは下図のようになります。



イベント名

(1) FrameTrigger　　：トリガ受付
(2) FrameTriggerError　　：トリガエラー
(3) FrameTriggerWait　　：トリガ受付待ち開始
(4) FrameTransferStart　　：映像転送開始
(5) FrameTransferEnd　　：映像転送終了
(6) ExposureStart　　：露光開始
(7) ExposureEnd　　：露光終了
(8) Timer0Start　　：Timer0開始
(9) Timer0End　　：Timer0終了

※BU2006MG/BU2006MCF は、オーバーラップトリガ不可

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
EventSelector	IEnumeration	4	R/W	イベント通知の種類を選択します。
EventNotification	IEnumeration	4	R/W	イベント通知の有効/無効を選択します。

イベント名	Event ID	Event Data	Length Byte / [bit]	説明
EventFrameTrigger	0x8020	EventFrameTriggerTimestamp	8	イベント発生時のタイムスタンプを返します。
EventFrameTriggeError	0x8021	EventFrameTriggerErrorTimestamp	8	イベント発生時のタイムスタンプを返します。
EventFrameTriggeWait	0x8022	EventFrameTriggerWaitTimestamp	8	イベント発生時のタイムスタンプを返します。
EventFrameTransferStart	0x8030	EventFrameTransferStartTimestamp	8	イベント発生時のタイムスタンプを返します。
EventFrameTransferEnd	0x8031	EventFrameTransferEndTimestamp	8	イベント発生時のタイムスタンプを返します。
EventExposureStart	0x8040	EventExposureStartTimestamp	8	イベント発生時のタイムスタンプを返します。
EventExposureEnd	0x8041	EventExposureEndTimestamp	8	イベント発生時のタイムスタンプを返します。
EventTimer0Start	0x9000	EventTimer0StartTimestamp	8	イベント発生時のタイムスタンプを返します。
EventTimer0End	0x9001	EventTimer0EndTimestamp	8	イベント発生時のタイムスタンプを返します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
EventNotificationOfFrame	Implemented	0x21F220	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x21F22C	4	R	[0] : FrameTrigger [1] : FrameTriggerError [2] : FrameTriggerWait [3] : FrameStart [4] : FrameEnd [16] : FrameTransferStart [17] : FrameTransferEnd
	Value	0x21F230	4	R/W	Fame イベント通知を有効にします。
EventNotificationOfExposure	Implemented	0x21F240	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x21F24C	4	R	[0] : ExposureStart [1] : ExposureEnd
	Value	0x21F250	4	R/W	UserSet コマンドを設定します。
EventNotificationOfTimerStart	Implemented	0x21F380	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x21F38C	4	R	[0] : Timer0Start
	Value	0x21F390	4	R/W	Timer0 開始イベント通知を有効にします。
EventNotificationOfTimerEnd	Implemented	0x21F3A0	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x21F3AC	4	R	[0] : Timer0End
	Value	0x21F3B0	4	R/W	Timer0 終了イベント通知を有効にします。

● TeliCamSDK 制御

Camera feature API

専用の API を使用して Event 機能を制御します。

API 名		説明
High-level API functions	Evt_OpenSimple	イベントインターフェースをオープンします。
	Evt_Activate	指定したカメライベントを有効にします。
	Evt_Deactivate	指定したカメライベントを無効にします。
Low-level API functions	Evt_Open	イベントインターフェースをオープンします。
	Evt_CreateRequest	イベントリクエストを作成します。
	Evt_ReleaseRequest	イベントリクエストを解放します。
	Evt_EnqueueRequest	イベントリクエストをイベント待機キューに投入します。
	Evt_DequeueRequest	イベント受信キューからイベントリクエストを一つ取り出します。
	Evt_FlushWaitQueue	すべての受信処理を停止し、待機キュー内のすべてのイベントリクエストを受信完了キューに移動させます。
Common functions	Evt_Close	イベントインターフェースをクローズします。

TeliCamSDK のインストールフォルダにある[TeliCamAPI Library manual]の
[Camera event notification functions]と[GrabEvent]のサンプルコードを参照してください。

GenICam function API

専用の API を使用して Event 機能を制御してください。

Register access API

専用の API を使用して Event 機能を制御してください。

LEDIndicatorLuminance

LED インジケータの輝度を設定します。

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
LEDIndicatorLuminance	IFloat	4	R/W	LED の輝度を設定します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
LEDIndicatorLuminance	Implemented	0x21F100	[31]	R	この機能が有効かどうかを返します。
	Mult	0x21F10C	4	R	絶対値［%］= Raw * (Mult / Div)
	Div	0x21F110	4	R	
	Min	0x21F114	4	R	LED の輝度最小値を返します。
	Max	0x21F118	4	R	LED の輝度最大値を返します。
	Value	0x21F11C	4	R/W	LED の輝度を設定します。

● TeliCamSDK 制御

GenICam function API

GeniCam API を使用して LED indicator luminance を制御します。

◆LEDIndicatorLuminance

LED indicator luminance を制御するには IFloat インターフェースを使用します。

```
// GenICam node handle
CAM_NODE_HANDLE   hNode = NULL;

// luminance = 50[%]
// actual value (4/7)*100 = 57.14[%]
float64_t      dLuminance = 50.0;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, "LEDIndicatorLuminance", &hNode);

// Set LED indicator luminance Value
Nd_SetFloatValue(s_hCam, hNode, dLuminance);
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IFloat node functions], [IEnumeration node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして LED indicator luminance を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆LEDIndicatorLuminance

LEDIndicatorLuminance レジスタの Value’フィールドに書き込みます。

```
// luminance = 50[%] (Raw value = 4)
// actual value (4/7)*100 = 57.14[%]
uint32_t      uiLuminanceRaw = 4;

// Set LED indicator luminance Value
Cam_WriteReg(s_hCam, 0x21F11C, 1, &uiLuminanceRaw);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

◆最小値 / 最大値 Value

LEDIndicatorLuminance	Raw 値	絶対値 (Float)
最小値	0	0.00 [%]
最大値	7	100.00 [%]
初期値	7	100.00 [%]
式	絶対値 [%] = Raw 値 / 7	

DPCControl

DPC(Defective Pixel Correction：欠陥画素補正)では、イメージセンサの欠陥画素を補正することが出来ます。欠陥画素の座標(X, Y)を指定することにより、指定座標の周囲画素値から演算をおこない、欠陥画素を補正します。

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
DPCEnable	IEnumeration	4	R/W	DPC 機能の ON/OFF を制御します。
DPCNumber	IInteger	4	R/W	補正する欠陥画素数を指定します。
DPCIndex	IInteger	4	R/W	設定する座標値の Index 番号を指定します。
DPCEntryX	IInteger	4	R/W	補正対象画素の X 座標を指定します。
DPCEntryY	IInteger	4	R/W	補正対象画素の Y 座標を指定します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
DPCEnable	Implemented	0x21F280	[31]	R	この機能が有効かどうかを返します。
	ListOfElements	0x21F28C	4	R	[0] : Off [1] : On
	Value	0x21F29C	4	R/W	DPC 機能の ON/OFF を制御します。
DPCNumber	Implemented	0x21F2A0	[31]	R	この機能が有効かどうかを返します。
	Value	0x21F2BC	4	R/W	補正する欠陥画素数を指定します。
DPCCoord	Value[0][0]	0x400000	4	R/W	欠陥画素の最初の X 座標を設定します。
	Value[0][1]	0x400004	4	R/W	欠陥画素の最初の Y 座標を設定します。
	Value[1][0]	0x400008	4	R/W	欠陥画素の 2 番目の X 座標を設定します。
	Value[1][1]	0x40000C	4	R/W	欠陥画素の 2 番目の Y 座標を設定します。
	…	…	…	…	…
	Value[255][0]	0x4007F8	4	R/W	欠陥画素の 255 番目の X 座標を設定します。
	Value[255][1]	0x4007FC	4	R/W	欠陥画素の 255 番目の Y 座標を設定します。

● TeliCamSDK 制御

GenICam function API

GeniCam API を使用して DPC を制御します。

◆DPC

1.DPCIndex によって座標値の Index 番号を、DPCEntryX によって補正対象画素の X 座標を、DPCEntryY によって補正対象画素の Y 座標をそれぞれ設定します。

2.DPCNumber によって補正する欠陥画素数を設定します。

3.DPCEnable によって DPC 機能の On/Off を設定します。

設定値は Enumeration 型で下記のとおりです。

Integer	String
0	Off
1	On

```
// GenICam node handle
CAM_NODE_HANDLE   hIndex = NULL;
CAM_NODE_HANDLE   hEntryX = NULL;
CAM_NODE_HANDLE   hEntryY = NULL;
CAM_NODE_HANDLE   hNumber = NULL;
CAM_NODE_HANDLE   hEnable = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, "DPCIndex", &hIndex);
Nd_GetNode(s_hCam, "DPCEntryX", &hEntryX);
Nd_GetNode(s_hCam, "DPCEntryY", &hEntryY);
Nd_GetNode(s_hCam, "DPCNumber", &hNumber);
Nd_GetNode(s_hCam, "DPCEnable", &hEnable);

int64_t entry[2][2] = {{100,200},{150,300}};   // {x,y}
int64_t number;

for(number=0; number<2; number++)
{
  // 1.Set the coordinates of defective pixels by 'DPCIndex', 'DPCEntryX'
and 'DPCEntryY'.
  Nd_SetIntValue(s_hCam, hIndex, number); // 0 origin
  Nd_SetIntValue(s_hCam, hEntryX, entry[number][0]);
  Nd_SetIntValue(s_hCam, hEntryY, entry[number][1]);
}

// 2.Sets the number of pixels to correct to 'DPCNumber'.
Nd_SetIntValue(s_hCam, hNumber, number); // 2 pixels are to be
corrected.

// 3.Sets the activation of DPC function by 'DPCEnable'.
Nd_SetEnumStrValue(s_hCam, hEnable, "On");
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IInteger node functions], [IEnumeration node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして DPC を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆DPC

1.DPC 対象画素の座標を設定します。

DPCCoord レジスタの Value[index][0], Value[index][1]フィールドに書き込みます。

Value レジスタの第 1 インデックスは座標値の Index 番号を示します。

第 2 インデックスの[0], [1]はそれぞれ X, Y 座標に対応します。

2.補正する欠陥画素数を設定します。

DPCNumber レジスタの Value フィールドに書き込みます。

3.DPC 機能を有効にします。

DPCEnable レジスタの Value フィールドに[1]を書き込みます。

```
// 1.Set the coordinates of defective pixels.
uint32_t entry[2][2] = {{100,200},{150,300}}; // {x,y}
uint32_t number;
uint64_t addr = 0x400000;
for(number=0; number<2; number++, addr+= 0x8)
{
        Cam_WriteReg(s_hCam, addr     , 1, &entry[number][0]);
        Cam_WriteReg(s_hCam, addr+0x4, 1, &entry[number][1]);
}

// 2.Sets the number of pixels to correct.
Cam_WriteReg(s_hCam, 0x21F2BC, 1, &number);

// 3.Set the activation of DPC function.
uint32_t dat = 1;
Cam_WriteReg(s_hCam, 0x21F29C, 1, &dat);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。

◆最小値／最大値

	DPCNumber	DPCIndex
最小値	0	0
最大値	256	255

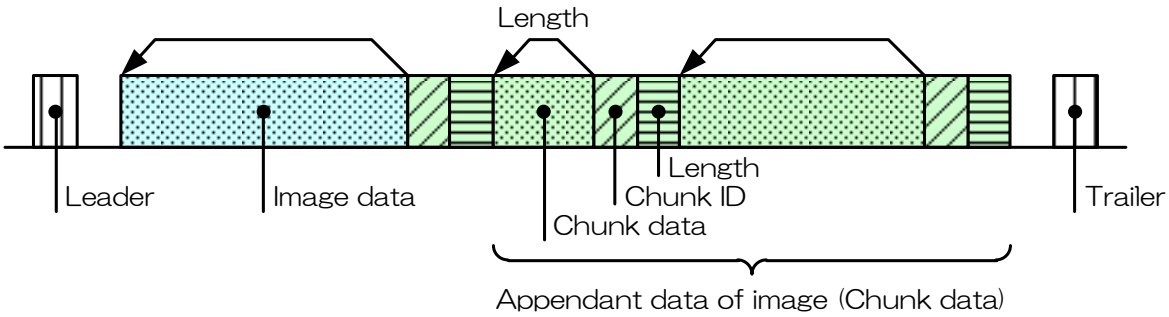
	DPCEntryX	DPCEntryY
最小値	0	0
最大値	WidthMax-1	HeightMax-1

Chunk

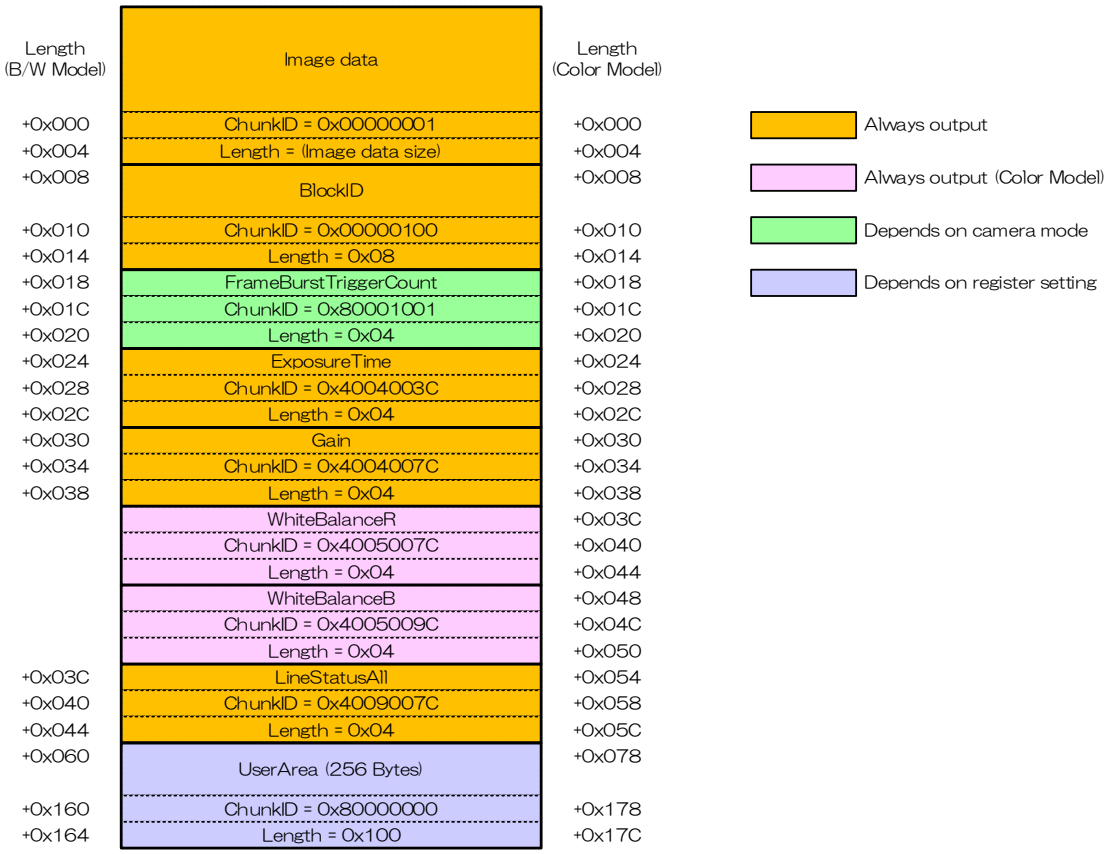
Chunk データとは画像データ毎に付加されたタグ情報を指します。

このタグ情報はアプリケーションがデータのペイロードを解析して様々な要素を抽出・識別できるようにするものです。

有効化された Chunk データの内容が多くなると、そのフレーム長は長くなります。



Chunk



Chunk データの構造

● GenICam ノード

名称	Interface	Length Byte / [bit]	Access	説明
ChunkModeActive	IBoolean	4	R/W	Chunk 機能を有効にします。
ChunkSelector	IEnumeration	4	R/W	有効にする Chunk データを選択します。
ChunkEnable	IBoolean	4	R/W	画像データのペイロードに Chunk データを付加します。
ChunkUserAreaLength	IInteger	4	R	ChunkUserAreaTable の長さを示します。
ChunkUserAreaTable	IString	256	R/W	ユーザー文字列を設定します。（最大：256byte）
ChunkFrameID	IInteger	8	R	Chunk データに付加された Block ID を返します。
ChunkExposureTime	IFloat	4	R	Chunk データに付加された ExposureTime の値を返します。
ChunkGain	IFloat	4	R	Chunk データに付加された Gain の値を返します。
ChunkWhiteBalanceR	IFloat	4	R	Chunk データに付加された WhiteBalanceR gain の値を返します。
ChunkWhiteBalanceB	IFloat	4	R	Chunk データに付加された WhiteBalanceB gain の値を返します。
ChunkLineStatusAll	IInteger	4	R	Chunk データに付加された LineStatusAll を返します。
ChunkFrameBurstTriggerCount	IInteger	4	R	Chunk データに付加された FrameBurstTriggerCount の値を返します。

● IIDC2 レジスタ

レジスタ名	Field	Address	Length Byte / [bit]	Access	説明
ChunkModeActive	Implemented	0x21D020	[31]	R	この機能が有効かどうかを返します。
	BitWritable	0x21D02C	4	R	[0]:Off [1]:On
	Value	0x21D030	4	R/W	Chunk 機能を有効にします。
ChunkEnableOfVendorSpecific	Implemented	0x21D040	[31]	R	この機能が有効かどうかを返します。
	BitWritable	0x21D04C	4	R	[0]：BlockID [8]：FrameBurstTriggerCount [9]：SequentialShutterNumber [10]：SequentialShutterElement [24]：UserArea
	Value	0x21D050	4	R/W	Chunk データを有効にします。
ChunkEnableOfCat4	Implemented	0x21D0E0	[31]	R	この機能が有効かどうかを返します。
	BitWritable	0x21D0EC	4	R	[0]：ExposureTime [2]：Gain
	Value	0x21D0F0	4	R/W	Chunk データを有効にします。
ChunkEnableOfCat5	Implemented	0x21D100	[31]	R	この機能が有効かどうかを返します。
	BitWritable	0x21D10C	4	R	[2]：WhiteBalaceR [3]：WhiteBalaceB
	Value	0x21D110	4	R/W	Chunk データを有効にします。
ChunkEnableOfCat9	Implemented	0x21D180	[31]	R	この機能が有効かどうかを返します。
	BitWritable	0x21D18C	4	R	[2]：LineStatusAll
	Value	0x21D190	4	R/W	Chunk データを有効にします。
ChunkUserArea	Implemented	0x21D7F0	[31]	R	この機能が有効かどうかを返します。
	NumberOf Elements	0x21D7FC	[30:0]	R	ChunkUserArea の長さ(byte 単位)を返します。
	Value[0]	0x21D800	length	R/W	ChunkUserArea の ASCII 文字列を設定します。
	…	…	…		
	Value[last]	0x21D800 +NumberOf Elements -4		R/W	ChunkUserArea の ASCII 文字列を設定します。

● TeliCamSDK 制御

Camera feature API

専用の API を使用して Event 機能を制御します。

API 名	説明
GetCamChunkModeActive	カメラのチャンクデータ出力モード(有効/無効)を取得します。
SetCamChunkModeActive	カメラのチャンクデータ出力モード(有効/無効)を設定します。

詳細は[TeliCamAPI Library manual]の[Controlling camera feature functions]を参照してください。

GenICam function API

GeniCam API を使用して Chunk を制御します。

◆Chunk

1.映像ストリームをクローズします。

2.ChunkModeActive によって Chunk 機能を有効にします。

3.ChunkSelector と ChunkEnable によって Chunk データを有効にします。

ほとんどの Chunk データは出力に固定されています。

4.ChunkUserAreaTable によってユーザー文字列を設定します。（任意）

5.映像ストリームをオープンします。

6.映像ストリームをキャプチャします。

TeliCamAPI はコマンド処理やバッファ処理などのストリーミング機能を提供し映像ストリームを簡単にキャプチャします。

TeliCamSDK インストールフォルダ内の[TeliCamAPI Library manual]の[Camera streaming functions]と[GrabStreamSimple]のサンプルコードを参照してください。

7.Chunk データを抽出します。

7-1.Chunk_AttachedBuffer でメモリを確保します。

7-2.Chunk データを読み出します。

```
// GenICam node handle
CAM_NODE_HANDLE   hMode = NULL;
CAM_NODE_HANDLE   hSelector = NULL;
CAM_NODE_HANDLE   hEnable = NULL;
CAM_NODE_HANDLE   hFrameID = NULL;
CAM_NODE_HANDLE   hExposureTime = NULL;
CAM_NODE_HANDLE   hUserAreaTable = NULL;

// Retrieve GenICam node.
Nd_GetNode(s_hCam, "ChunkModeActive", &hMode);
Nd_GetNode(s_hCam, "ChunkSelector", &hSelector);
Nd_GetNode(s_hCam, "ChunkEnable", &hEnable);
Nd_GetNode(s_hCam, "ChunkFrameID", &hFrameID);
Nd_GetNode(s_hCam, "ChunkExposureTime", &hExposureTime);
Nd_GetNode(s_hCam, "ChunkUserAreaTable", &hUserAreaTable);

// 2.Activate ChunkModeActive
Nd_SetBoolValue(s_hCam, hMode, true);

//3.Enable ChunkSelector
Nd_SetEnumStrValue(s_hCam, hSelector, "ExposureTime");
Nd_SetBoolValue(s_hCam, hEnable, true);
Nd_SetEnumStrValue(s_hCam, hSelector, " UserArea");
Nd_SetBoolValue(s_hCam, hEnable, true);

// 4.Set the user string
Nd_SetStrValue(s_hCam, hUserAreaTable, "Test");
```

```
// 5.6.Open and capture image
Strm_ReadCurrentImage(hStrm, pvPayloadBuf, &uiPyldSize, &sImageInfo);

// 7-1. Attach Buffer
Chunk_AttachBuffer(s_hStrm, pvPayloadBuf, PyldSize)

// 7-2.Get FrameID of Chunk data.
int64_t fid= 0;
Nd_GetIntValue(s_hCam, hFrameID, &fid);

// 7-2.Get Exposure Time of Chunk data.
float64_t   exptime = 0;
Nd_GetFloatValue(s_hCam, hExposureTime, &exptime);

// 7-2.Get User Area data of Chunk data.
char userarea[256];
Uint32_t uiSize = 256;
Nd_GetStrValue(s_hCam, hUserAreaTable, &userarea, &uiSize);
```

詳細は[TeliCamAPI Library manual]の[INode functions], [IBoolean functions], [IInteger node functions], [IEnumeration node functions]を参照してください。

Register access API

IIDC2 レジスタに直接アクセスして Chunk を制御します。

API 名	説明
Cam_ReadReg	値を読み出します。
Cam_WriteReg	値を書き込みます。

◆Chunk

1.映像ストリームをクローズします。

2.Chunk 機能を有効にします。

ChunkModeActive レジスタの Value フィールドに書き込みます。

3.Chunk データを有効にします。

ChunkEnableOfVendorSpecific, ChunkEnableOfCat4, ChunkEnableOfCat5, ChunkEnableOfCat9 レジスタの Value フィールドに書き込みます。

4.ユーザー文字列を設定します。（任意）

ChunkUserArea レジスタの Value[0]～Value[last]フィールドに書き込みます。

5.映像ストリームをオープンします。

6.映像ストリームをキャプチャします。

TeliCamAPI はコマンド処理やバッファ処理などのストリーミング機能を提供し映像ストリームを簡単にキャプチャします。

TeliCamSDK インストールフォルダ内の[TeliCamAPI Library manual]の[Camera streaming functions]と[GrabStreamSimple]のサンプルコードを参照してください。

7.GenICam API を使用して Chunk データを抽出します。

```
// 2.Activate ChunkModeActive
int32_t active = 1;
Cam_WriteReg(s_hCam, 0x21D030,1, &active);

// 3.Enable Chunk
int32_t cat4 = 5;
Cam_WriteReg(s_hCam, 0x21D0F0,1, &cat4);

// 5.6.Open and capture image
Strm_ReadCurrentImage(hStrm, pvPayloadBuf, &uiPyldSize, &sImageInfo);

// 7-1. Attach Buffer
Chunk_AttachBuffer(s_hStrm, pvPayloadBuf, PyldSize)

// 7-2.Get FrameID of Chunk data.
int64_t fid= 0;
Nd_GetIntValue(s_hCam, hFID, &fid);

// 7-2.Get Exposure Time of Chunk data.
float64_t   exptime = 0;
Nd_GetFloatValue(s_hCam, hExposureTime, &exptime);

// 7-2.Get User Area data of Chunk data.
char userarea[256];
uint32_t uiSize = 256;
Nd_GetStrValue(s_hCam, hUserAreaTable, &userarea, &uiSize);
```

詳細は[TeliCamAPI Library manual]の[Camera functions]を参照してください。.

付録

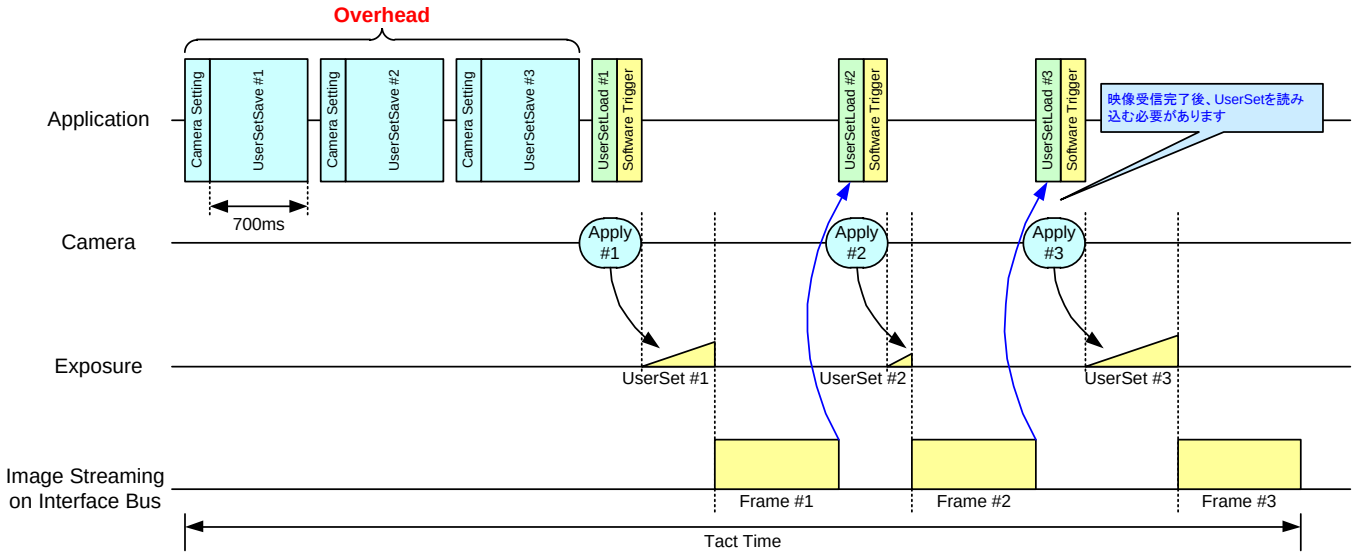
UserSetSave と UserSetQuickSave の違い

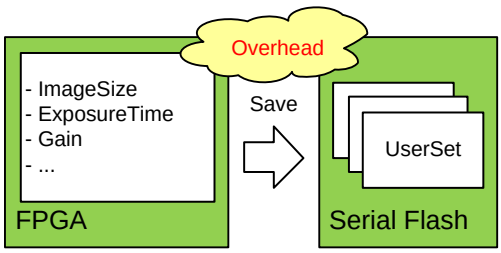
一度に複数のカメラの設定値（ROI 設定など）を変更したい場合は、ユーザーセット機能が便利です。

アプリケーションは事前または初期段階でカメラの設定値をユーザーセットのメモリ内に保存する必要があります。

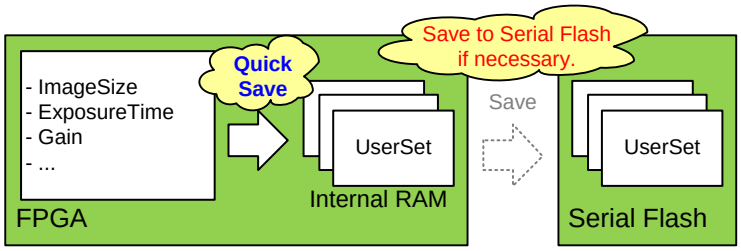
ユーザーセットは **UserSetSave** によって不揮発性フラッシュメモリに格納されます。

書込む前に不揮発性フラッシュメモリを消去する必要があるため、UserSetSave の実行には**約 700m s** の時間を要します。複数の異なる設定の切り替えを実行する場合、**UserSetSave** の処理時間は避けられません。





ユーザーセットは UserSetQuickSave によって FPGA 内部の RAM に格納されます。
UserSetQuickSave の実行には 100u s 以下の時間を要します。
これにより UserSetSave の処理時間を大幅に短縮することができます。
必要に応じて不揮発性フラッシュメモリにユーザーセットを保存することも可能です。



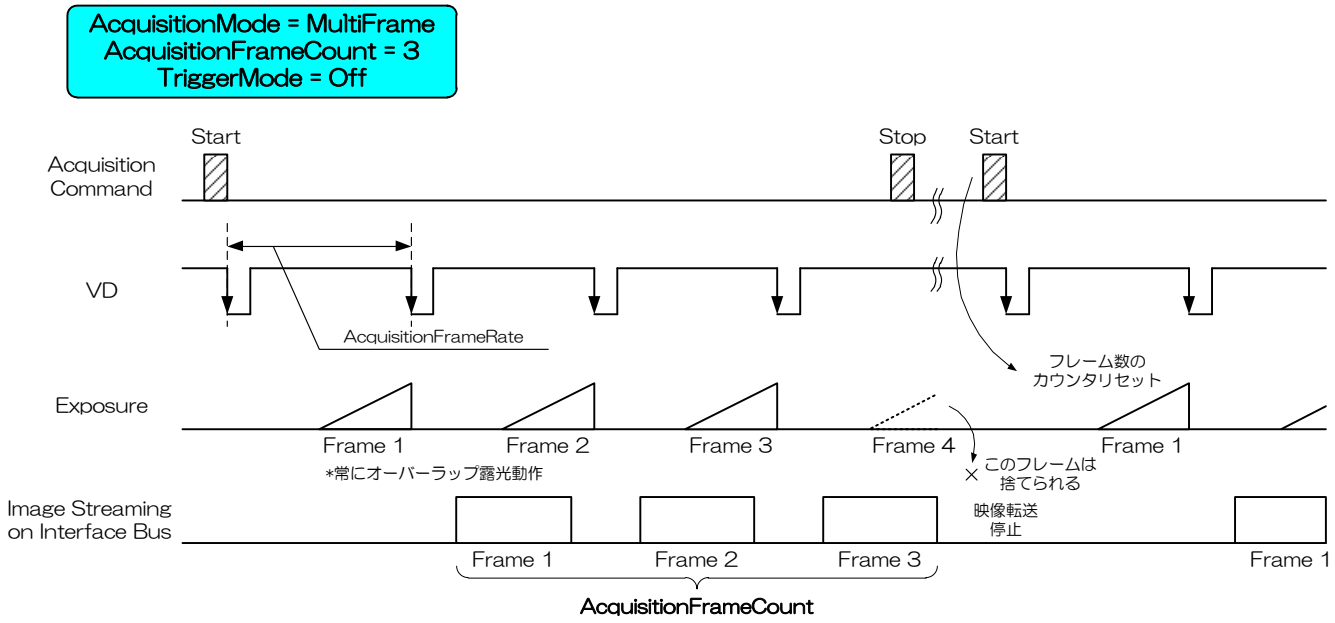
MultiFrame と Bulk モード動作の違い

本項では、AcquisitionMode の MultiFrame 動作と、TriggerSequence の Bulk モード動作の違いについて説明します。

-MultiFrame は、転送するフレーム枚数を AcquisitionFrameCount レジスタにより設定します。
-Bulk モードは、露光するフレーム枚数を TriggerAdditionalParameter レジスタにより設定します。

-MultiFrame 動作（ノーマルシャッタモード：TriggerMode = Off）

カメラは AcquisitionFrameCount で設定された枚数のフレームを転送します。

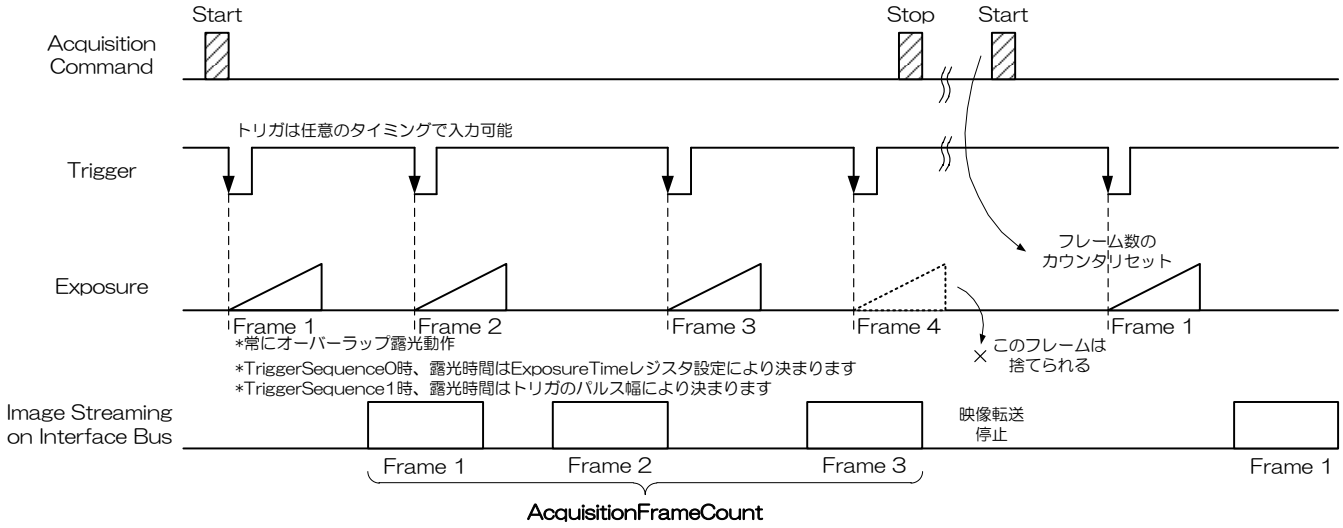


-MultiFrame 動作（ランダムトリガシャッタモード：TriggerMode = On, TriggerSequence = 0 or 1）

カメラは AcquisitionFrameCount で設定された枚数のフレームを転送します。

AcquisitionFrameCount の回数分のトリガ入力が必要です。

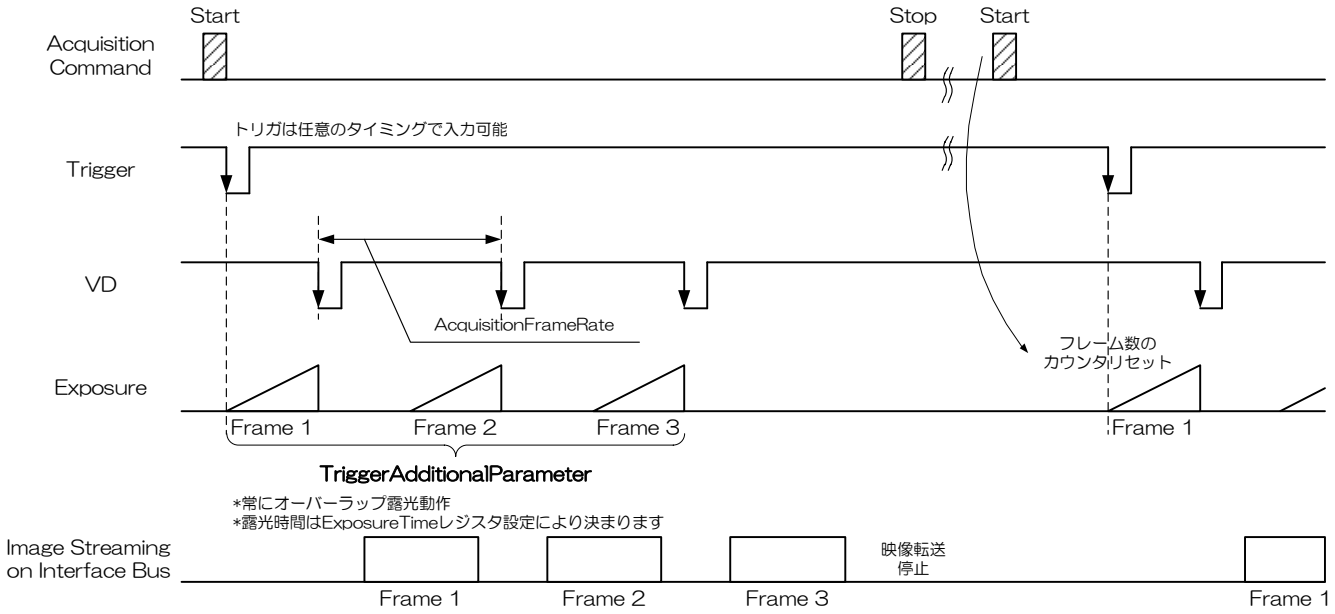
AcquisitionMode = MultiFrame
AcquisitionFrameCount = 3
TriggerMode = On
TriggerSequence = 0 or 1



-Bulk モード動作（ランダムトリガシャッタモード：TriggerMode = On, TriggerSequence = 6）

カメラは 1 回のトリガで、TriggerAdditionalParameter で設定された枚数のフレームを転送します。

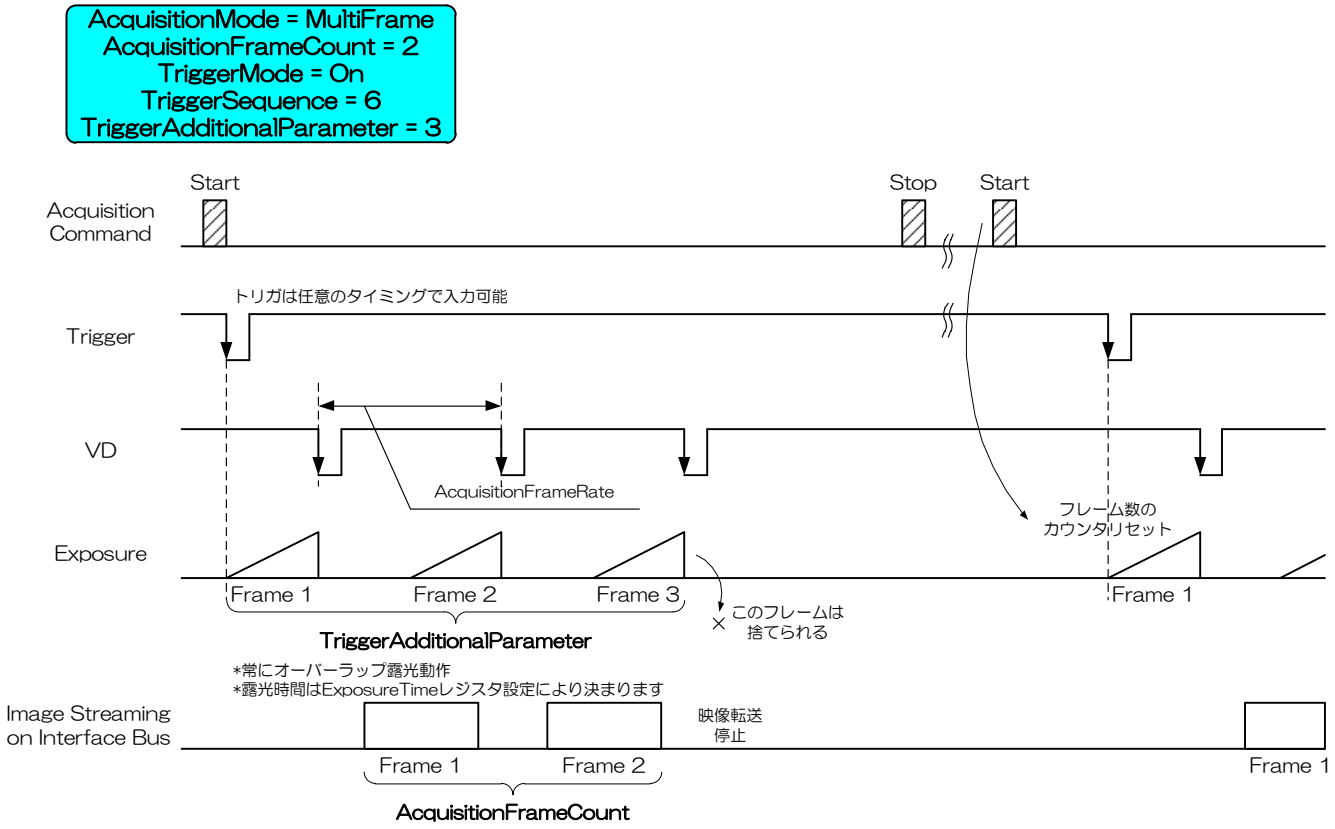
AcquisitionMode = Continuous
TriggerMode = On
TriggerSequence = 6
TriggerAdditionalParameter = 3



● 備考

Bulk モード設定時は、通常、AcquisitionMode を Continuous に設定してください。

AcquisitionMode を MultiFrame 設定にすることもできますが、その場合、フレーム数は AcquisitionFrameCount に制限されます。



保証規定

● 無償保証期間

保証期間はお客様お買い上げ後 36 ヶ月です。ただし、お買い上げ日が不明な場合、弊社出荷日から判断させていただきます。

● 無償保証対象外範囲

下記の場合の故障・損傷・損失は無償保証の対象外とさせていただきます。

1. 消耗部品の自然消耗、磨耗、劣化した場合

2. 取扱説明書記載の使用方法や使用条件、または注意に反したお取扱による場合

3. 改造・調整や部品交換による場合。(本体ケースの開封及び改造など)

4. 構成品に含まれる付属品または弊社指定オプション品を使用していなかった場合

5. お客様のお手元に渡った後の輸送、移動時の落下等お取り扱いの不備、腐食性のある環境・日光・火・砂・土・熱・湿気への放置、不適当な収納方法による場合

6. 火災・地震･水害・落雷・その他の天災、公害や漏電、異常電圧、過度な物理的圧力、盗難・その他の事故による場合

7. 相互接続に対する推奨のない製品へ接続した場合

8. 正しくない電源に接続した場合

9. 偽造製品・弊社のシリアル番号のない製品・シリアル番号が変造、汚損、削除された製品

10.無償保証期間満了後に起こったすべての欠陥

修理

● 修理方法

代替品または同等機能製品への交換対応となります。

● 修理依頼方法

修理ご依頼の際は弊社ホームページより「故障状況調査書」をダウンロードいただき、必要事項をご記入のうえ、弊社製品単品とあわせてご依頼ください。

故障修理依頼

https://www.toshiba-teli.co.jp/support/failure-situation.htm

なお、修理ご依頼の際には、以下の注意事項をご確認いただきますようお願いいたします。

1. お客様装置に組み込まれた状態での修理は受付けておりませんので、弊社製品構成外の物品が添付されている場合は、お客様にて取り外しを行い発送ください。
2. お客様添付の機番，管理番号，識別シールなどの情報は、ご返却はできませんので、お客様にて取り外しや、メモなど記録をお取り頂けます様、お願いいたします。
3. カメラ内部に保存されたデータは、修理後保持されませんので、発送前にデータの取り出しをお願いいたします。
4. お客様の都合による修理依頼後のキャンセルはお受けしておりません。
5. 修理品運送費につきましては、お客様から弊社宛の送料はお客様にご負担いただきます。弊社からお客様宛の送料は、無償期間内に限り、弊社が負担いたします。
6. 配送の日時指定について製品の配送日や配送時間帯、配送方法はご指定できませんのでご了承ください。
7. 故障要因調査、修理報告書のご依頼は受付けておりません。
8. 無償修理期間経過後の修理は、修理可能なものに限り有償にてお受けいたします。
9. 交換修理後の修理依頼品の所有権は弊社に帰属します。
10. 修理完了品においても製品の免責事項が適用されます。

※ソフトウェアに関するお問い合わせは、弊社ホームページまたは、弊社営業担当までお問い合わせください。