Teledyne-Dalsa
SaperaLT を使用した
連続アドレス領域について

本資料は、今までに事例の合った内容から抜粋したものです。
また、連続アドレス領域で画像データを取得する場合の注意事項にいて記載したものです。

SapBuffer に関しましては、Teledyne-Dalsa より提供のある資料をご参考願います。

注意事項

リングバッファ（連続アドレス領域）を使用する際のメモリ領域については、
PC の OS とメインメモリ量に依存します。
また、画像データ処理速度などは、PCIe の転送帯域と CPU 処理速度とグラフィックカードの
性能に依存するところがあります。

大容量の画像データを処理する場合、上記を注意する事とともに、下記の内容について注意が
必要です。

Contiguous memory について
Contiguous memory とは、物理アドレス（仮想アドレスではありません）が連続したメモリ空間を、
OS のブート時に SaperaLT が確保し、それをユーザーアプリケーションに提供する機能です。
この機能は既に最上位のプライオリティを与えらておりますが、同じプライオリティのリ
ソースがあると起動順序が遅れることがあります。明示的に一番先に実行させる有効な方
法は無いようです。

Contiguous memory とは、本来 OS やカーネルドライバが使用しているものであり、SaperaLT ではブー
ト後直ぐにカーネルファンクションをコールしてメモリを確保しています。
他のドライバなどが先に起動すると、取得メモリの大きさの最大値が影響されます。

OS や他の最上位のプライオリティを持つドライバなどが起動した後、SaperaLT のこの機能
が実行されるのですが、メモリ領域がフラグメンテーションしている場合など、連続する
物理メモリ領域が著しく少ないことがあり、実際にいくつかのシステムでこのような症状
が在る事が確認できております。

64bit OS になると、OS が使うリソースが増大し、この傾向が高まります。
マルチ CPU＋Windows7 にて、効率化されたパラレル処理によってブート時間が短縮してい
るのですが、複数の CPU が個々に動作するので、フラグメンテーションが発生しやすい傾
向にあるようです。

物理メモリの追加はあまり効果がなく、逆にマッピングによってリソースを使用している
ため、負の影響が出る可能性があるとのことです。

Trash バッファについて
Trash バッファに入るということは転送しようとしたバッファが Full ステートだったことを示します。
この場合、転送先のバッファを増やせるか検討が必要になります。

Contiguous memory と Trash バッファに関しては、
Sapera LT Basic Modules Reference Manual の P40 を参考願います。
また、 SapBufferWithTrash Class P38 を参照願います。
Trash バッファについて
Sapera++ Programmer's Manual の P127　SapBufferWithTrash を参照願います。

連続アドレス領域（リングバッファ）の参考例

－参考例①－
CamExpert はメモリを 5-10 フレーム程要します。
8bit として、1 フレームあたり 144MB のメモリを確保しようとします。
メモリ容量または連続仮想メモリ領域が不足することが考えられます。

簡単に行うには 64bitOS に移行し、メモリを増やすことにあります。

CamExpert は使用できませんが、以下の方法で GrabDemo を使用して画像取り込みはできます。
　　1000 ライン程度でカメラファイルを作り、テキストエディタで Crop Vertical および V Active を 12000 に変更する。
ただし、GrabDemo もバッファを 3 つ作りますので、その容量は必要です。
足りなければ以下のようにして 2 つに減らすことができます。
　　OnInitDialog()にて
　　　　m_Buffer = new SapBufferWithTrash(2, ...);
　　を
　　　　m_Buffer = new SapBufferWithTrash(1, ...);
　　とする。

－参考例②－
8K 8bit で 100Kline の場合、800MB の領域が必要です。
これを連続アドレス領域（メモリ領域ではありません）で取得する場合、32bitOS では困難な場合があります。
32bitOS ではアドレス領域は 4GB までしかありません。
フラグメンテーションが発生すると 800MB の連続アドレス領域を取得できない可能性があります。
連続アドレス領域が必要な場合は、64bitOS を使用される方がよろしいかと思います。

VirtualAlloc で 8Kx12K のメモリを確保し、確保したメモリに SapBuffer をマッピングする方法があります。
以下、サンプルです。なお、VirtualAlloc で確保したメモリを開放するためには VirtualFree()を使用してください。
VirtualAlloc、VirtualFree の詳細は MSDN をご参照ください。

```
SapAcquisition *m_Acq;
SapBuffer *m_Buffers;
SapTransfer *m_Xfer;

void* m_ptBuffers[6];
int m_BufferWidth;
int m_BufferHeight;
int m_PixelSize;

CAcqConfigDlg dlg(this, NULL);
if (dlg.DoModal() == IDOK)
{
    // Define on-line objects
    m_Acq = new SapAcquisition(dlg.GetAcquisition());

                // dummy buffer to get width, height, pixel size
                 SapBuffer* buffer;
                 SapFormat format;
                 buffer = new SapBuffer(1,m_Acq);
                if (!m_Acq->Create())
                                return FALSE;
                if (!buffer->Create())
                                return FALSE;
                 m_BufferWidth = buffer->GetWidth();
                 m_BufferHeight = buffer->GetHeight();
                 m_PixelSize = buffer->GetBytesPerPixel();
                 format = buffer->GetFormat();
                if (!buffer->Destroy())
                                return FALSE;
                if (!m_Acq->Destroy())
                                return FALSE;

                // allocate memory (12000lines)
                m_ptBuffers[0] = VirtualAlloc(NULL, m_BufferWidth * 12000 * m_PixelSize, MEM_RESERVE | MEM_COMMIT, PAGE_READWRITE);
                                m_ptBuffers[1] = m_ptBuffers[0] + m_BufferWidth * 2000 * m_PixelSize;
                                m_ptBuffers[2] = m_ptBuffers[1] + m_BufferWidth * 2000 * m_PixelSize;
                                m_ptBuffers[3] = m_ptBuffers[2] + m_BufferWidth * 2000 * m_PixelSize;
                                m_ptBuffers[4] = m_ptBuffers[3] + m_BufferWidth * 2000 * m_PixelSize;
                                m_ptBuffers[5] = m_ptBuffers[4] + m_BufferWidth * 2000 * m_PixelSize;

                 // give allocated memory info to SapBuffer and allocate 5000 lines.
                m_Buffers = new SapBufferWithTrash(6, m_ptBuffers, m_BufferWidth, m_BufferHeight, format, SapBuffer::TypeScatterGather);
                m_Xfer = new SapAcqToBuf(m_Acq, m_Buffers, XferCallback, this);
}
```