Programmer´s Guide for feature HDR

AN201414/v02/2017-03-03

Description

HDR (High Dynamic Range) is a technique that allows a greater dynamic range of luminance between the lightest and darkest areas of an image.

This document describes the programming of the HDR parameters.

Products

The following camera models support the HDR feature:

GigE	Dual-GigE	USB	Camera Link®
VisiLine®	HXG (Rel. 2)	MXU	LXC
VLG-22M / VLG-22C VLG-40M / VLG-40C	all models	MXUC20 / MXUC20c.2 MXUC40.2 / MXUC40c.2	(except LXC-250M / LXC-250C)
VisiLine® IP	LXG		
VLG-22M.I / VLG-22C.I VLG-40M.I / VLG-40C.I	all models (except LXG-250M / LXG-250C)		
MXG			
MXGC20 / MXGC20c MXGC40 / MXGC40c			

Notice

For HX cameras with Camera Link® interface the feature is controlled via register.

Notice

The function *HDREnableTriggerAutoMode* is only supported by HX cameras.

Preparation

The following SDKs support the HDR feature:

Baumer GAPI SDK
Baumer GAPI SDK v2.1 (and higher)

Supported Programming Languages

With the following programming languages, the HDR parameters are adjustable:

Programming Languages
C++
C#

Table of Contents

1 General Information

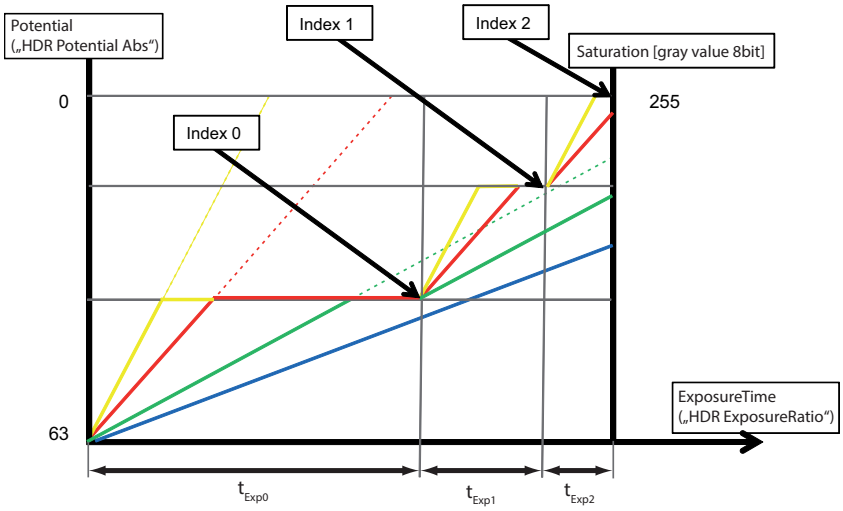
1.1 Normal HDR

The HDR mode limits the capacity of a pixel for a predetermined time. The capacity and the time can be set for two knee points in the camera.

For each index a time (*HDRExposureRatio*) and a capacity (potential → *HDRPotentialAbs*) can be chosen.

The *HDRExposureRatio* value has a range of 1...255 (1%...100%). The *HDRPotentialAbs* value has a range of 0...63 (0 = no limit; 63 = maximum limit).

The diagram below is used for further explanations.



Case 1 (low illumination) = blue line:

The illumination which arrives at the sensor is very low, so no knee points are affected. There is no difference between HDR mode enabled / disabled.

Case 2 (medium illumination) = green line:

The illumination which arrives at the sensor is low, only first knee point is affected. If there was no HDR mode enabled, the small dotted green line would show the resulting gray value.

Case 3 (high illumination) = red line:

The illumination which arrives at the sensor is high, so that both knee points are affected. As the red dotted line shows, the pixel would be saturated after just half of the total exposure time, if no HDR mode would be enabled.

Case 4 (very high illumination) = yellow line:

The illumination which arrives at the sensor is very high, so that both knee points are affected and the pixel is still saturated. If there would be no HDR mode enabled, the pixel would be saturated after just one third of the total exposure time, as can be seen from the dotted yellow line.

We like to compare a pixel of a camera with a container, which will be filled with photons.

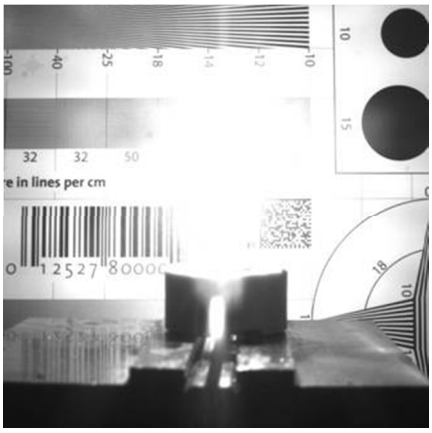
If a "container" is filled with photons, the limit (*HDRPotentialAbs Index 0*) will be reached eventually. No more photons can be filled into the container. After a certain time (*HDRExposureRatio Index 0*), the capacity of the container will be increased to *HDRPotentialAbs Index 1* and more photons can be filled in and so on.

This happens until the total exposure time is reached.

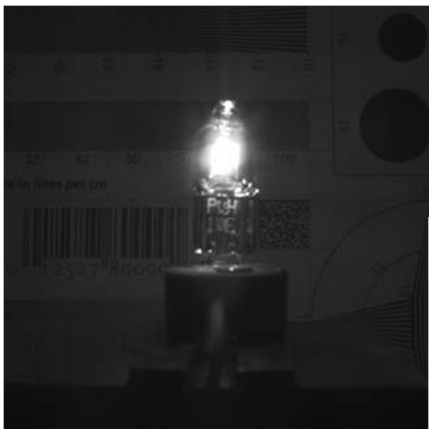
With the HDR mode it is possible to make very bright (saturated) pixels darker, but it is not possible to make dark pixels brighter. Furthermore it is not possible to adjust the values for *HDRPotentialAbs* and *HDRExposureRatio Index 2*. This is always 0 (*HDRPotentialAbs*) respectively is automatically calculated (*HDRExposureRatio Index 2 = total exposure time - HDRExposureRatio Index 1 - HDRExposureRatio Index 0*)

If you have an image on which you would like the HDR mode, adjust the total exposure time to a suitable value, so the dark parts of the image fit to your requirements. After that the saturated pixels have to be "damped". This can be done by changing the limit of the pixel or by changing the exposure time of a certain time slot.

Example



HDR Off



HDR On

1.2 Trigger controlled HDR

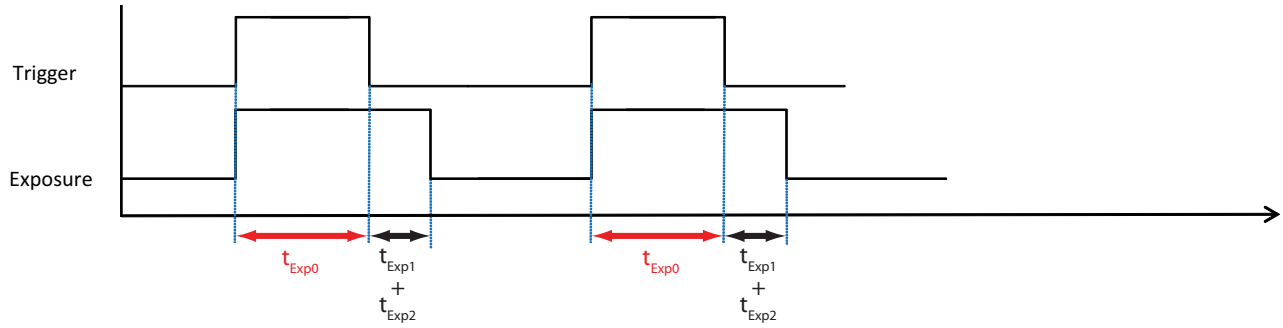
The *HDREnableTriggerAutoMode* is used when ExposureMode *TriggerControlled* or *TriggerWidth* is chosen. This means, *HDREnableTriggerAutoMode* is only useful when ExposureTime is controlled by pulsewidth of triggersource.

In general, when *HDREnableTriggerAutoMode* is enabled, the ExposureTime for the first exposure slot (t_{Exp0}) is controlled by trigger and the other two exposure time sections are automatically calculated according to HDR settings.

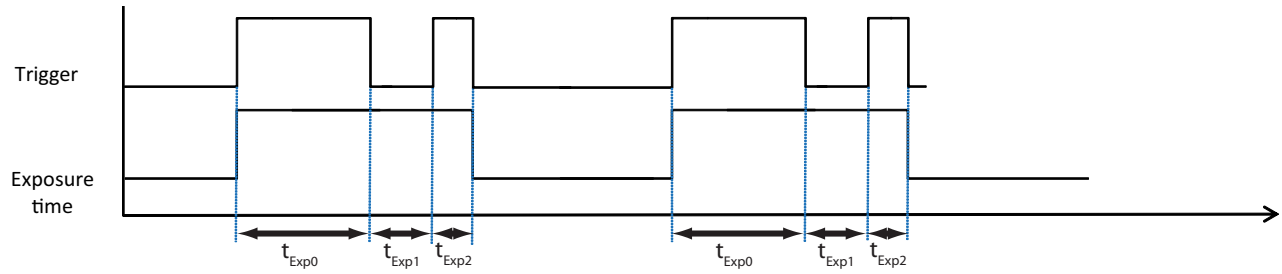
When *HDREnableTriggerAutoMode* is disabled, ExposureTime for all three sections is controlled by trigger.

Below all four possibilities are shown schematically (in this examples, the triggering is carried out by *RisingEdge*).

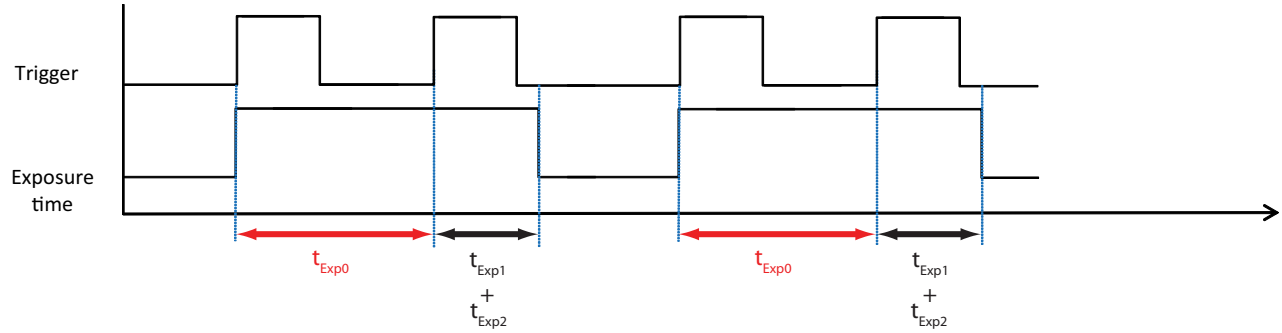
HDR EnableTriggerAutoMode = True / ExposureMode = TriggerWidth



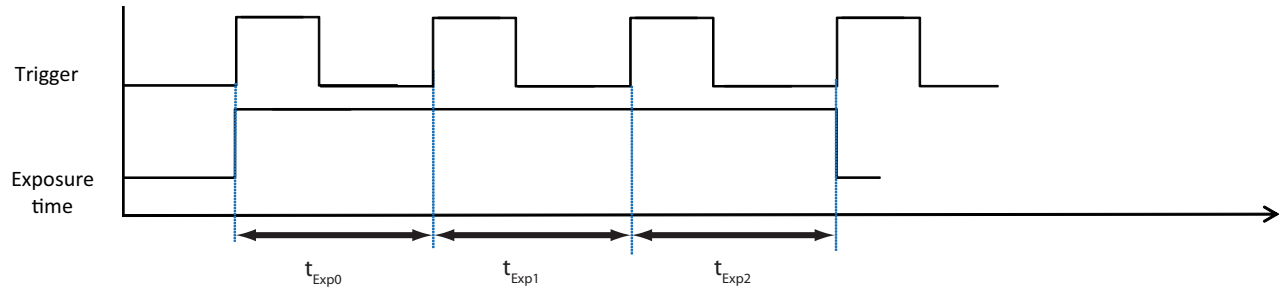
HDR EnableTriggerAutoMode = False / ExposureMode = TriggerWidth



HDR EnableTriggerAutoMode = True / ExposureMode = TriggerControlled



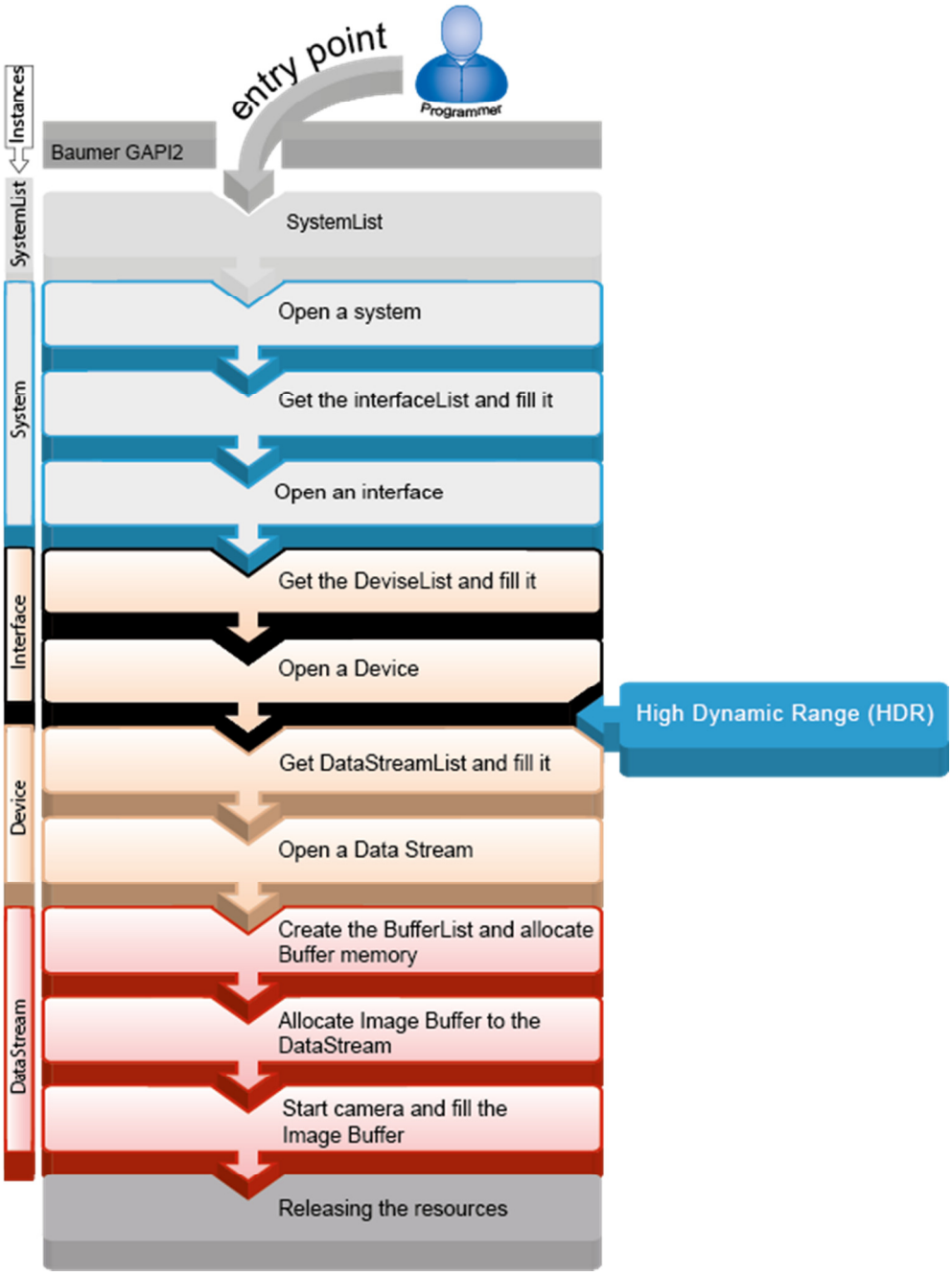
HDR EnableTriggerAutoMode = False / ExposureMode = TriggerControlled



2 HDR Settings

The following figure shows where the feature HDR is located in the program sequence.

Notice
To increase the clarity, the code for initialization and image grabbing is removed. Only the changing of the HDR parameters is shown.

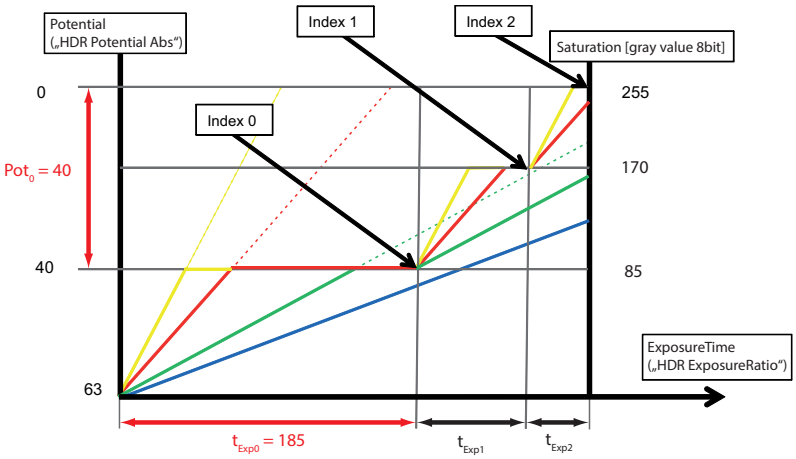


2.1 Normal HDR

2.1.1 C++

In this chapter, the setting of *Normal HDR* in C++ is shown.

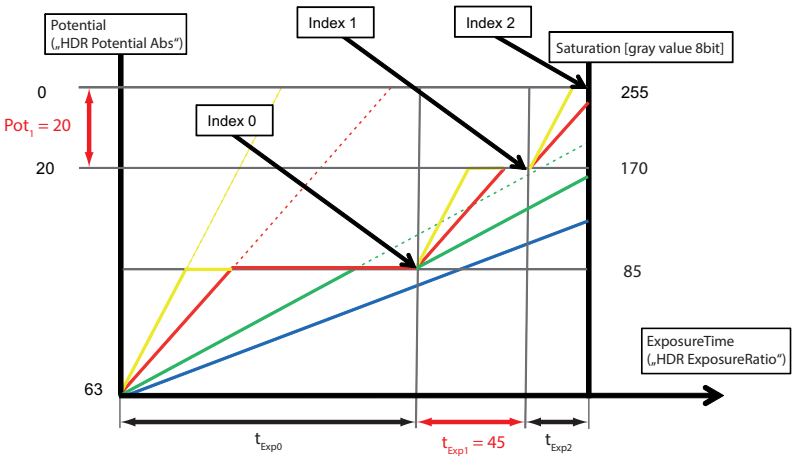
```
SystemList
Open a System
Get the InterfaceList and fill it
Open an Interface
Get the DeviceList and fill it
Open a Device


//Device Parameter Setup

pDevice->GetRemoteNode("TriggerMode")->SetString("Off");

pDevice->GetRemoteNode("ExposureTime")->SetDouble(10000.0);

std::cout << " ExposureTime : "
          << pDevice->GetRemoteNode("ExposureTime")->GetDouble()
          << std::endl


//HDR parameter change

std::cout << "HDR parameter change" << std::endl;

pDevice->GetRemoteNode("HDREnable")->SetBool(true);

std::cout << "HDREnable : "
          << pDevice->GetRemoteNode("HDREnable")->GetBool()
          << std::endl;


pDevice->GetRemoteNode("HDRIndex")->SetInt(0);

std::cout << "HDRIndex: "
          << pDevice->GetRemoteNode("HDRIndex")->GetInt()
          << std::endl;

pDevice->GetRemoteNode("HDRExposureRatio")->SetInt(185); //t_Exp_0

std::cout << "HDRExposureRatio: "
          << pDevice->GetRemoteNode("HDRExposureRatio")->GetInt()
          << std::endl;

std::cout << "HDRExposureRatioPercent : "
          << pDevice->GetRemoteNode("HDRExposureRatioPercent")->GetDouble()
          << std::endl;

pDevice->GetRemoteNode("HDRPotentialAbs")->SetInt(40); //Pot_0
std::cout << "HDRPotentialAbs : "
          << pDevice->GetRemoteNode("HDRPotentialAbs")->GetInt()
          << std::endl;
```




```
pDevice->GetRemoteNode("HDRIndex")->SetInt(1);
std::cout << "HDRIndex: "
          << pDevice->GetRemoteNode("HDRIndex")->GetInt()
          << std::endl;
pDevice->GetRemoteNode("HDRExposureRatio")->SetInt(45); //t_Exp_1
std::cout << "HDRExposureRatio: "
          << pDevice->GetRemoteNode("HDRExposureRatio")->GetInt()
          << std::endl;
std::cout << "HDRExposureRatioPercent: "
          << pDevice->GetRemoteNode("HDRExposureRatioPercent")->GetDouble()
          << std::endl;
pDevice->GetRemoteNode("HDRPotentialAbs")->SetInt(20); //Pot_1
std::cout << "HDRPotentialAbs: "
          << pDevice->GetRemoteNode("HDRPotentialAbs")->GetInt()
          << std::endl;
```

Notice

The value for t_{Exp2} will be calculated automatically in the camera.
($t_{Exp2} = t_{exposure} - t_{Exp0} - t_{Exp1}$
Pot_2 is always 0.



```
Get DataStreamList and fill it
Open a Data Stream
Create the BufferList and allocate Buffer memory
Allocate Image Buffer to the DataStream
Start Camera and fill the Image Buffer
Releasing the resources
```

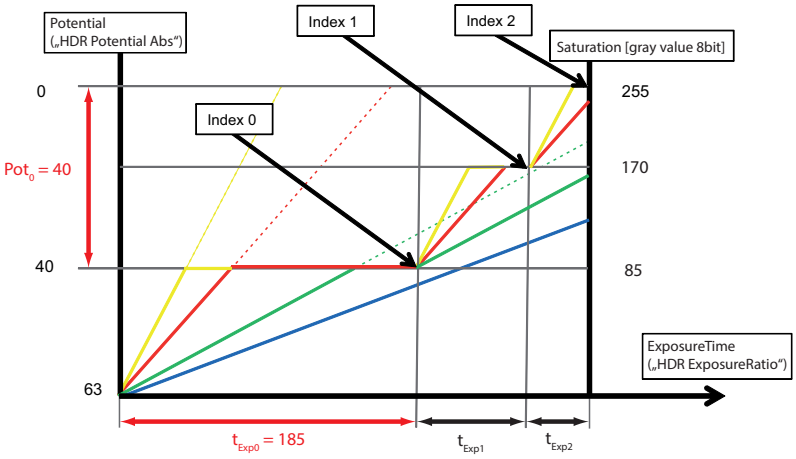
```
ExposureTime                : 10000

HDR parameter change
  HDREnable                 : 1
  HDRIndex                  : 0
  HDRExposureRatio          : 185
  HDRExposureRatio
  Percent                   : 72.54
  HDRPotentialAbs           : 40

HDRIndex                    : 1
  HDRExposureRatio          : 45
  HDRExposureRatio
  Percent                   : 17.64
  HDRPotentialAbs           : 20
```

Console Output (C++)

2.1.2 C#

In this chapter, the setting of *Normal HDR* in C# is shown.

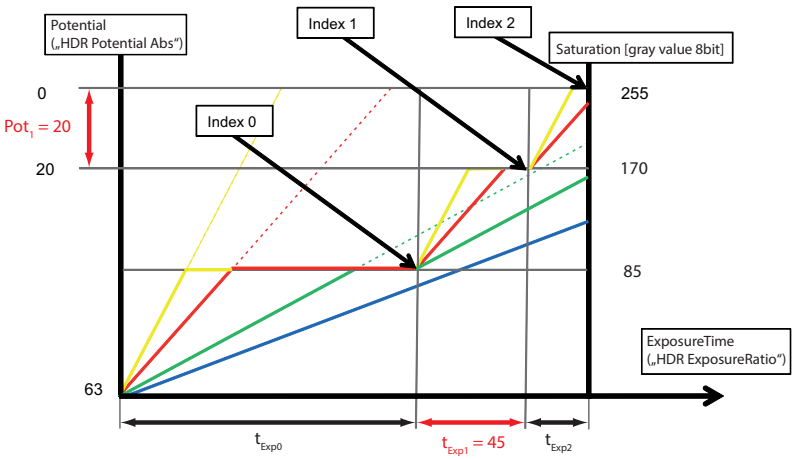
```
SystemList
Open a System
Get the InterfaceList and fill it
Open an Interface
Get the DeviceList and fill it
Open a Device


//Device Parameter Setup

mDevice.RemoteNodeList["TriggerMode"].Value = "Off";

mDevice.RemoteNodeList["ExposureTime"].Value = 10000.0;

System.Console.Write("ExposureTime : {0}\n",
   (double)mDevice.RemoteNodeList["ExposureTime"].Value);

System.Console.Write("\n");


//HDR parameter change

System.Console.Write("HDR parameter change\n");

mDevice.RemoteNodeList["HDREnable"].Value = true;

System.Console.Write("  HDREnable : {0}\n",
   (bool)mDevice.RemoteNodeList["HDREnable"].Value);


//only HXG

mDevice.RemoteNodeList["HDREnableTriggerAutoMode"].Value = false;

System.Console.Write("HDREnableTriggerAutoMode : {0}\n",
   (bool)mDevice.RemoteNodeList["HDREnableTriggerAutoMode"].Value);


mDevice.RemoteNodeList["HDRIndex"].Value = (long)0;

System.Console.Write("HDRIndex : {0}\n",
   (long)mDevice.RemoteNodeList["HDRIndex"].Value);

mDevice.RemoteNodeList["HDRExposureRatio"].Value = (long)185; //t_Exp_0

System.Console.Write("HDRExposureRatio : {0}\n",
   (long)mDevice.RemoteNodeList["HDRExposureRatio"].Value);

System.Console.Write("HDRExposureRatioPercent : {0}\n",
   (double)mDevice.RemoteNodeList["HDRExposureRatioPercent"].Value);

mDevice.RemoteNodeList["HDRExposureRatio"].Value = (long)40; //Pot_0
System.Console.Write("HDRPotentialAbs : {0}\n",
   (long)mDevice.RemoteNodeList["HDRPotentialAbs"].Value);
```




```
mDevice.RemoteNodeList["HDRIndex"].Value = (long)1;

System.Console.Write("HDRIndex : {0}\n",
   (long)mDeviceRemoteNodeList["HDRIndex"].Value);

mDevice.RemoteNodeList["HDRExposureRatio"].Value = (long)45; //t_Exp_1

System.Console.Write("HDRExposureRatio : {0}\n",
   (long)mDevice.RemoteNodeList["HDRExposureRatio"].Value);

System.Console.Write("HDRExposureRatioPercent : {0}\n",
   (double)mDevice.RemoteNodeList["HDRExposureRatioPercent"].Value);

mDevice.RemoteNodeList["HDRPotentialAbs"].Value = (long)20; //Pot_1

System.Console.Write("HDRPotentialAbs : {0}\n",
   (long)mDevice.RemoteNodeList["HDRPotentialAbs"].Value);
System.Console.Write("\n");
```

Notice

The value for t_{Exp2} will be calculated automatically in the camera.
($t_{Exp2} = t_{exposure} - t_{Exp0} - t_{Exp1}$)
Pot_2 is always 0.



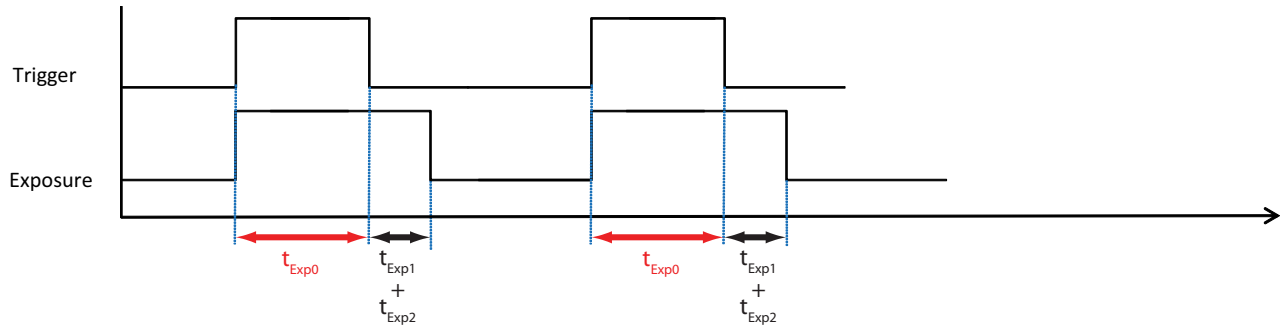
```
Get DataStreamList and fill it
Open a Data Stream
Create the BufferList and allocate Buffer memory
Allocate Image Buffer to the DataStream
Start Camera and fill the Image Buffer
Releasing the resources
```

```
ExposureTime                : 10000

HDR parameter change
  HDREnable                 : True
  HDRIndex                  : 0
  HDRExposureRatio          : 185
  HDRExposureRatio
  Percent                   : 72.54
  HDRPotentialAbs           : 40

HDRIndex                    : 1
  HDRExposureRatio          : 45
  HDRExposureRatio
  Percent                   : 17.64
  HDRPotentialAbs           : 20
```

Console Output (C#)

2.2 Trigger controlled HDR

2.2.1 HDREnableTriggerAutoMode = True & ExposureMode = TriggerWidth



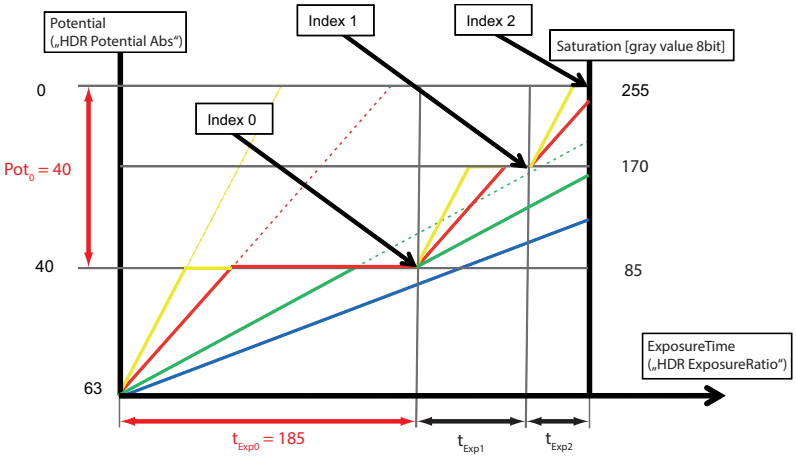
The exposure time for the first exposure slot (t_{Exp0}) is controlled by trigger and the other two exposure time sections (t_{Exp0}, t_{Exp1}) are calculated automatically according to HDR settings.

2.2.1.1 C++

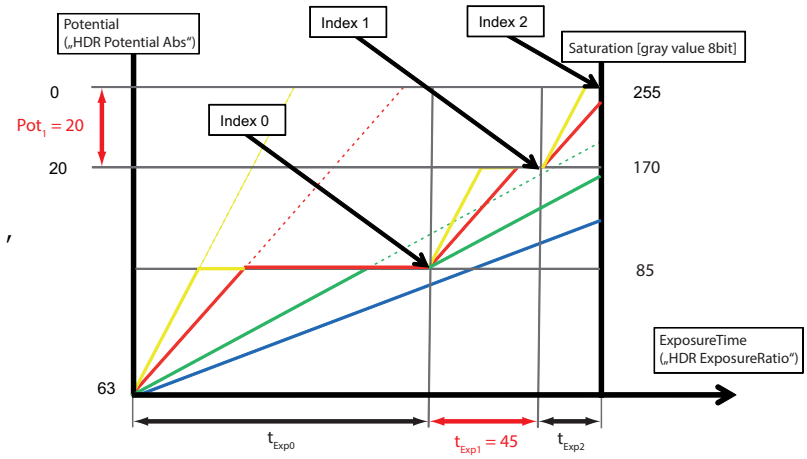
In this chapter, the setting of *HDREnableTriggerAutoMode = True & ExposureMode = TriggerWidth* in C ++ is shown.

```
SystemList
Open a System
Get the InterfaceList and fill it
Open an Interface
Get the DeviceList and fill it
Open a Device


//Device Parameter Setup
if(pDevice->GetRemoteNode("ExposureMode")->GetValue() == "Timed")
{
    pDevice->GetRemoteNode("TriggerMode")->SetString("On");
}
std::cout << "TriggerMode: "
          << pDevice->GetRemoteNode("TriggerMode")->GetValue()
          << std::endl;


pDevice->GetRemoteNode("TriggerSource")->SetString("Line0");
std::cout << "TriggerSource: "
          << pDevice->GetRemoteNode("TriggerSource")->GetValue()
          << std::endl;
```

```
pDevice->GetRemoteNode("TriggerActivation")->SetString("RisingEdge");

std::cout << "TriggerActivation: "
          << pDevice->GetRemoteNode("TriggerActivation")->GetValue()
          << std::endl;


pDevice->GetRemoteNode("ExposureMode")->SetString("TriggerWidth");

std::cout << "ExposureMode:"
          << pDevice->GetRemoteNode("ExposureMode")->GetValue()
          << std::endl;


//HDR parameter change

std::cout << "HDR parameter change" << std::endl;

pDevice->GetRemoteNode("HDREnable")->SetBool(true);

std::cout << "HDREnable:"
          << pDevice->GetRemoteNode("HDREnable")->GetBool()
          << std::endl;


//only HXG

pDevice->GetRemoteNode("HDREnableTriggerAutoMode")->SetBool(true);

std::cout << "HDREnableTriggerAutoMode:"
          << pDevice->GetRemoteNode("HDREnableTriggerAutoMode")->GetBool()
          << std::endl;


pDevice->GetRemoteNode("HDRIndex")->SetInt(0);

std::cout << "HDRIndex:"
          << pDevice->GetRemoteNode("HDRIndex")->GetInt()
          << std::endl;

pDevice->GetRemoteNode("HDRExposureRatio")->SetInt(185); //t_Exp_0

std::cout << "HDRExposureRatio: "
          << pDevice->GetRemoteNode("HDRExposureRatio")->GetInt()
          << std::endl;

std::cout << "HDRExposureRatioPercent: "
          << pDevice->GetRemoteNode("HDRExposureRatioPercent")->GetDouble()
          << std::endl;

pDevice->GetRemoteNode("HDRPotentialAbs")->SetInt(40); //Pot_0

std::cout << "HDRPotentialAbs:"
          << pDevice->GetRemoteNode("HDRPotentialAbs")->GetInt()
          << std::endl;
```




```
pDevice->GetRemoteNode("HDRIndex")->SetInt(1);

std::cout << "HDRIndex:"
          << pDevice->GetRemoteNode("HDRIndex")->GetInt()
          << std::endl;

pDevice->GetRemoteNode("HDRExposureRatio")->SetInt(45); //t_Exp_1

std::cout << "HDRExposureRatio:"
          << pDevice->GetRemoteNode("HDRExposureRatio")->GetInt()
          << std::endl;

std::cout << "HDRExposureRatioPercent:"
          << pDevice->GetRemoteNode("HDRExposureRatioPercent")->GetDouble()
          << std::endl;

pDevice->GetRemoteNode("HDRPotentialAbs")->SetInt(20); //Pot_1

std::cout << "HDRPotentialAbs:"
          << pDevice->GetRemoteNode("HDRPotentialAbs")->GetInt()
          << std::endl;
```



,

```
Get DataStreamList and fill it
Open a Data Stream
Create the BufferList and allocate Buffer memory
Allocate Image Buffer to the DataStream
Start Camera and fill the Image Buffer
Releasing the resources
```

```
TriggerMode: On

TriggerSource: Line0

TriggerActivation: RisingEdge

ExposureMode: TriggerWidth

HDR parameter change
HDREnable: True
HDREnableTriggerAutoMode: True
HDRIndex: 0
HDRExposureRatio: 185
HDRExposureRatioPercent: 72.83
HDRPotentialAbs: 40

HDRIndex: 1
HDRExposureRatio: 45
HDRExposureRatioPercent: 17.72
HDRPotentialAbs: 20
```

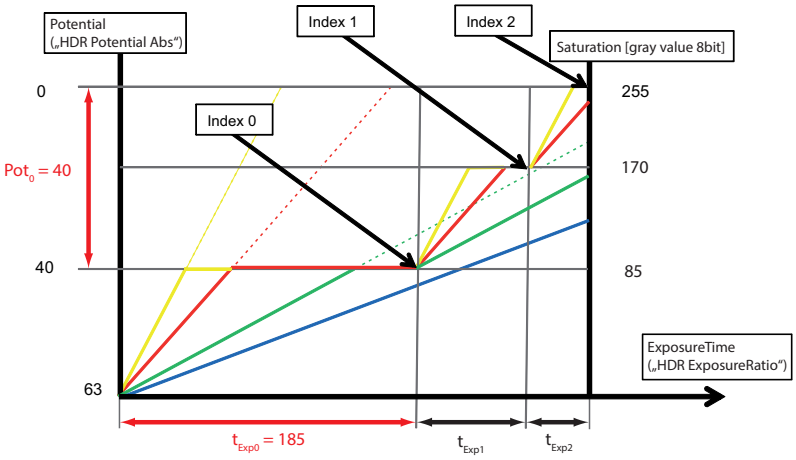
Console Output (C++)

2.2.1.2 C#

In this chapter, the setting of *HDREnableTriggerAutoMode = True & ExposureMode = TriggerWidth* in C# is shown.

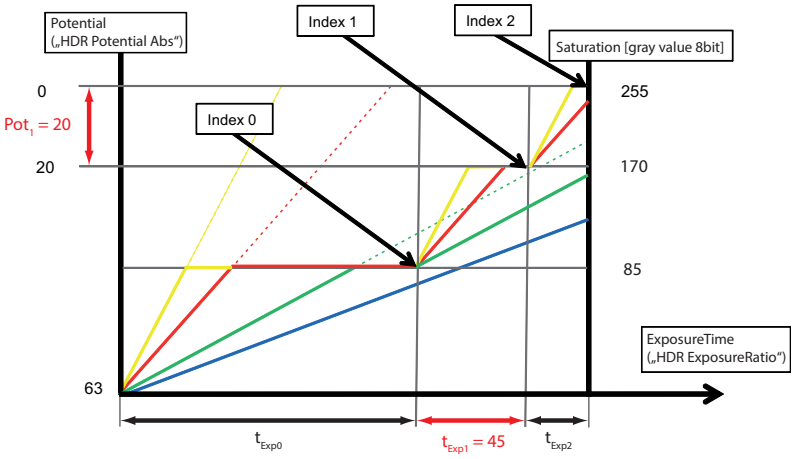
```
SystemList
Open a System
Get the InterfaceList and fill it
Open an Interface
Get the DeviceList and fill it
Open a Device


//Device Parameter Setup
if ((string)mDevice.RemoteNodeList["ExposureMode"].Value == "Timed")
{
    mDevice.RemoteNodeList["TriggerMode"].Value = "On";
}
System.Console.Write("TriggerMode: {0}\n\n",
   (string)mDevice.RemoteNodeList["TriggerMode"].Value);


mDevice.RemoteNodeList["TriggerSource"].Value = "Line0";
System.Console.Write("TriggerSource : {0}\n\n",
   (string)mDevice.RemoteNodeList["TriggerSource"].Value);


mDevice.RemoteNodeList["TriggerActivation"].Value = "RisingEdge";
System.Console.Write("  TriggerActivation: {0}\n\n",
   (string)mDevice.RemoteNodeList["TriggerActivation"].Value);


mDevice.RemoteNodeList["ExposureMode"].Value = "TriggerWidth";
System.Console.Write("ExposureMode : {0}\n\n",
   (string)mDevice.RemoteNodeList["ExposureMode"].Value);


//HDR parameter change
System.Console.Write("HDR parameter change\n");
mDevice.RemoteNodeList["HDREnable"].Value = true;
System.Console.Write(" HDREnable : {0}\n",
   (bool)mDevice.RemoteNodeList["HDREnable"].Value);
```

```
//only HXG

mDevice.RemoteNodeList["HDREnableTriggerAutoMode"].Value = true;

System.Console.Write("  HDREnableTriggerAutoMode  : {0}\n",
   (bool)mDevice.RemoteNodeList["HDREnableTriggerAutoMode"].Value);


mDevice.RemoteNodeList["HDRIndex"].Value = (long)0;

System.Console.Write("  HDRIndex : {0}\n",
   (long)mDevice.RemoteNodeList["HDRIndex"].Value);


mDevice.RemoteNodeList["HDRExposureRatio"].Value = (long)185; //t_Exp_0

System.Console.Write(" HDRExposureRatio: {0}\n",
   (long)mDevice.RemoteNodeList["HDRExposureRatio"].Value);

System.Console.Write("HDRExposureRatioPercent: {0:F2}\n",
   (double)mDevice.RemoteNodeList["HDRExposureRatioPercent"].Value);

mDevice.RemoteNodeList["HDRPotentialAbs"].Value = (long)40; //Pot_0

System.Console.Write("HDRPotentialAbs : {0}\n",
   (long)mDevice.RemoteNodeList["HDRPotentialAbs"].Value);
```




```
mDevice.RemoteNodeList["HDRIndex"].Value = (long)1;

System.Console.Write("HDRIndex: {0}\n",
   (long)mDevice.RemoteNodeList["HDRIndex"].Value);

mDevice.RemoteNodeList["HDRExposureRatio"].Value = (long)45; //Pot_1

System.Console.Write("HDRExposureRatio : {0}\n",
   (long)mDevice.RemoteNodeList["HDRExposureRatio"].Value);

System.Console.Write("HDRExposureRatioPercent: {0:F2}\n",
   (double)mDevice.RemoteNodeList["HDRExposureRatioPercent"].Value);

mDevice.RemoteNodeList["HDRPotentialAbs"].Value = (long)20; //t_Exp_1
```

```
System.Console.Write("HDRPotentialAbs: {0}\n",
   (long)mDevice.RemoteNodeList["HDRPotentialAbs"].Value);

System.Console.Write("\n");
```

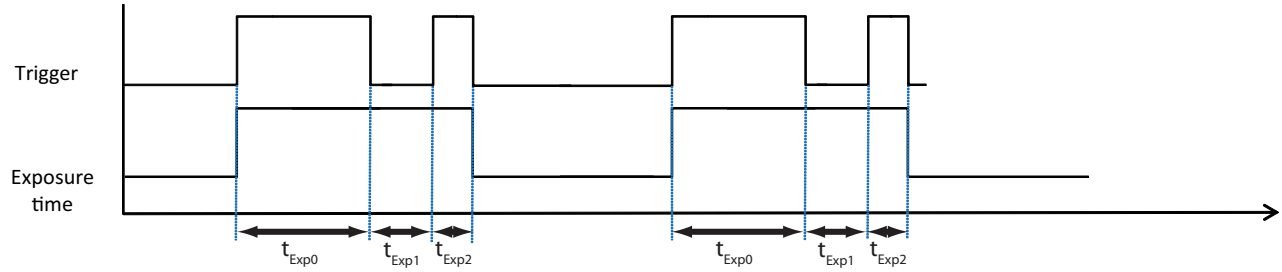



```
Get DataStreamList and fill it
Open a Data Stream
Create the BufferList and allocate Buffer memory
Allocate Image Buffer to the DataStream
Start Camera and fill the Image Buffer
Releasing the resources
```

```
TriggerMode: On

TriggerSource: Line0
TriggerActivation: RisingEdge
ExposureMode: TriggerWidth
HDR parameter change
HDREnable: True
HDREnableTriggerAutoMode:
True

HDRIndex: 0
HDRExposureRatio: 185
HDRExposureRatioPercent:
72.83
HDRPotentialAbs: 40
HDRIndex: 1
HDRExposureRatio: 45
HDRExposureRatioPercent:
17,72
HDRPotentialAbs: 20
```

Console Output (C#)

2.2.2 EnableTriggerAutoMode = False & ExposureMode = TriggerWidth



The exposure times for all three exposure slots are controlled by trigger.

2.2.2.1 C++

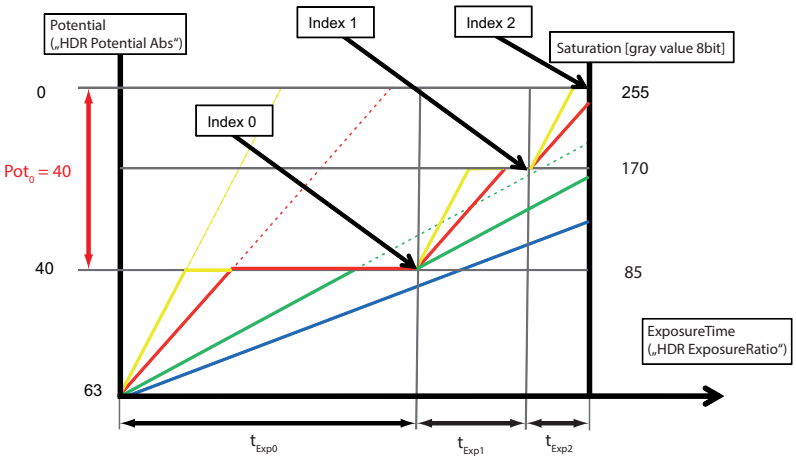
In this chapter, the setting of *EnableTriggerAutoMode = False & ExposureMode = TriggerWidth* in C++ is shown.

```
SystemList
Open a System
Get the InterfaceList and fill it
Open an Interface
Get the DeviceList and fill it
Open a Device


//Device Parameter Setup
if(pDevice->GetRemoteNode("ExposureMode")->GetValue() == "Timed")
{
    pDevice->GetRemoteNode("TriggerMode")->SetString("On");
}
std::cout << "TriggerMode: "
          << pDevice->GetRemoteNode("TriggerMode")->GetValue()
          << std::endl;


pDevice->GetRemoteNode("TriggerSource")->SetString("Line0");
std::cout << "TriggerSource: "
          << pDevice->GetRemoteNode("TriggerSource")->GetValue()
          << std::endl;


pDevice->GetRemoteNode("TriggerActivation")->SetString("RisingEdge");
```

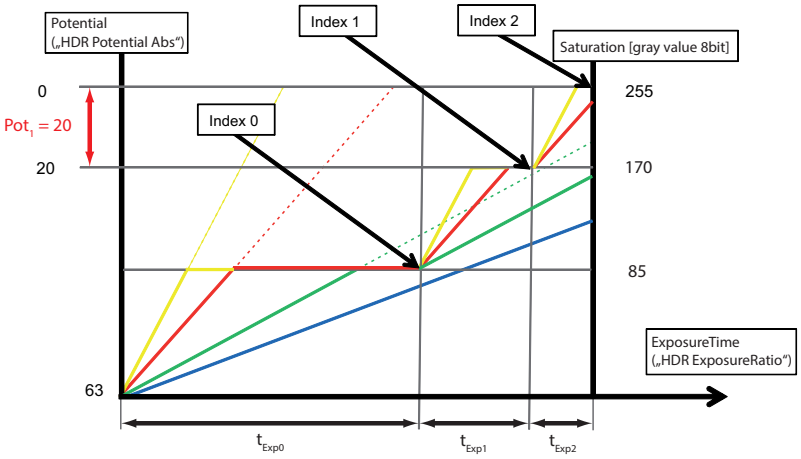
```
std::cout << "  TriggerActivation: "
          << pDevice->GetRemoteNode("TriggerActivation")->GetValue()
          << std::endl;


pDevice->GetRemoteNode("ExposureMode")->SetString("TriggerWidth");

std::cout << "ExposureMode: "
          << pDevice->GetRemoteNode("ExposureMode")->GetValue()
          << std::endl;


//HDR parameter change

std::cout << "HDR parameter change" << std::endl;

pDevice->GetRemoteNode("HDREnable")->SetBool(true);

std::cout << "HDREnable : "
          << pDevice->GetRemoteNode("HDREnable")->GetBool()
          << std::endl;


//only HXG

pDevice->GetRemoteNode("HDREnableTriggerAutoMode")->SetBool(false);

std::cout << "HDREnableTriggerAutoMode: "
          << pDevice->GetRemoteNode("HDREnableTriggerAutoMode")->GetBool()
          << std::endl;


pDevice->GetRemoteNode("HDRIndex")->SetInt(0);

std::cout << "HDRIndex: "
          << pDevice->GetRemoteNode("HDRIndex")->GetInt()
          << std::endl;


//pDevice->GetRemoteNode("HDRExposureRatio")->SetInt(185); //t_expo_0

//std::cout << "HDRExposureRatio: "
            << pDevice->GetRemoteNode("HDRExposureRatio")->GetInt()
            << std::endl;

//std::cout << "HDRExposureRatioPercent :"
            << pDevice->GetRemoteNode("HDRExposureRatioPercent")->GetDouble()
            << std::endl;

pDevice->GetRemoteNode("HDRPotentialAbs")->SetInt(40); //Pot_0

std::cout << "HDRPotentialAbs: "
          << pDevice->GetRemoteNode("HDRPotentialAbs")->GetInt()
          << std::endl;
```

Notice

If `HDREnableTriggerAutoMode = False` then `HDRExposureRatio` and `HDRExposureRatioPercent` is not effected.




```
pDevice->GetRemoteNode("HDRIndex")->SetInt(1);

std::cout << "HDRIndex: "
          << pDevice->GetRemoteNode("HDRIndex")->GetInt()
          << std::endl;


//pDevice->GetRemoteNode("HDRExposureRatio")->SetInt(45); //t_expo_1

//std::cout << "HDRExposureRatio: "
            << pDevice->GetRemoteNode("HDRExposureRatio")->GetInt()
            << std::endl;

//std::cout << "HDRExposureRatioPercent: "
            << pDevice->GetRemoteNode("HDRExposureRatioPercent")->GetDouble()
            << std::endl;

pDevice->GetRemoteNode("HDRPotentialAbs")->SetInt(20); //Pot_1

std::cout << "HDRPotentialAbs: "
          << pDevice->GetRemoteNode("HDRPotentialAbs")->GetInt()
          << std::endl;
```




```
Get DataStreamList and fill it
Open a Data Stream
Create the BufferList and allocate Buffer memory
Allocate Image Buffer to the DataStream
Start Camera and fill the Image Buffer
Releasing the resources
```

```
TriggerMode: On

TriggerSource: Line0
TriggerActivation: RisingEdge
ExposureMode: TriggerWidth
HDR parameter change
HDREnable: True

HDREnableTriggerAutoMode:
False
HDRIndex: 0
HDRPotentialAbs: 40
HDRIndex: 1
HDRPotentialAbs: 20
```

Console Output (C++)

2.2.2.2 C#

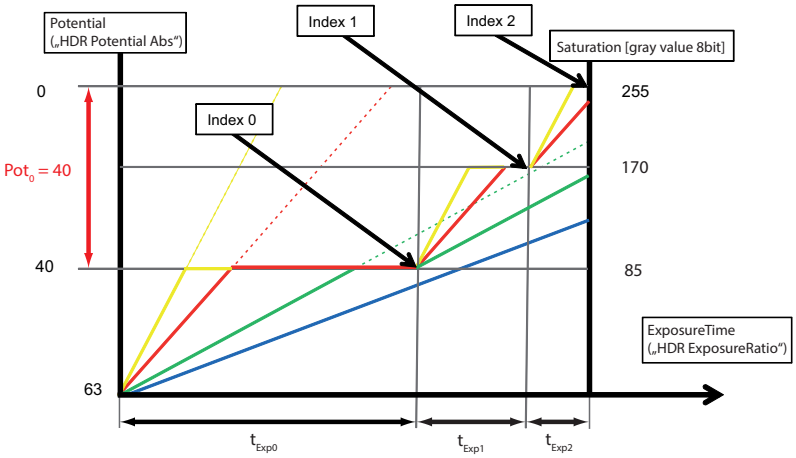
In this chapter, the setting of *EnableTriggerAutoMode* = *False & ExposureMode* = *TriggerWidth* in C# is shown.

```
SystemList
Open a System
Get the InterfaceList and fill it
Open an Interface
Get the DeviceList and fill it
Open a Device


//Device Parameter Setup
if ((string)mDevice.RemoteNodeList["ExposureMode"].Value == "Timed")
{
    mDevice.RemoteNodeList["TriggerMode"].Value = "On";
}
System.Console.Write("TriggerMode: {0}\n\n",
   (string)mDevice.RemoteNodeList["TriggerMode"].Value);


mDevice.RemoteNodeList["TriggerSource"].Value = "Line0";
System.Console.Write("TriggerSource: {0}\n\n",
   (string)mDevice.RemoteNodeList["TriggerSource"].Value);


mDevice.RemoteNodeList["TriggerActivation"].Value = "RisingEdge";
System.Console.Write("TriggerActivation: {0}\n\n",
   (string)mDevice.RemoteNodeList["TriggerActivation"].Value);


mDevice.RemoteNodeList["ExposureMode"].Value = "TriggerWidth";
System.Console.Write("ExposureMode: {0}\n\n",
   (string)mDevice.RemoteNodeList["ExposureMode"].Value);


//HDR parameter change
System.Console.Write("HDR parameter change\n");
mDevice.RemoteNodeList["HDREnable"].Value = true;
System.Console.Write("HDREnable: {0}\n",
   (bool)mDevice.RemoteNodeList["HDREnable"].Value);


//only HXG
```

```
mDevice.RemoteNodeList["HDREnableTriggerAutoMode"].Value = false;

System.Console.Write("HDREnableTriggerAutoMode : {0}\n",
   (bool)mDevice.RemoteNodeList["HDREnableTriggerAutoMode"].Value);


mDevice.RemoteNodeList["HDRIndex"].Value = (long)0;

System.Console.Write("HDRIndex: {0}\n",
   (long)mDevice.RemoteNodeList["HDRIndex"].Value);


//mDevice.RemoteNodeList["HDRExposureRatio"].Value = (long)185;

//System.Console.Write("HDRExposureRatio: {0}\n",
    (long)mDevice.RemoteNodeList["HDRExposureRatio"].Value);

//System.Console.Write("HDRExposureRatioPercent: {0:F2}\n",
    (double)mDevice.RemoteNodeList["HDRExposureRatioPercent"].Value);


mDevice.RemoteNodeList["HDRPotentialAbs"].Value = (long)40; //Pot_0

System.Console.Write("HDRPotentialAbs: {0}\n",
   (long)mDevice.RemoteNodeList["HDRPotentialAbs"].Value);
```

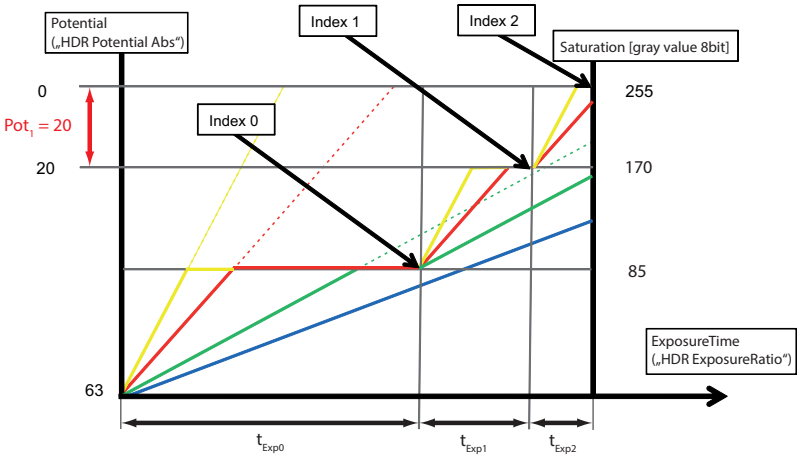
Notice

If `HDREnableTriggerAutoMode = False` then `HDRExposureRatio` and `HDRExposureRatioPercent` is not effected.




```
mDevice.RemoteNodeList["HDRIndex"].Value = (long)1;

System.Console.Write("HDRIndex: {0}\n",
   (long)mDevice.RemoteNodeList["HDRIndex"].Value);
```

```
//mDevice.RemoteNodeList["HDRExposureRatio"].Value = (long)45;

//System.Console.Write("HDRExposureRatio: {0}\n",
    (long)mDevice.RemoteNodeList["HDRExposureRatio"].Value);

//System.Console.Write("HDRExposureRatioPercent: {0:F2}\n",
    (double)mDevice.RemoteNodeList["HDRExposureRatioPercent"].Value);


mDevice.RemoteNodeList["HDRPotentialAbs"].Value = (long)20; //Pot_1

System.Console.Write("HDRPotentialAbs: {0}\n",
   (long)mDevice.RemoteNodeList["HDRPotentialAbs"].Value);

System.Console.Write("\n");
```

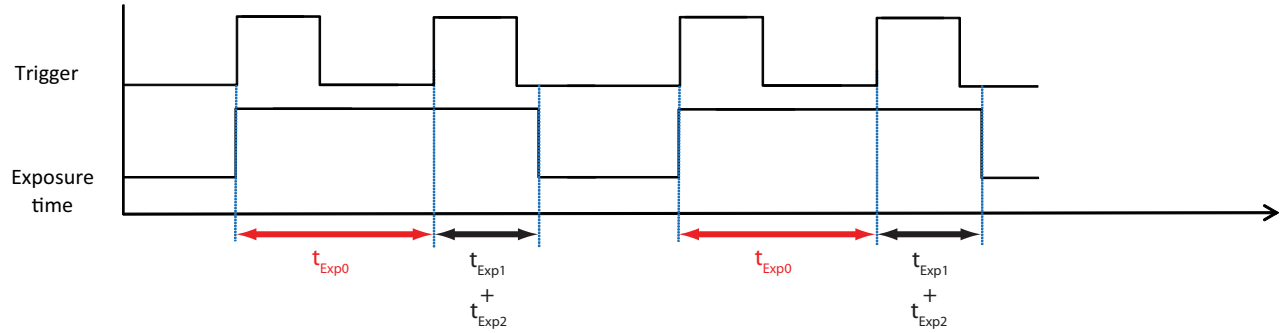



```
Get DataStreamList and fill it
Open a Data Stream
Create the BufferList and allocate Buffer memory
Allocate Image Buffer to the DataStream
Start Camera and fill the Image Buffer
Releasing the resources
```

```
TriggerMode: On

TriggerSource: Line0

TriggerActivation: RisingEdge

ExposureMode: TriggerWidth

HDR parameter change
HDREnable: True
HDREnableTriggerAutoMode: False
HDRIndex: 0
HDRPotentialAbs: 40
HDRIndex: 1
HDRPotentialAbs: 20
```

Console Output (C#)

2.2.3 HDREnableTriggerAutoMode = True & ExposureMode = TriggerControlled



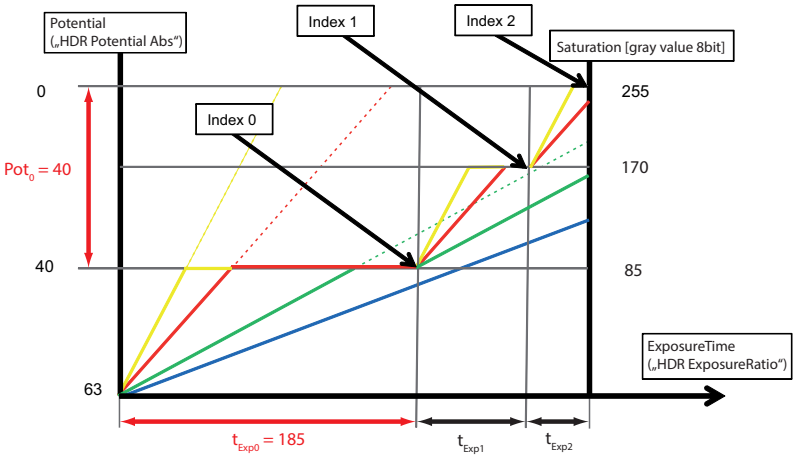
The ExposureTime for the first exposure slot (t_{Exp0}) is controlled by trigger and the other two exposure time sections are calculated automatically according to HDR settings.

2.2.3.1 C++

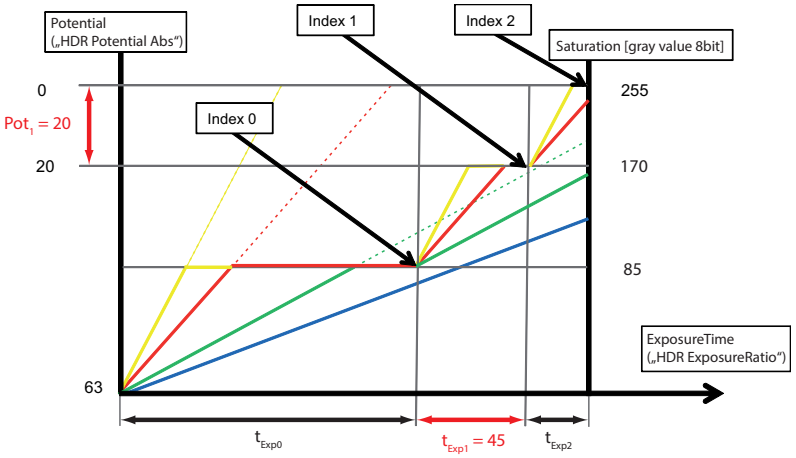
In this chapter, the setting of *HDREnableTriggerAutoMode = True & ExposureMode = TriggerControlled* in C++ is shown.

```
SystemList
Open a System
Get the InterfaceList and fill it
Open an Interface
Get the DeviceList and fill it
Open a Device


//Device Parameter Setup
if(pDevice->GetRemoteNode("ExposureMode")->GetValue() == "Timed")
{
    pDevice->GetRemoteNode("TriggerMode")->SetString("On");
}
std::cout << "TriggerMode: "
          << pDevice->GetRemoteNode("TriggerMode")->GetValue()
          << std::endl;


pDevice->GetRemoteNode("TriggerSource")->SetString("Line0");
std::cout << "TriggerSource: "
          << pDevice->GetRemoteNode("TriggerSource")->GetValue()
          << std::endl;


pDevice->GetRemoteNode("TriggerActivation")->SetString("RisingEdge");
```

```
std::cout << "TriggerActivation: "
          << pDevice->GetRemoteNode("TriggerActivation")->GetValue()
          << std::endl;

pDevice->GetRemoteNode("ExposureMode")->SetString("TriggerControlled");

std::cout << "ExposureMode: "
          << pDevice->GetRemoteNode("ExposureMode")->GetValue()
          << std::endl;


//HDR parameter change

std::cout << "HDR parameter change" << std::endl;

pDevice->GetRemoteNode("HDREnable")->SetBool(true);

std::cout << "HDREnable: "
          << pDevice->GetRemoteNode("HDREnable")->GetBool()
          << std::endl;


//only HXG

pDevice->GetRemoteNode("HDREnableTriggerAutoMode")->SetBool(true);

std::cout << "HDREnableTriggerAutoMode: "
          << pDevice->GetRemoteNode("HDREnableTriggerAutoMode")->GetBool()
          << std::endl;


pDevice->GetRemoteNode("HDRIndex")->SetInt(0);

std::cout << "HDRIndex: "
          << pDevice->GetRemoteNode("HDRIndex")->GetInt()
          << std::endl;

pDevice->GetRemoteNode("HDRExposureRatio")->SetInt(185); //t_Exp_0

std::cout << "HDRExposureRatio: "
          << pDevice->GetRemoteNode("HDRExposureRatio")->GetInt()
          << std::endl;

std::cout << "HDRExposureRatioPercent: "
          << pDevice->GetRemoteNode("HDRExposureRatioPercent")->GetDouble()
          << std::endl;

pDevice->GetRemoteNode("HDRPotentialAbs")->SetInt(40); //Pot_0

std::cout << "HDRPotentialAbs :"
          << pDevice->GetRemoteNode("HDRPotentialAbs")->GetInt()
          << std::endl;
```




```
pDevice->GetRemoteNode("HDRIndex")->SetInt(1);

std::cout << "HDRIndex: "
          << pDevice->GetRemoteNode("HDRIndex")->GetInt()
          << std::endl;

pDevice->GetRemoteNode("HDRExposureRatio")->SetInt(45); //t_expo_1

std::cout << "HDRExposureRatio: "
          << pDevice->GetRemoteNode("HDRExposureRatio")->GetInt()
          << std::endl;

std::cout << "HDRExposureRatioPercent: "
          << pDevice->GetRemoteNode("HDRExposureRatioPercent")->GetDouble()
          << std::endl;

pDevice->GetRemoteNode("HDRPotentialAbs")->SetInt(20); //Pot_1

std::cout << "HDRPotentialAbs: "
          << pDevice->GetRemoteNode("HDRPotentialAbs")->GetInt()
          << std::endl;
```



```
Get DataStreamList and fill it
Open a Data Stream
Create the BufferList an allocate Buffer memory
Allocate Image Buffer to the DataStream
Start Camera and fill the Image Buffer
Releasing the resources
```

```
TriggerMode: On

TriggerSource: Line0

TriggerActivation: RisingEdge

ExposureMode: TriggerControlled

HDR parameter change
HDREnable: True
HDREnableTriggerAutoMode: True
HDRIndex: 0
HDRExposureRatio: 185
HDRExposureRatioPercent: 72.83
HDRPotentialAbs: 40
HDRIndex: 1
HDRExposureRatio: 45
HDRExposureRatioPercent: 17.72
HDRPotentialAbs: 20
```

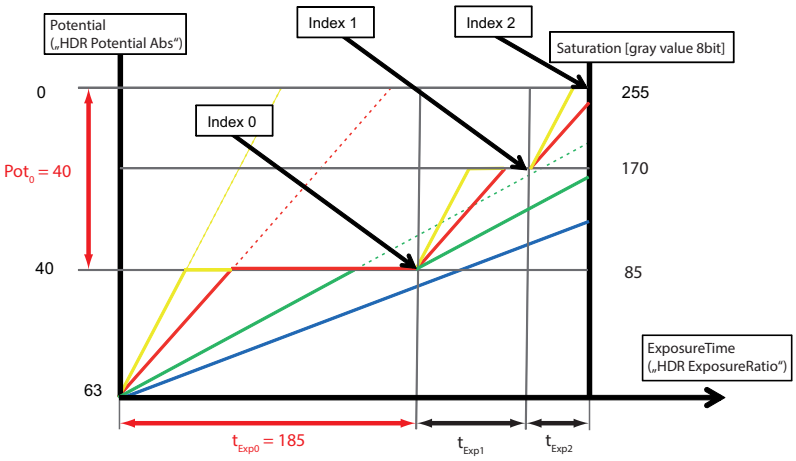
Console Output (C++)

2.2.3.2 C#

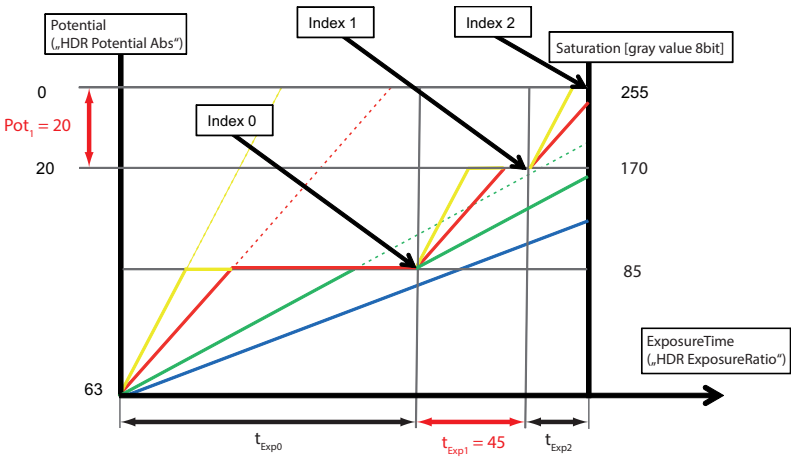
In this chapter, the setting of *HDREnableTriggerAutoMode = True & ExposureMode = TriggerControlled* in C# is shown.

```
SystemList
Open a System
Get the InterfaceList and fill it
Open an Interface
Get the DeviceList and fill it
Open a Device


//Device Parameter Setup
if ((string)mDevice.RemoteNodeList["ExposureMode"].Value == "Timed")
{
    mDevice.RemoteNodeList["TriggerMode"].Value = "On";
}
System.Console.Write("TriggerMode: {0}\n\n",
   (string)mDevice.RemoteNodeList["TriggerMode"].Value);


mDevice.RemoteNodeList["TriggerSource"].Value = "Line0";
System.Console.Write("TriggerSource: {0}\n\n",
   (string)mDevice.RemoteNodeList["TriggerSource"].Value);


mDevice.RemoteNodeList["TriggerActivation"].Value = "RisingEdge";
System.Console.Write("TriggerActivation: {0}\n\n",
   (string)mDevice.RemoteNodeList["TriggerActivation"].Value);


mDevice.RemoteNodeList["ExposureMode"].Value = "TriggerControlled";
System.Console.Write("ExposureMode: {0}\n\n",
   (string)mDevice.RemoteNodeList["ExposureMode"].Value);


//HDR parameter change
System.Console.Write("HDR parameter change\n");
mDevice.RemoteNodeList["HDREnable"].Value = true;
System.Console.Write("HDREnable: {0}\n",
   (bool)mDevice.RemoteNodeList["HDREnable"].Value);
```

```
//only HXG

mDevice.RemoteNodeList["HDREnableTriggerAutoMode"].Value = true;

System.Console.Write("  HDREnableTriggerAutoMode  : {0}\n",
   (bool)mDevice.RemoteNodeList["HDREnableTriggerAutoMode"].Value);

mDevice.RemoteNodeList["HDRIndex"].Value = (long)0;

System.Console.Write("HDRIndex: {0}\n",
   (long)mDevice.RemoteNodeList["HDRIndex"].Value);

mDevice.RemoteNodeList["HDRExposureRatio"].Value = (long)185; //t_Exp_0

System.Console.Write("HDRExposureRatio: {0}\n",
   (long)mDevice.RemoteNodeList["HDRExposureRatio"].Value);

System.Console.Write("HDRExposureRatioPercent: {0:F2}\n",
   (double)mDevice.RemoteNodeList["HDRExposureRatioPercent"].Value);

mDevice.RemoteNodeList["HDRPotentialAbs"].Value = (long)40; //Pot_0

System.Console.Write("HDRPotentialAbs: {0}\n",
   (long)mDevice.RemoteNodeList["HDRPotentialAbs"].Value);
```




```
mDevice.RemoteNodeList["HDRIndex"].Value = (long)1;

System.Console.Write("HDRIndex: {0}\n",
   (long)mDevice.RemoteNodeList["HDRIndex"].Value);

mDevice.RemoteNodeList["HDRExposureRatio"].Value = (long)45; //t_Exp_1

System.Console.Write("HDRExposureRatio: {0}\n",
   (long)mDevice.RemoteNodeList["HDRExposureRatio"].Value);

System.Console.Write("HDRExposureRatioPercent: {0:F2}\n",
   (double)mDevice.RemoteNodeList["HDRExposureRatioPercent"].Value);

mDevice.RemoteNodeList["HDRPotentialAbs"].Value = (long)20; //Pot_1

System.Console.Write("HDRPotentialAbs: {0}\n",
   (long)mDevice.RemoteNodeList["HDRPotentialAbs"].Value);

System.Console.Write("\n");
```

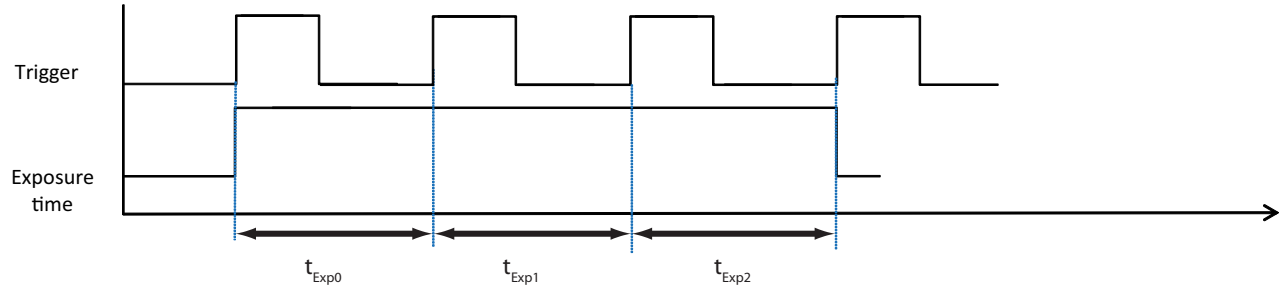



```
Get DataStreamList and fill it
Open a Data Stream
Create the BufferList an allocate Buffer memory
Allocate Image Buffer to the DataStream
Start Camera and fill the Image Buffer
Releasing the resources
```

```
TriggerMode: On

TriggerSource: Line0

TriggerActivation: RisingEdge

ExposureMode: TriggerControlled
HDR parameter change
HDREnable: True
HDREnableTriggerAutoMode: True
HDRIndex: 0
HDRExposureRatio: 185
HDRExposureRatioPercent: 72.83
HDRPotentialAbs: 40

HDRIndex: 1
HDRExposureRatio: 45
HDRExposureRatioPercent: 17.72
HDRPotentialAbs: 20
```

Console Output (C#)

2.2.4 HDREnableTriggerAutoMode = False & ExposureMode = TriggerControlled



The exposure times for all three exposure slots (t_{Exp0}, t_{Exp1}, t_{Exp2}) are controlled by trigger.

2.2.4.1 C++

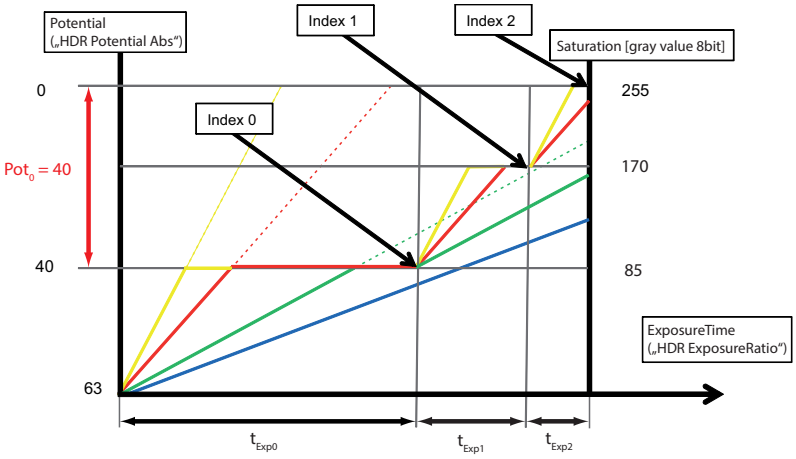
In this chapter, the setting of *HDREnableTriggerAutoMode = False & ExposureMode = TriggerControlled* in C++ is shown.

```
SystemList
Open a System
Get the InterfaceList and fill it
Open an Interface
Get the DeviceList and fill it
Open a Device


//Device Parameter Setup

if(pDevice->GetRemoteNode("ExposureMode")->GetValue() == "Timed")

{

    pDevice->GetRemoteNode("TriggerMode")->SetString("On");

}

std::cout << " TriggerMode:"
          << pDevice->GetRemoteNode("TriggerMode")->GetValue()
          << std::endl;


pDevice->GetRemoteNode("TriggerSource")->SetString("Line0");

std::cout << "TriggerSource:"
          << pDevice->GetRemoteNode("TriggerSource")->GetValue()
          << std::endl;


pDevice->GetRemoteNode("TriggerActivation")->SetString("RisingEdge");

std::cout << "TriggerActivation: "
          << pDevice->GetRemoteNode("TriggerActivation")->GetValue()
          << std::endl;
```

```
pDevice->GetRemoteNode("ExposureMode")->SetString("TriggerControlled");

std::cout << "ExposureMode:"
          << pDevice->GetRemoteNode("ExposureMode")->GetValue()
          << std::endl;


//HDR parameter change

std::cout << "HDR parameter change" << std::endl;

pDevice->GetRemoteNode("HDREnable")->SetBool(true);

std::cout << "HDREnable: "
          << pDevice->GetRemoteNode("HDREnable")->GetBool()
          << std::endl;


//only HXG

pDevice->GetRemoteNode("HDREnableTriggerAutoMode")->SetBool(false);

std::cout << "HDREnableTriggerAutoMode: "
          << pDevice->GetRemoteNode("HDREnableTriggerAutoMode")->GetBool()
          << std::endl;


pDevice->GetRemoteNode("HDRIndex")->SetInt(0);

std::cout << "HDRIndex: "
          << pDevice->GetRemoteNode("HDRIndex")->GetInt()
          << std::endl;


//pDevice->GetRemoteNode("HDRExposureRatio")->SetInt(185);

//std::cout << "HDRExposureRatio: "
            << pDevice->GetRemoteNode("HDRExposureRatio")->GetInt()
            << std::endl;

//std::cout << "HDRExposureRatioPercent: "
            << pDevice->GetRemoteNode("HDRExposureRatioPercent")->GetDouble()
            << std::endl;


pDevice->GetRemoteNode("HDRPotentialAbs")->SetInt(40); //Pot_0

std::cout << "HDRPotentialAbs: "
          << pDevice->GetRemoteNode("HDRPotentialAbs")->GetInt()
          << std::endl;
```

Notice

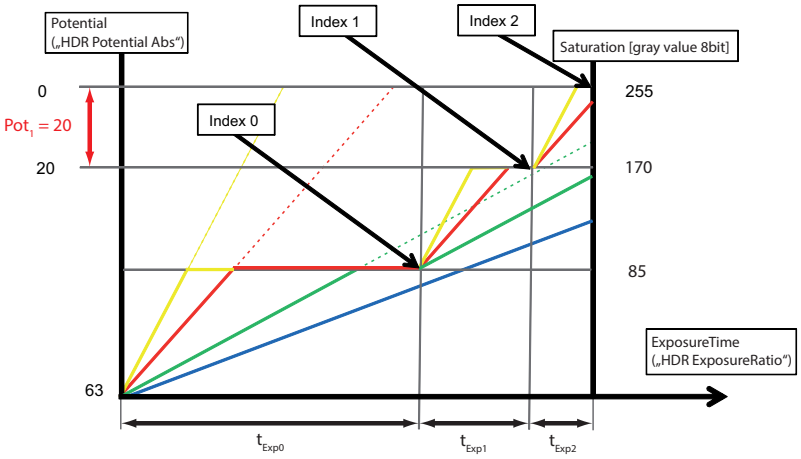
If `HDREnableTriggerAutoMode = False` then `HDRExposureRatio` and `HDRExposureRatioPercent` is not effected.




```
pDevice->GetRemoteNode("HDRIndex")->SetInt(1);

std::cout << "HDRIndex: "
          << pDevice->GetRemoteNode("HDRIndex")->GetInt()
          << std::endl;


//pDevice->GetRemoteNode("HDRExposureRatio")->SetInt(45); //texp1

//std::cout << "   HDRExposureRatio: "
            << pDevice->GetRemoteNode("HDRExposureRatio")->GetInt()
            << std::endl;

//std::cout << "HDRExposureRatioPercent: "
            << pDevice->GetRemoteNode("HDRExposureRatioPercent")->GetDouble()
            << std::endl;


pDevice->GetRemoteNode("HDRPotentialAbs")->SetInt(20); //Pot_1

std::cout << "HDRPotentialAbs: "
          << pDevice->GetRemoteNode("HDRPotentialAbs")->GetInt()
          << std::endl;
```

Notice

If `HDREnableTriggerAutoMode = False` then `HDRExposureRatio` and `HDRExposureRatioPercent` is not effected.




```
Get DataStreamList and fill it
Open a Data Stream
Create the BufferList an allocate Buffer memory
Allocate Image Buffer to the DataStream
Start Camera and fill the Image Buffer
Releasing the resources
```

```
TriggerMode: On

TriggerSource: Line0

TriggerActivation: RisingEdge

ExposureMode: TriggerControlled

HDR parameter change
HDREnable: True
HDREnableTriggerAutoMode: False
HDRIndex: 0
HDRPotentialAbs: 40
HDRIndex: 1
HDRPotentialAbs: 20
```

Console Output (C++)

2.2.4.2 C#

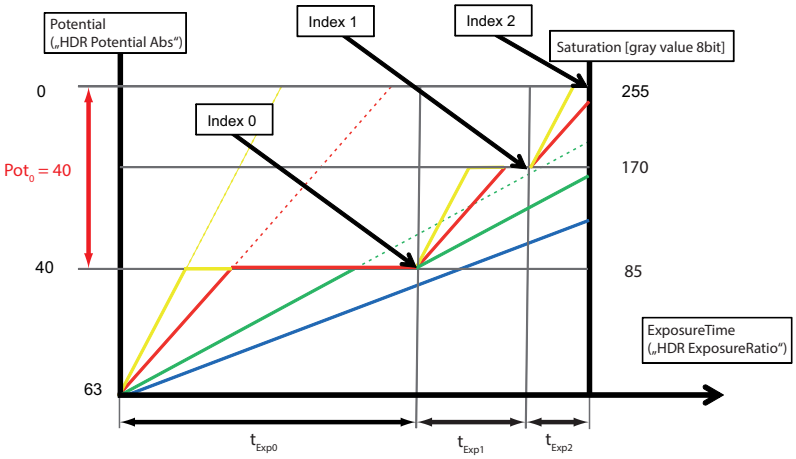
In this chapter, the setting of *HDREnableTriggerAutoMode = False & ExposureMode = TriggerControlled* in C# is shown.

```
SystemList
Open a System
Get the InterfaceList and fill it
Open an Interface
Get the DeviceList and fill it
Open a Device


//Device Parameter Setup
if ((string)mDevice.RemoteNodeList["ExposureMode"].Value == "Timed")
{
    mDevice.RemoteNodeList["TriggerMode"].Value = "On";
}
System.Console.Write("TriggerMode: {0}\n\n",
   (string)mDevice.RemoteNodeList["TriggerMode"].Value);


mDevice.RemoteNodeList["TriggerSource"].Value = "Line0";
System.Console.Write("TriggerSource: {0}\n\n",
   (string)mDevice.RemoteNodeList["TriggerSource"].Value);


mDevice.RemoteNodeList["TriggerActivation"].Value = "RisingEdge";
System.Console.Write("TriggerActivation: {0}\n\n",
   (string)mDevice.RemoteNodeList["TriggerActivation"].Value);


mDevice.RemoteNodeList["ExposureMode"].Value = "TriggerControlled";
System.Console.Write("ExposureMode: {0}\n\n",
 (string)mDevice.RemoteNodeList["ExposureMode"].Value);


//HDR parameter change
System.Console.Write("HDR parameter change\n");
mDevice.RemoteNodeList["HDREnable"].Value = true;
System.Console.Write("HDREnable: {0}\n",
   (bool)mDevice.RemoteNodeList["HDREnable"].Value);
```

```
//only HXG

mDevice.RemoteNodeList["HDREnableTriggerAutoMode"].Value = false;

System.Console.Write("HDREnableTriggerAutoMode: {0}\n",
   (bool)mDevice.RemoteNodeList["HDREnableTriggerAutoMode"].Value);

mDevice.RemoteNodeList["HDRIndex"].Value = (long)0;

System.Console.Write("  HDRIndex: {0}\n",
   (long)mDevice.RemoteNodeList["HDRIndex"].Value);


//mDevice.RemoteNodeList["HDRExposureRatio"].Value = (long)185;

//System.Console.Write("HDRExposureRatio: {0}\n",
   (long)mDevice.RemoteNodeList["HDRExposureRatio"].Value);

//System.Console.Write("HDRExposureRatioPercent: {0:F2}\n",
   (double)mDevice.RemoteNodeList["HDRExposureRatioPercent"].Value);


mDevice.RemoteNodeList["HDRPotentialAbs"].Value = (long)40; //Pot_0

System.Console.Write("HDRPotentialAbs: {0}\n",
   (long)mDevice.RemoteNodeList["HDRPotentialAbs"].Value);
```

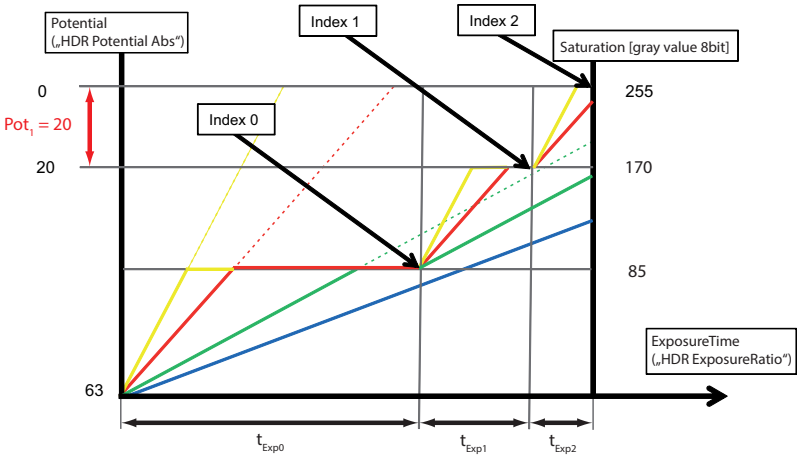
Notice

If `HDREnableTriggerAutoMode = False` then `HDRExposureRatio` and `HDRExposureRatioPercent` is not effected.




```
mDevice.RemoteNodeList["HDRIndex"].Value = (long)1;

System.Console.Write("HDRIndex: {0}\n",
   (long)mDevice.RemoteNodeList["HDRIndex"].Value);

//mDevice.RemoteNodeList["HDRExposureRatio"].Value = (long)45;
```

```
//System.Console.Write("HDRExposureRatio: {0}\n",
   (long)mDevice.RemoteNodeList["HDRExposureRatio"].Value);

//System.Console.Write("HDRExposureRatioPercent: {0:F2}\n",
   (double)mDevice.RemoteNodeList["HDRExposureRatioPercent"].Value);

mDevice.RemoteNodeList["HDRPotentialAbs"].Value = (long)20; //Pot_0

System.Console.Write("HDRPotentialAbs: {0}\n",
   (long)mDevice.RemoteNodeList["HDRPotentialAbs"].Value);

System.Console.Write("\n")
```




```
Get DataStreamList and fill it
Open a Data Stream
Create the BufferList an allocate Buffer memory
Allocate Image Buffer to the DataStream
Start Camera and fill the Image Buffer
Releasing the resources
```

```
TriggerMode: On

TriggerSource: Line0

TriggerActivation: RisingEdge

ExposureMode: TriggerControlled

HDR parameter change
HDREnable: True
HDREnableTriggerAutoMode: False
HDRIndex: 0
HDRPotentialAbs: 40
HDRIndex: 1
HDRPotentialAbs: 20
```

Console Output (C#)

3 Keep in mind/Special cases

-

4 Downloads

-

5 Support

In the case of any questions or for troubleshooting please contact our support team.

Worldwide

Baumer Optronic GmbH
Badstrasse 30 · DE-01454 Radeberg
Phone +49 3528 4386 845
support.cameras@baumer.com

6 Disclaimer

Baumer
Passion for Sensors

Baumer Optronic GmbH
Badstrasse 30 · DE-01454 Radeberg
Phone +49 3528 4386 0 · Fax +49 3528 4386 86
sales@baumeroptronic.com · www.baumer.com