

CUDA と X を同時使用した場合の副作用

Rev	Date	Description			
	Issue	Author	Check	App	
Rev 1	Date	Description			
	Issue	Author	Check	App	
Rev 0	Date	2010/03/18	Description		
	Issue	Professional Service	Author	T.Oshima	Check
Title	The report about the side effect in the case of using Xorg and CUDA together			No	Revision
					Page

始めに

RedHawk の Version 5.4 以降には、標準で NVIDIA の CUDA™ がインストールされています。

今回は、この CUDA を Xorg と同時に利用した場合の VSIPL の計算速度に対する副作用と、そのジッターの回避法を報告します。

例題プログラム

このオリジナルプログラムは、下記サイトの VSIPL プログラムを利用しました。

<http://www.vsipl.org/documents>

[Prototype Extension of VSIPL into C++ and Object Oriented Design](#) (Dan Campbell - GTRI - September 25, 2000)

- Download [tar.gz file](#)

この報告では、GPU VSIPL を利用して評価しました。

GPU VSIPL とは

GPU VSIPL は、Georgia Institute of Technology で開発された NVIDIA の CUDA2.3 プラットフォームに実装した VSIPL で、下記サイトからダウンロードできます。

<http://gpu-vsipl.gtri.gatech.edu/index.html>

インストールは単純で、以下の手順で使うことができます。(README.txt を参照)

1. http://gpu-vsipl.gtri.gatech.edu/builds/gpuvsipl_2009Aug11.zip をダウンロードする
2. `tar xvfz gpuvsipl_2009Aug11.zip` で展開する。
3. 必要なライブラリとインクルードファイルをコピーする。
4. `gpu-vsipl/include` をインクルードパスに含める。
5. C または C++ のソースコードに `#include <vsip.h>` を加える。
6. `gpu-vsipl/linux32` または `gpu-vsipl/linux64` をライブラリパスに含める。
7. リンク時に `libgpu_vsip.a libcudart.a libcufft.a` を指定する。
例えば、GCC では: `-lgpu_vsip -lm -lcufft -lcudart`

テストプログラムの変更点について

オリジナルプログラムからの変更点は青字で示されています。

関数 `toggle()` は、`stdout` と `stderr` を切り替えます。

この関数は、GPU-VSIPL のコピーライト表示を、`stdout` から `stderr` に切り替え、`stderr` に GPU VSIPL の著作権通知をリダイレクトします。

`stdout` に書き出されたデータはバイナリの棒グラフデータです。

リアルタイム処理と独立してこのデータを表示するプログラムが作成されています。

```
# run -s fifo -P 20 -n Xorg
```

```
# shield -a 1
```

```
# run -b1 -s fifo -P1 ./vsip_gpu_sigproc|run -b2 -s fifo -P1 ./scope -p 4096 -h100 -l0
```

```
[root@rh54 sigproc]# run -b1 -s fifo -P1 ./vsip_gpu_sigproc |run -b2 -s fifo -P1 ./scope -p 4096 -h100 -l0
```

```
-----
```

GPU VSIPL (c) 2009 Georgia Tech Research Institute, All Rights Reserved

GPU VSIPL was supported in part by DARPA and AFRL under contracts

FA8750-06-1-0012 and FA8650-07-C-7724.

Title	The report about the side effect in the case of using Xorg and CUDA together	No	Revision
			Page 2

This software is provided as is without warranty or guarantee of suitability for any purpose.

Redistribution prohibited.

Driver Index: 2030

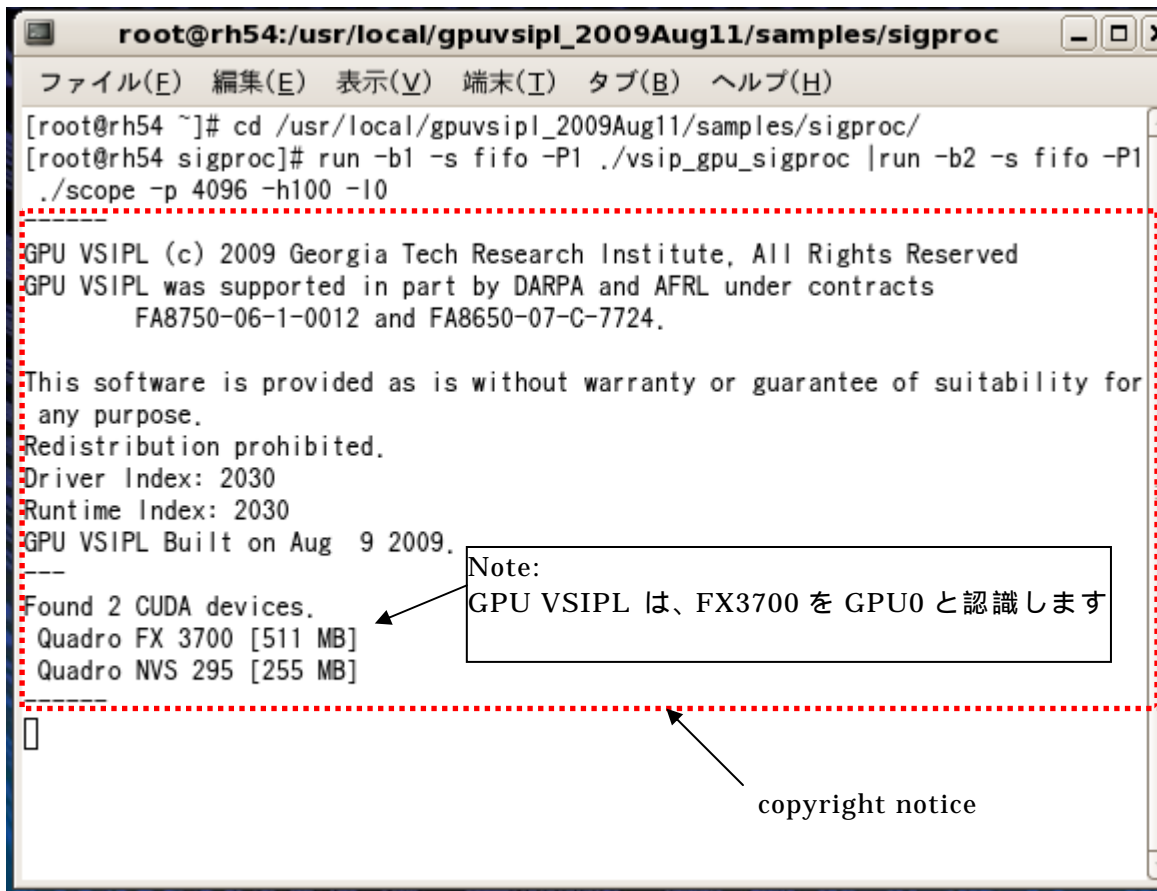
Runtime Index: 2030

GPU VSIPL Built on Aug 9 2009.

Found 2 CUDA devices.

Quadro FX 3700 [511 MB]

Quadro NVS 295 [255 MB]



```
root@rh54:/usr/local/gpuvsipl_2009Aug11/samples/sigproc
ファイル(E) 編集(E) 表示(V) 端末(T) タブ(B) ヘルプ(H)
[root@rh54 ~]# cd /usr/local/gpuvsipl_2009Aug11/samples/sigproc/
[root@rh54 sigproc]# run -b1 -s fifo -P1 ./vsip_gpu_sigproc |run -b2 -s fifo -P1
./scope -p 4096 -h100 -l0
GPU VSIPL (c) 2009 Georgia Tech Research Institute, All Rights Reserved
GPU VSIPL was supported in part by DARPA and AFRL under contracts
FA8750-06-1-0012 and FA8650-07-C-7724.
This software is provided as is without warranty or guarantee of suitability for
any purpose.
Redistribution prohibited.
Driver Index: 2030
Runtime Index: 2030
GPU VSIPL Built on Aug 9 2009.
---
Found 2 CUDA devices.
Quadro FX 3700 [511 MB]
Quadro NVS 295 [255 MB]

```

Note:
GPU VSIPL は、FX3700 を GPU0 と認識します

copyright notice

試験プログラムソースコード

```
/* sigproc_model.c Version 0.000 May 26, 2000
//
// This file contains a VSIPL signal processing model.
//
// Functional Description:
//   sigproc_model does the following exemplary signal processing:
//   Input I & Q buffers.
//   Convert fix-to-float.
//   Window the data with a Blackman window.
//   Compute the FFT of the windowed input data.
//   Compute the magnitude of the spectrum.
//   Find the max bin in the spectrum and its index.
//
// Copyright May 2000, Georgia Institute of Technology.
// This is an unpublished work. All rights reserved.
//
// Author: Randy Janka
//       randy.janka@gtri.gatech.edu
//
// Revision History:
// May 26, 2000 - Created
*/
#include <stdio.h>
#include <vsip.h>
#include <time.h>
#include <stdlib.h>
#include <unistd.h>
#include <signal.h>
#include <string.h>
#include <sys/mman.h>
#include <sys/rescntl.h>
#include "defs.h"

#define LENGTH (1024*1)          /* Length of data, blocks, and vectors */
#define N LENGTH
#define BITS 12                 /* #Bits of quantization (2's complelement) */
#define FULLSCALE 4.8828125e-004 /* 1/(2^(BITS-1)) */

#define HPOINT (4096)
static int hist[HPOINT];

void readtime(struct timespec *nowtime)
{
    clock_gettime(CLOCK_REALTIME,nowtime);
}

int adjust(struct timespec *start,struct timespec *finish,float *realtime)
{
    register int    sec,nsec;

    sec = finish->tv_sec - start->tv_sec;
    nsec = finish->tv_nsec - start->tv_nsec;
    if (nsec<0)
    {
        sec --;
        nsec += 1000000000;
    }
    *realtime = (float)(nsec/1000)+(float)(sec*1000000);
    return((int)*realtime);
}
```

Title	The report about the side effect in the case of using Xorg and CUDA together	No	Revision
			Page 4

```
void read_file (char *filename,
               vsip_scalar_f *det,
               vsip_scalar_vi *det_index,
               vsip_scalar_f *w,
               vsip_scalar_i *I,
               vsip_scalar_i *Q)
{
    FILE *fp;
    int i, read;
    vsip_scalar_i it1, it2;
    vsip_scalar_f ft1;

    fp = fopen (filename, "r");
    if (!fp)
    {
        printf ("Could not open %s for reading\n", filename);
        exit(1);
    }
    read = fscanf (fp, "%f %d\n", det, &it1);
    if (read < 2)
    {
        printf ("Read %i in det line, expecting 2\n", read);
        exit(1);
    }
    *det_index = it1;
    for (i=0; i<LENGTH; i++)
    {
        read = fscanf (fp, "%f %i %i\n", &ft1, &it1, &it2);
        if (read < 3)
        {
            printf ("Read %i not 3 on line %i of data file\n", read, i);
            exit (1);
        }
        w[i] = ft1;
        I[i] = it1;
        Q[i] = it2;
    }
    fclose (fp);
}

vsip_vview_f *rx_l, *rx_Q, *Rx_mag;
vsip_cvview_f *rx_raw, *rx, *rx_w, *Rx;
vsip_fft_f *fft_obj;

static int so=0,se=0;
void toggle(void)
{
    so = dup(STDOUT_FILENO);
    se = dup(STDERR_FILENO);
    close(STDOUT_FILENO);
    close(STDERR_FILENO);
    dup2(se,STDOUT_FILENO);
    dup2(so,STDERR_FILENO);
}

void init_function (void)
```

Title	The report about the side effect in the case of using Xorg and CUDA together	No	Revision
			Page 5

```

{
    rx_raw = vsip_cvcreate_f(N,0);          /* Create complex vector for raw input I/Q signal */
    rx      = vsip_cvcreate_f(N,0);          /* Create complex vector for scaled/normalized I/Q signal */
    rx_w    = vsip_cvcreate_f(N,0);          /* Create complex vector for windowed input signal */
    rx_I    = vsip_vrealview_f(rx_raw);      /* Create vector view of Re[rx] */
    rx_Q    = vsip_vimagview_f(rx_raw);      /* Create vector view of Im[rx] */
    Rx      = vsip_cvcreate_f(N,0);          /* Create FFT output vector */
    Rx_mag  = vsip_vcreate_f(N,0);           /* Create |FFT| vector */
    fft_obj = vsip_ccfftop_create_f(LENGTH,1,VSIP_FFT_FWD,1,VSIP_ALG_TIME); /* Create FFT object */
}

void do_function (vsip_vview_i *vec_I,
                 vsip_vview_i *vec_Q,
                 vsip_vview_f *vec_w,
                 vsip_scalar_f *det,
                 vsip_scalar_vi *det_index)
{
    vsip_scalar_f full_scale=FULLSCALE;
    vsip_vcopy_i_f(vec_I,rx_I);             /* Copy/convert (int)I input to (float)Re[rx] */
    vsip_vcopy_i_f(vec_Q,rx_Q);             /* Copy/convert (int)Q input to (float)Im[rx] */
    vsip_rscvmul_f (full_scale,rx_raw,rx);   /* Scale A/D values by full-scale */
    vsip_rcvmul_f (vec_w,rx,rx_w);          /* Apply Blackman window to input signal */
    vsip_ccfftop_f(fft_obj,rx_w,Rx);        /* Compute FFT of windowed input signal */
    vsip_cvmag_f (Rx, Rx_mag);              /* Compute magnitude of FFT to get spectral estimate */
    *det=vsip_vmaxval_f(Rx_mag, det_index); /* Find max bin value and bin index */
}

static struct resched_var rv;
static int Stop_flag=0;
static int global_setup_signal(int signo, void (*interrupt_handler)(int,signo_t *, void *))
{
    static struct sigaction newact;
    static sigset_t set,oset;

    if (sigprocmask(0,NULL,&set)==(-1))
    {
        return(-1);
    }
    sigdelset(&set,signo);
    if (sigprocmask(SIG_SETMASK,&set,&oset)==(-1))
    {
        return(-2);
    }
    sigemptyset(&newact.sa_mask);
    sigaddset(&newact.sa_mask,signo);
    newact.sa_sigaction = interrupt_handler;
    newact.sa_flags = SA_SIGINFO|SA_RESTART;
    if (sigaction(signo, &newact, 0)==(-1))
    {
        return(-3);
    }
    return(0);
}

static void sigint_handler(int signo, signo_t *siginfo, void *misc)
{
    if ((siginfo->si_pid==0)||((getpid()==siginfo->si_pid)|| (getppid()==siginfo->si_pid)))
    {
        fprintf(stderr,"%nSIG%d DETECT%n",signo);
        Stop_flag = 1;
    }
}

int main(int argc, char *argv[])
{

```

Title	The report about the side effect in the case of using Xorg and CUDA together	No	Revision
			Page 6

```

vsip_scalar_i I[N], /* In-phase (real) part of received A/D'd signal */
                Q[N]; /* Quadrature-phase (imaginary) part of received A/D'd signal */
vsip_scalar_f w[N]; /* Window function to apply to received signal before FFT */
vsip_scalar_vi det_index, target_det_index;
vsip_scalar_f det, target_det;
vsip_block_i *blk_I, *blk_Q;
vsip_block_f *blk_w;
vsip_vview_i *vec_I, *vec_Q;
vsip_vview_f *vec_w;
int i,hex;
struct timespec start, end;
struct timespec t0, t1;
float time,rtime;

resched_cntl (RESCHED_SET_VARIABLE, (char *)&rv);
global_setup_signal(SIGINT,sigint_handler);
toggle();
vsip_init((void *)0);
toggle();
memset(hist,0,HPOINT*sizeof(int));
fwrite(hist, sizeof(long),HPOINT,stdout);
mlockall(MCL_CURRENT|MCL_FUTURE);
blk_I = vsip_blockbind_i(I,N,0); /* Bind block to I user array */
blk_Q = vsip_blockbind_i(Q,N,0); /* Bind block to Q user array */
blk_w = vsip_blockbind_f(w,N,0); /* Bind block to w user array */
vec_I = vsip_vbind_i(blk_I, 0, 1, N); /* Bind vector view to blk_I block */
vec_Q = vsip_vbind_i(blk_Q, 0, 1, N); /* Bind vector view to blk_Q block */
vec_w = vsip_vbind_f(blk_w, 0, 1, N); /* Bind vector view to blk_w block */
read_file ("IQdata", &target_det, &target_det_index, w, I, Q);
vsip_blockadmit_f(blk_w, 1); /* Admit blk_w after populating window user data array */
vsip_blockadmit_i(blk_I, 1); /* Admit blk_I after populating input-I user data array */
vsip_blockadmit_i(blk_Q, 1); /* Admit blk_Q after populating input-Q user data array */
init_function();
readtime(&t0);
while(Stop_flag==0)
{
    for (i=0; i<COMPLEX_REPETITIONS; i++)
    {
        resched_lock (&rv);
        readtime(&start);
        do_function (vec_I, vec_Q, vec_w, &det, &det_index);
        readtime(&end);
        resched_unlock (&rv);
        hex=adjust(&start,&end,&time);
        hist[hex]++;
    }
    fwrite(hist, sizeof(long)*HPOINT,1,stdout);fflush(stdout);
    memset(hist,0,HPOINT*sizeof(int));
}
vsip_finalize ((void *)0);
return 0;
}

```

Title	The report about the side effect in the case of using Xorg and CUDA together	No	Revision
			Page 7

/etc/X11/xorg.conf

```
# nvidia-xconfig: X configuration file generated by nvidia-xconfig
# nvidia-xconfig: version 1.0 (buildmeister@builder63) Tue Oct 20 21:00:15 PDT 2009
```

Section "ServerLayout"

```
Identifier      "Layout0"
Screen         0  "Screen0"
InputDevice    "Keyboard0" "CoreKeyboard"
InputDevice    "Mouse0" "CorePointer"
```

EndSection

Section "Files"

```
FontPath       "unix/:7100"
```

EndSection

Section "Module"

```
Load           "dbe"
Load           "extmod"
Load           "type1"
Load           "freetype"
Load           "glx"
```

EndSection

Section "InputDevice"

```
# generated from default
Identifier     "Mouse0"
Driver        "mouse"
Option        "Protocol" "auto"
Option        "Device"  "/dev/psaux"
Option        "Emulate3Buttons" "no"
Option        "ZAxisMapping" "4 5"
```

EndSection

Section "InputDevice"

```
# generated from data in "/etc/sysconfig/keyboard"
Identifier     "Keyboard0"
Driver        "kbd"
Option        "XkbLayout" "jp"
Option        "XkbModel" "jp106"
```

EndSection

Section "Monitor"

```
Identifier     "Monitor0"
VendorName     "Unknown"
ModelName     "Unknown"
HorizSync     28.0 - 33.0
VertRefresh   43.0 - 72.0
Option        "DPMS"
```

EndSection

Section "Device"

```
Identifier     "Device0"
Driver        "nvidia"
VendorName     "NVIDIA Corporation"
BoardName     "Quadro NVS 295"
BusID         "PCI:1:0:0"
```

EndSection

Title	The report about the side effect in the case of using Xorg and CUDA together	No	Revision
			Page 8

Section "Device"

Identifier "Device1"
Driver "nvidia"
VendorName "NVIDIA Corporation"
BoardName "Quadro FX 3700"
BusID "PCI:2:0:0"

EndSection

Xorg の実行 GPU は、「Device0」を
「Device1」に変えることによって、変
更することができます。

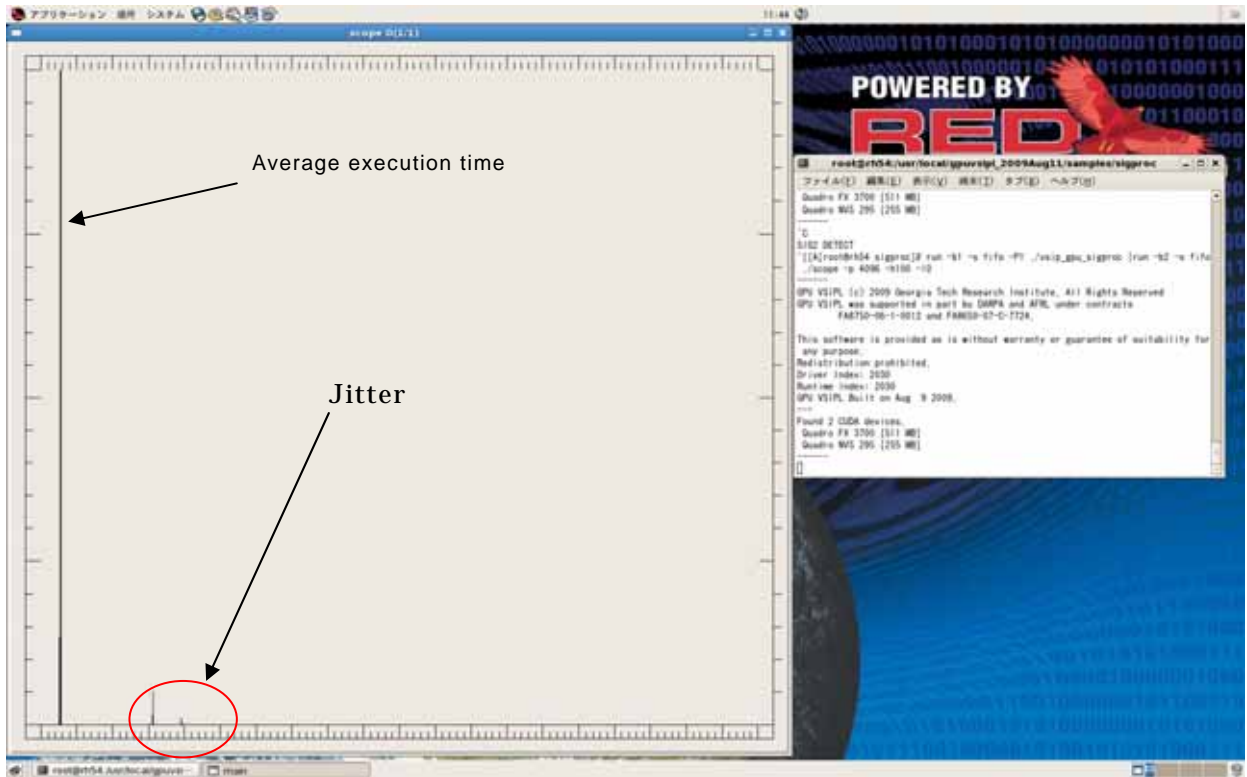
Section "Screen"

Identifier "Screen0"
Device "Device0"
Monitor "Monitor0"
DefaultDepth 24
SubSection "Display"
Depth 24

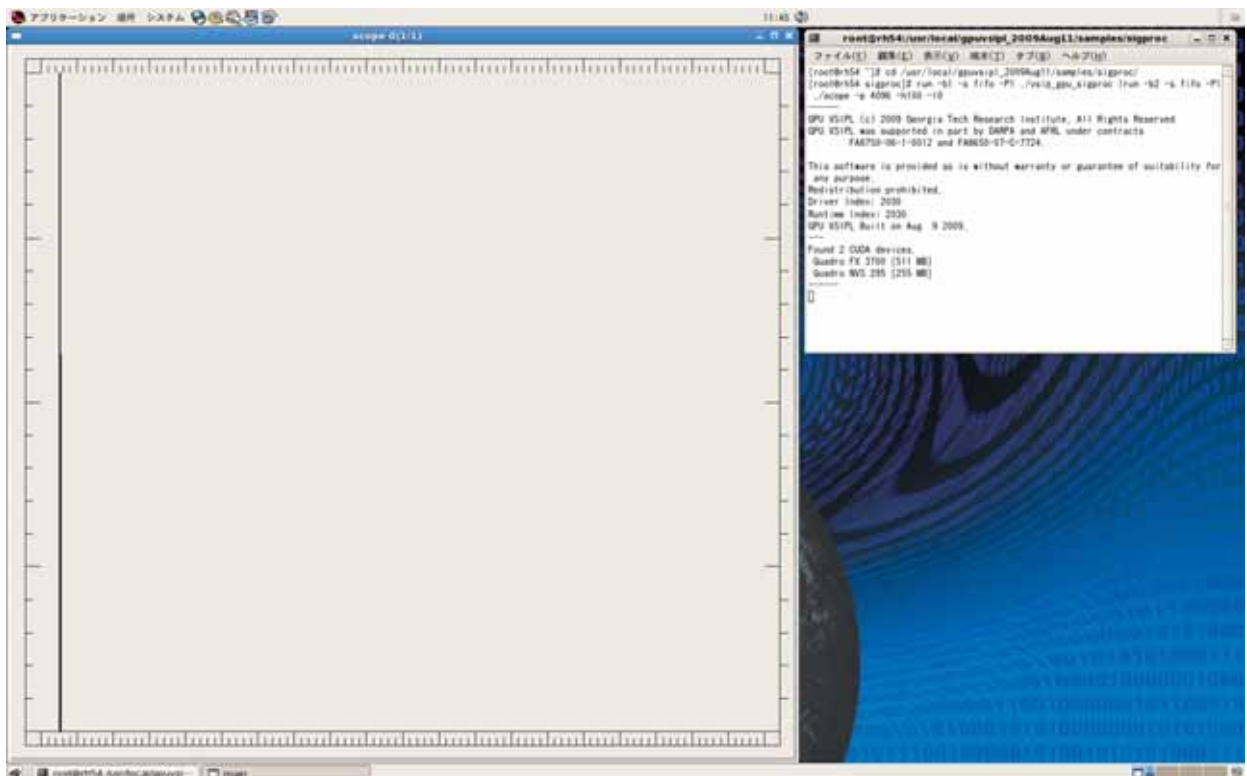
EndSubSection

EndSection

CUDA 計算と Xorg を 1st GPU で処理した場合の結果。
 マウスを移動させるとシッターの値と数は増加します。
 このジッターは、平均実行時間の 3 倍以上の値を示します。



CUDA 計算を 1st GPU で行い、Xorg の処理を 2nd GPU で行った場合の結果。
 1st GPU での処理結果にはジッターは含まれません。これは、2ndGPUですべてXの処理を行
 っているためです。



Title The report about the side effect in the case of using Xorg and CUDA together	No	Revision
		Page 10

まとめ

CUDA は数値計算アクセラレータとしてのコストパフォーマンスが非常に優れています。しかしながら、リアルタイム処理においては、ジッターが存在してはなりません。ジッターを回避するためには、2つの `Nvidia` をインストールし、そして `Xorg` を `2ndGPU` で実行させる必要があります。

Title	The report about the side effect in the case of using Xorg and CUDA together	No	Revision
			Page 11