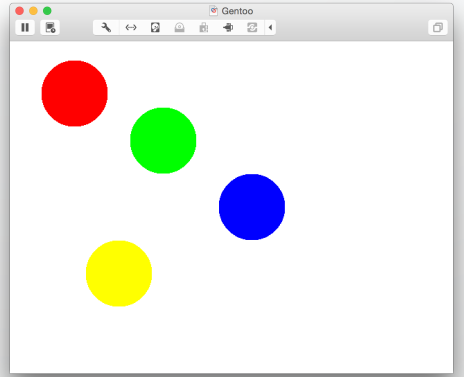
MappedByteBuffer.hurray()!

Programming the Linux Framebuffer in Java

Christopher Friedt
Principle Embedded Firmware Engineer

chris@mmbnetworks.com
chrisfriedt@gmail.com

code available at http://github.com/cfriedt



MMB Networks
Simply Connected

Overview

History
Hypothesis
Apparatus
Methods
Observations
Conclusions
Thanks
Questions

MMB Networks
Simply Connected

- History
- Hypothesis
- Apparatus
- Methods
- Observations
- Conclusions
- Thanks
- Questions

History

A long time ago… in a galaxy far, far away…

JAR WARS: Revenge of the Disc

Oracle vs. Google

History

- Oracle claimed that Android had fragmented the Java API
    - Android used the Java for apps (at least syntactically)
    - new, different, and widely adopted Java windowing API
    - Java ME ... wot ??

History

- I thought it would be a good idea to create an **AWT** port for Android
- **AWT - Abstract Windowing Toolkit**
- Each platform must implement AWT in order to support windows, buttons, forms, ...
- Oracle would have a lesser basis for their lawsuit against Google / Android OHA

History

Eventually saw the **Caciocavallo Project**

- http://openjdk.java.net/projects/caciocavallo
- Framework for developing an AWT port
- Enables one to (possibly) create an AWT port entirely in Java
- Java is great for rapid prototyping

History

After some effort, I had written a few widgets, and then experienced **deja vu**.

- Ready to start painting to a screen!
- but then I couldn't map the Linux Framebuffer
    - Needed to **use JNI to map** /dev/fb0
- and then I remembered Video For Linux (V4L) several years prior
    - Needed to **use JNI to map** /dev/video0
- Why duplicate code? Why doesn't FileChannel.map() **Just Work™**?
    - because /dev nodes are "special"

History

Back to the **Oracle vs. Google**...

- Luckily, Google (kind of) won that lawsuit - **for all of us**!
- ”**So long as the specific code used to implement a method is different, anyone is free under the Copyright Act to write his or her own code to carry out exactly the same function or specification of any methods used in the Java API. It does not matter that the declaration or method header lines are identical.**”
- U.S. Copyright Act: 102(b) ... **"system or method of operation.”**
- I was free to put my ***just for fun*** project on the back burner...

MMB Networks
Simply Connected

History
Hypothesis
Apparatus
Methods
Observations
Conclusions
Thanks
Questions

Hypothesis

“With a small bit of hacking on weekends,

I can get this to work in no time at all !!~!”

-- some idiot, 5 years ago

MMB Networks
Simply Connected

Hypothesis

- **Every OS* implements **mmap(2)**
- Java has **MappedByteBuffer**, via **FileChannel.map()**, extends **ByteBuffer**
- Able to get a FileChannel object with **RandomAccessFile.getChannel()**
- All of the above **works with a plain text file**, but **does not work on /dev/XXX**
- Unable to get **MappedByteBuffer.array()** object ()
    - kind of required for direct pixelpushing
- The above is true for ***any JVM*

* that I care to use

** that I have been able to test

MMB Networks
Simply Connected

- History
- Hypothesis
- Apparatus
- Methods
- Observations
- Conclusions
- Thanks
- Questions

Apparatus

JamVM

- Most familiar with hacking
- Squashing into embedded since 2006
- Easy to modify
- Multi-platform
- Multi-classpath
- Met the author at FOSDEM'12
- … straaangely similar to Dalvik ...

GNU Classpath

- Most familiar with hacking
- Squashing into embedded since 2006
- Good code structure
- Several existing AWT implementations
- Easy to add a new, Framebuffer AWT

Apparatus

VMWare

- Would prefer to *not* need a separate test machine
- Easier for others to test on Mac / Linux / Windows
- Can easily run Linux in VMWare
- VMWare has *a Linux framebuffer driver

Linux

- Most familiar with hacking
- I have been squashing it into stuff since 1998
- Pretty OK code structure…
- Linux runs on a few things…
- Easy to add a new stuff

* quasi-functional

Apparatus

Java

```
FB4JFrameBuffer fb = new FrameBuffer();
FB4JVarScreenInfo vinfo = fb.getVarScreenInfo();
vinfo.setXresVirtual( vinfo.getXres() );
vinfo.setYresVirtual( 2 * vinfo.getYres() );
fb.putVarScreenInfo( vinfo );
FB4JFixScreenInfo finfo = fb.getFixScreenInfo();
int[] pixel = fb.asByteBuffer().asIntBuffer().array();
final int w = vinfo.getXres();
final int h = vinfo.getYres();
final int hmax = vinfo.getYresVirtual();
```

Peanut Gallery

```
// ioctl via JNI / JNA
// ioctl via JNI / JNA
// relies on ByteBuffer .read() / .put()!
// relies on ByteBuffer .read() / .put()!
// ioctl via JNI / JNA
// ioctl via JNI / JNA
// calls FileChannel.map() on special node
```

Apparatus

Java

```
for( int yoffs = h ;; yoffs += h, yoffs %= hmax ) {
  // draw stuff
  fb.flip();
}
```

Peanut Gallery

```
// initially draw to 1 / N back buffers, loop 4 EVAR!!

// ioctl via JNI / JNA
// code available at http://github.com/cfriedt/fb4j
```

MMB Networks
Simply Connected

History
Hypothesis
Apparatus
Methods
Observations
Conclusions
Thanks
Questions

Methods

Ensure **VMWare / Linux Framebuffer** is accessible and can flip pages natively

- #include <sys/mman.h>
  void ***mmap**(void *addr, size_t length, int prot, int flags, int fd, off_t offset);
  int **munmap**(void *addr, size_t length);

- #include <sys/ioctl.h>
  int **ioctl**(int d, unsigned long request, ...);

- #include <**linux/fb.h**>

MMB Networks
Simply Connected

Methods

Ensure **VMWare / Linux Framebuffer** is accessible and can flip pages natively

- int fd = **open**("/dev/fb0", O_RDWR);
- struct fb_var_screeninfo vinfo = {};
  ioctl(fd, **FBIOGET_VSCREENINFO**, &vinfo);
- vinfo.xres_virtual = vinfo.xres;
  vinfo.yres_virtual = 2 * vinfo.yres; // front & back buffer
  ioctl(fd, **FBIOPUT_VSCREENINFO**, &vinfo);
- size_t maplen = vinfo.xres_virtual * vinfo.yres_virtual * vinfo.bits_per_pixel / 8;

Methods

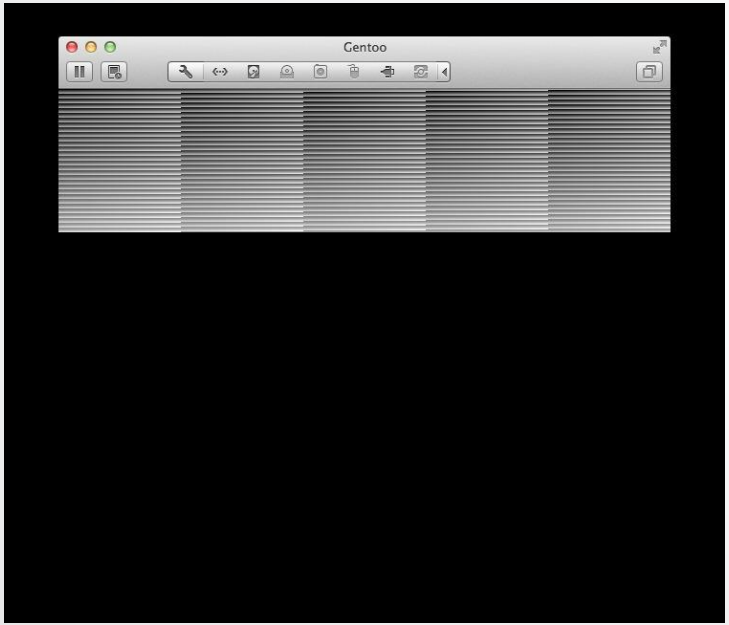
Ensure VMWare / Linux Framebuffer is accessible and can flip pages natively

- uint8_t *map =
  mmap(NULL, maplen, PROT_READ | PROT_WRITE, MAP_SHARED, fd, 0);

- ```
  uint16_t w = vinfo.yres;
  uint16_t h = vinfo.xres;
  for( uint16_t yoffs = h ;; yoffs += h, yoffs %= vinfo.yres_virtual ) {
      uint8_t *pixel = &map[ yoffs * w ];
      vinfo.yoffset = yoffs;
      // … draw stuff to back buffer
      ioctl( fd, FBIOPAN_DISPLAY, &vinfo );
  }
  ```
  ```

MMB Networks
Simply Connected

Methods



https://bugs.gentoo.org/show_bug.cgi?id=494794

Linux VMWare Framebuffer Driver

Two problems:

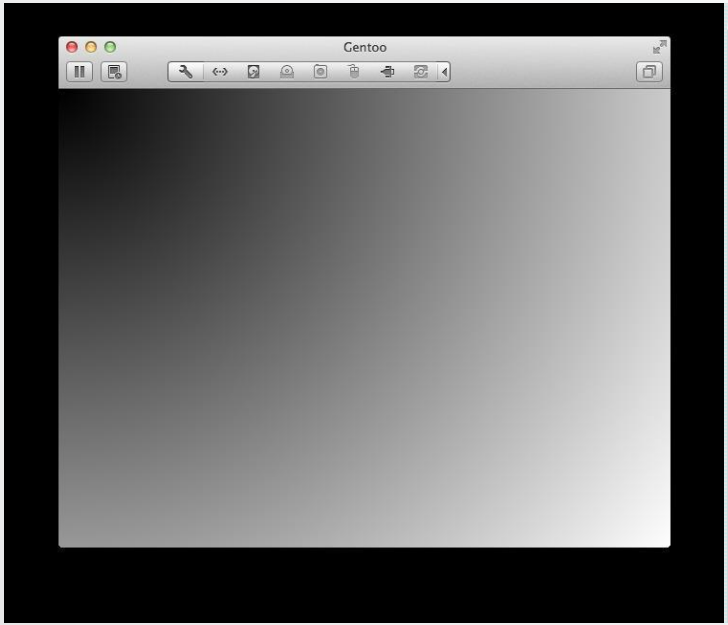
1) fb_fix_screeninfo.line_length

2) FBIOPAN_DISPLAY broken

```
int vmw_fb_pan_display(
  struct fb_var_screeninfo *var,
  struct fb_info *info) {
    return 0;
}
// should at least return -ENOSYS!
```

MMB Networks
Simply Connected

Methods (fix problem 1 in vmwgfx_fb.c)




```
From     Christopher Friedt <>
Subject  [PATCH 1/1] drm/vmwgfx: correct fb_fix_screeninfo.line_length
Date     Sat, 1 Feb 2014 10:26:55 -0500

Previously, the vmwgfx_fb driver would allow users to call FBIOSET_VINFO, but it would not adjust
the FINFO properly, resulting in distorted screen rendering. The patch corrects that behaviour.

See https://bugs.gentoo.org/show_bug.cgi?id=494794 for examples.

Signed-off-by: Christopher Friedt <chrisfriedt@gmail.com>
---
 drivers/gpu/drm/vmwgfx/vmwgfx_fb.c | 5 ++++-
 1 file changed, 4 insertions(+), 1 deletion(-)
diff --git a/drivers/gpu/drm/vmwgfx/vmwgfx_fb.c b/drivers/gpu/drm/vmwgfx/vmwgfx_fb.c
index ed5ce2a..021b522 100644
--- a/drivers/gpu/drm/vmwgfx/vmwgfx_fb.c
+++ b/drivers/gpu/drm/vmwgfx/vmwgfx_fb.c
@@ -147,7 +147,7 @@ static int vmw_fb_check_var(struct fb_var_screeninfo *var,
         }

         if (!vmw_kms_validate_mode_vram(vmw_priv,
-                                        info->fix.line_length,
+                                        var->xres * var->bits_per_pixel/8,
                                         var->yoffset + var->yres)) {
                 DRM_ERROR("Requested geom can not fit in framebuffer\n");
                 return -EINVAL;
@@ -162,6 +162,8 @@ static int vmw_fb_set_par(struct fb_info *info)
         struct vmw_private *vmw_priv = par->vmw_priv;
         int ret;

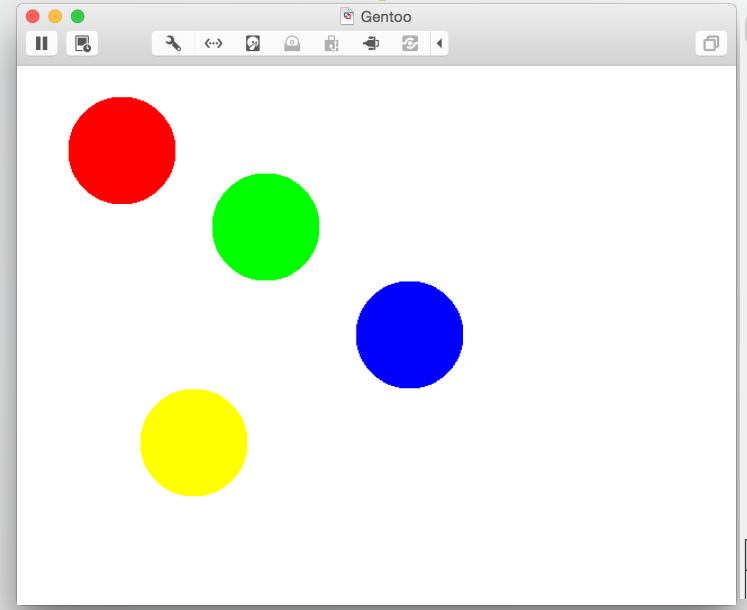
+       info->fix.line_length = info->var.xres * info->var.bits_per_pixel/8;
+
         ret = vmw_kms_write_svga(vmw_priv, info->var.xres, info->var.yres,
                                  info->fix.line_length,
                                  par->bpp, par->depth);
@@ -177,6 +179,7 @@ static int vmw_fb_set_par(struct fb_info *info)
                 vmw_write(vmw_priv, SVGA_REG_DISPLAY_WIDTH, info->var.xres);
                 vmw_write(vmw_priv, SVGA_REG_DISPLAY_HEIGHT, info->var.yres);
+                vmw_write(vmw_priv, SVGA_REG_BYTES_PER_LINE, info->fix.line_length);
                 vmw_write(vmw_priv, SVGA_REG_DISPLAY_ID, SVGA_ID_INVALID);
         }

--
1.8.3.2
```

https://bugs.gentoo.org/show_bug.cgi?id=494794

MMB Networks
Simply Connected

Methods (fix problem 2 in vmwgfx_fb.c)



Bug 496516 - enable double-buffering in the vmware fbdev driver

Status: RESOLVED NEEDINFO

Product: Gentoo Linux
Component: Core system
Version: 10.1
Hardware: All Linux

Importance: Normal normal (vote)
Target Milestone: ---
Assigned To: Gentoo Kernel Bug Wranglers and Kernel Maintainers

URL: https://plus.google.com/1070582778123...
Whiteboard:
Keywords: PATCH

Depends on:
Blocks:
Show dependency tree

Attachments	
linux-3.10.7-gentoo-enable-vmwgfx-double-buffering.patch (5.13 KB, patch)	Details | Diff

https://bugs.gentoo.org/show_bug.cgi?id=496516

Methods

Ensure **Classpath** is able to

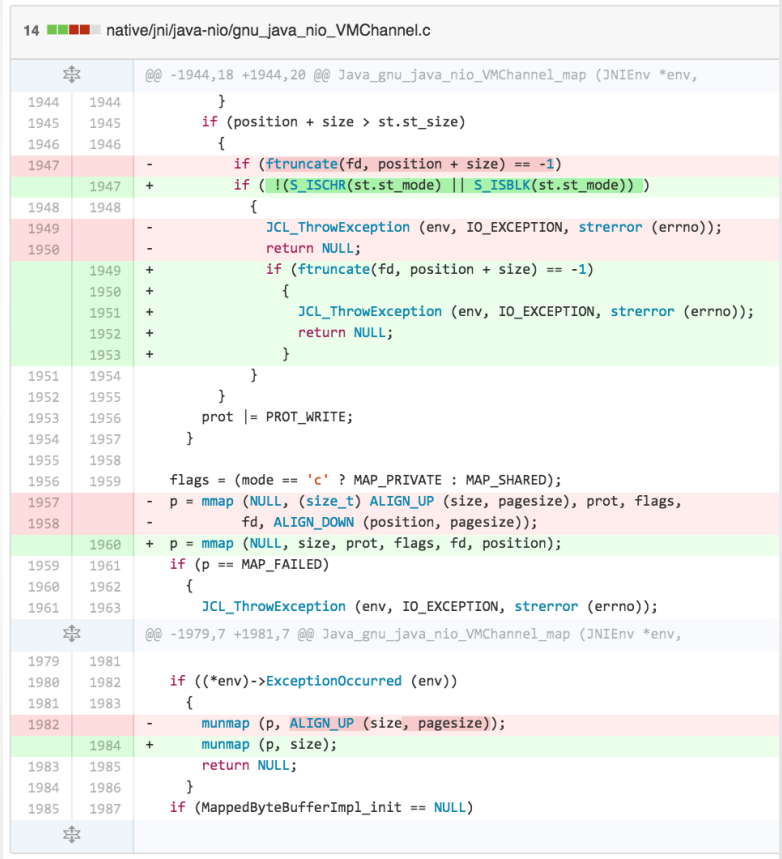
1) **mmap(2)** special files
2) FileDescriptor / VMChannel provides integer file descriptor, for JNI FB ioctls
   a) Available in most VMs for a very long time* (search kfu.com java file descriptor)
   b) Android may use a slightly different field name (other than “fd”)
3) Pass Pointer object from classpath to VM, portably, so VM can allocate array object
   a) New classes: **VMFlexArray**, **VMFlexArrayInfo**
4) **sun.misc.Unsafe** support
5) **Buffer** and subclasses
6) Object LifeCycle

MMB Networks
Simply Connected

Methods: Special Files

- **mmap(2)** special files
- **VMChannel.map()**
- Simple check to see if the the file is special using S_ISCHR(), S_ISBLK())
- aligning up with mmap(2) is not necessary on
  - Linux
  - Mac OS X

https://github.com/cfriedt/classpath/compare/mmap-special-files



14 native/jni/java-nio/gnu_java_nio_VMChannel.c

```
@@ -1944,18 +1944,20 @@ Java_gnu_java_nio_VMChannel_map (JNIEnv *env,
          }
        if (position + size > st.st_size)
          {
-           if (ftruncate(fd, position + size) == -1)
+           if ( !(S_ISCHR(st.st_mode) || S_ISBLK(st.st_mode)) )
              {
-               JCL_ThrowException (env, IO_EXCEPTION, strerror (errno));
-               return NULL;
+               if (ftruncate(fd, position + size) == -1)
+                 {
+                   JCL_ThrowException (env, IO_EXCEPTION, strerror (errno));
+                   return NULL;
+                 }
              }
          }
        prot |= PROT_WRITE;
      }

    flags = (mode == 'c' ? MAP_PRIVATE : MAP_SHARED);
-   p = mmap (NULL, (size_t) ALIGN_UP (size, pagesize), prot, flags,
-             fd, ALIGN_DOWN (position, pagesize));
+   p = mmap (NULL, size, prot, flags, fd, position);
    if (p == MAP_FAILED)
      {
        JCL_ThrowException (env, IO_EXCEPTION, strerror (errno));
@@ -1979,7 +1981,7 @@ Java_gnu_java_nio_VMChannel_map (JNIEnv *env,

    if ((*env)->ExceptionOccurred (env))
      {
-       munmap (p, ALIGN_UP (size, pagesize));
+       munmap (p, size);
        return NULL;
      }
    if (MappedByteBufferImpl_init == NULL)
```

MMB Networks
Simply Connected

Methods: File Descriptor

integer file descriptor

- **VMChannel(final int native_fd)**
- public so that it is accessible from FileDescriptor class

2 vm/reference/gnu/java/nio/VMChannel.java

```
@@ -85,7 +85,7 @@ public VMChannel()
85  85      * @param native_fd The native file descriptor integer.
86  86      * @throws IOException
87  87      */
88      -   VMChannel(final int native_fd) throws IOException
    88  +   public VMChannel(final int native_fd) throws IOException
89  89      {
90  90        this();
91  91        this.nfd.setNativeFD(native_fd);
```

https://github.com/cfriedt/classpath/compare/add-integer-filedescriptor

MMB Networks
Simply Connected

Methods: File Descriptor

integer file descriptor

- **int fd**
- 2 new constructors requried for
  FileChannelImpl
  VMChannel
- **isValid()**

https://github.com/cfriedt/classpath/compare/add-integer-filedescriptor

17 java/io/FileDescriptor.java

```
@@ -40,6 +40,7 @@
 package java.io;

 import gnu.java.nio.FileChannelImpl;
+import gnu.java.nio.VMChannel;

 import java.nio.channels.ByteChannel;
 import java.nio.channels.FileChannel;
@@ -80,6 +81,7 @@
   = new FileDescriptor (FileChannelImpl.err);

   final ByteChannel channel;
+  int fd;

   /**
    * This method is used to initialize an invalid FileDescriptor object.
@@ -97,6 +99,17 @@ public FileDescriptor()
     this.channel = channel;
   }

+  FileDescriptor(FileChannelImpl channel)
+  {
+    this.channel = channel;
+    this.fd = channel.getNativeFD();
+  }
+
+  FileDescriptor(int fd, int mode) throws IOException
+  {
+    channel = new FileChannelImpl(new VMChannel(fd), mode);
+    this.fd = fd;
+  }

   /**
    * This method forces all data that has not yet been physically written to
@@ -135,6 +148,8 @@ public void sync () throws SyncFailedException
   public boolean valid ()
   {
     ByteChannel c = channel;
-    return (c != null) && (c.isOpen());
+    boolean valid = (c != null) && (c.isOpen());
+    valid = (channel instanceof FileChannel) ? (fd >= 0) : valid;
+    return valid;
   }
 }
```

MMB Networks
Simply Connected

Methods: File Descriptor

integer file descriptor

- Make constructor visible to FileDescriptor
- Uncomment getNativeFD()

https://github.com/cfriedt/classpath/compare/add-integer-filedescriptor

11 gnu/java/nio/FileChannelImpl.java

```
@@ -176,7 +176,7 @@ private FileChannelImpl(File file, int mode)
176 176     *
177 177     * @param mode READ or WRITE
178 178     */
179     -   FileChannelImpl (VMChannel ch, int mode)
    179 +   public FileChannelImpl (VMChannel ch, int mode)
180 180     {
181 181       this.mode = mode;
182 182       this.description = "descriptor(" + ch.getState() + ")";
@@ -564,9 +564,14 @@ public String toString()
564 564
565 565     /**
566 566      * @return The native file descriptor.
567     -    */
    567 +    */
568 568     public int getNativeFD()
569 569     {
    570 +     int fd = -1;
    571 +     try {
    572 +       fd = ch.getState().getNativeFD();
    573 +     } catch ( IOException e ) {
    574 +     }
570 575       return fd;
571     - }*/
    576 + }
572 577   }
```

MMB Networks
Simply Connected

Methods: VMFlexArray

VMFlexArray

- flexible because they allow both regularly allocated java arrays and arrays defined by arbitrary pointers
- uses system property
  gnu.classpath.flexarray.enable
- uses sun.misc.Unsafe, reflection
- statically initialized once upon VM init
- static Object pointerToArray(Pointer address, int capacity, int array_offset, Class<?> cls);

https://github.com/cfriedt/classpath/compare/use-sun-misc-unsafe-for-pointer-arrays

VMFlexArrayInfo

- Private static interface **IVMFlexArrayInfo**
- A VMFlexArrayInfo is needed for each VM that supports VMFlexArray
- String jamvminfo = System.getProperty("java.vm.info");
  flexible = null == jamvminfo ? false : jamvminfo.contains("flexarray");
- ```
  private static interface IVMFlexArrayInfo {
    boolean isArrayObjectFlexible();
    int arraySizeOffset();
    int dataPointerOffset();
  }
  ```
  ```

MMB Networks
Simply Connected

Methods: sun.misc.Unsafe

18 vm/reference/sun/misc/Unsafe.java

```
@@ -78,6 +78,24 @@ public static Unsafe getUnsafe()
78  78     }
79  79
80  80     /**
    81 +    * Report the size in bytes of a native pointer, as stored via
    82 +    * <code>putAddress</code>. This value will be either 4 or 8.  Note that
    83 +    * the sizes of other primitive types (as stored in native memory blocks)
    84 +    * is determined fully by their information content.
    85 +    *
    86 +    * @return the size of an address
    87 +    */
    88 +   public native int addressSize();
    89 +
    90 +   /**
    91 +    * Allocate an instance but do not run any constructor.
    92 +    * Initializes the class if it has not yet been.
    93 +    *
    94 +    * @return an Object
    95 +    */
    96 +   public native Object allocateInstance( Class cls ) throws InstantiationException;
    97 +
    98 +   /**
81  99     * Returns the memory address offset of the given static field.
82  100    * The offset is merely used as a means to access a particular field
83  101    * in the other methods of this class.  The value is unique to the given
```

Unsafe

Requires two additional method declarations in Classpath. Implementations are in the VM.

1) addressSize()
   a) used by VMFlexArrayInfo, VMFlexArray
2) allocateInstance()
   a) used by VMFlexArray

https://github.com/cfriedt/classpath/compare/use-sun-misc-unsafe-for-pointer-arrays

MMB Networks
Simply Connected

Methods: Buffers, Views et al

9 java/nio/IntViewBufferImpl.java

```
@@ -48,8 +48,13 @@
48  48
49  49   IntViewBufferImpl (ByteBuffer bb, int capacity)
50  50   {
51      -    super (capacity, capacity, 0, -1, bb.isDirect() ?
52      -           VMDirectByteBuffer.adjustAddress(bb.address, bb.position()):null, null, 0);
    51  +    super (capacity, capacity, 0, -1,
    52  +      bb.isDirect()
    53  +        ? VMDirectByteBuffer.adjustAddress(bb.address, bb.position()):null,
    54  +      bb.hasArray()
    55  +        ? (int[]) VMFlexArray.pointerToArray(bb.address, capacity,
    56  +                                             bb.position(), int[].class):null,
    57  +      0);
53  58       this.bb = bb;
54  59       this.offset = bb.position();
55  60       this.readOnly = bb.isReadOnly();
```

5 java/nio/LongBuffer.java

```
@@ -53,7 +53,10 @@
53  53               Pointer address, long[] backing_buffer, int array_offset)
54  54   {
55  55     super (capacity, limit, position, mark, address);
56      -  this.backing_buffer = backing_buffer;
    56  +  this.backing_buffer =
    57  +    backing_buffer == null
    58  +    ? (long[])pointerToArray( address, capacity, array_offset, long[].class )
    59  +    : backing_buffer;
57  60     this.array_offset = array_offset;
58  61   }
59  62
```

Buffers & Views

ByteBuffer, CharBuffer, ShortBuffer, … really just rely on VMFlexArray.DirectpointerToArray()

Views allow one type of buffer to be interpreted as having different types of elements, not unlike a cast.

https://github.com/cfriedt/classpath/compare/use-sun-misc-unsafe-for-pointer-arrays

MMB Networks
Simply Connected

Methods: Mapped Array LifeCycle

4 java/nio/DirectByteBufferImpl.java

```
@@ -110,7 +110,7 @@ public boolean isReadOnly()
110 110    DirectByteBufferImpl(int capacity)
111 111    {
112 112      super(capacity, capacity, 0, -1,
113     -        VMDirectByteBuffer.allocate(capacity), null, 0);
    113 +      VMDirectByteBuffer.allocate(capacity), null, 0);
114 114      this.owner = this;
115 115    }
116 116
@@ -138,7 +138,7 @@ public static ByteBuffer allocate(int capacity)
138 138
139 139    protected void finalize() throws Throwable
140 140    {
141     -    if (owner == this)
    141 +    if (owner == this && null != this.backing_buffer )
142 142        VMDirectByteBuffer.free(address);
143 143    }
144 144
```

VMDirectByteBuffer

- MappedByteBuffers umap their memory upon finalization – same as before
- DirectByteBuffers only free memory they allocate (and only if there is no backing buffer) upon finalization – same as before

https://github.com/cfriedt/classpath/compare/buffers

Methods

Ensure **JamVM**

1) Implements methods in **sun.misc.Unsafe**
2) Supports **VMFlexArray**

MMB Networks
Simply Connected

Methods: sun.misc.Unsafe

Additional Methods

- **int addressSize()**
  - 32-bit (4-bytes), 64-bit (8-bytes)
- **Object allocateInstancw(Class<?> cls)**
  - allocate but do not initialize an object

https://github.com/cfriedt/jamvm/compare/additional-unsafe-methods

2 src/classlib/gnuclasspath/natives.c

```
@@ -1416,6 +1416,8 @@ VMMethod vm_stack_walker[] = {
 };

 VMMethod sun_misc_unsafe[] = {
+    {"addressSize",                NULL, addressSize},
+    {"allocateInstance",           NULL, allocateInstance},
     {"objectFieldOffset",          NULL, objectFieldOffset},
     {"compareAndSwapInt",          NULL, compareAndSwapInt},
```

6 src/natives.c

```
@@ -314,7 +314,11 @@ uintptr_t *putObject(Class *class, MethodBlock *mb, uintptr_
 }

 uintptr_t *arrayBaseOffset(Class *class, MethodBlock *mb, uintptr_t *ostack) {
-    *ostack++ = (uintptr_t)ARRAY_DATA((Object*)NULL, void);
+#ifdef VM_FLEXARRAY
+       *ostack++ = offsetof( VMFlexArrayObject, contig_data );
+#else
+       *ostack++ = (uintptr_t)ARRAY_DATA((Object*)NULL, void);
+#endif
     return ostack;
 }
```

MMB Networks
Simply Connected

Methods: VMFlexArray

- Objects in JamVM:
  typedef struct object {
    uintptr_t lock;
    Class *class;
  } Object;
- The Class type is simply an Object with data allocated after for the ClassBlock and MethodBlock – i.e. the Class type wraps around the Object type.

https://github.com/cfriedt/jamvm/compare/array-object-modifications

29 src/interp/engine/interp.c

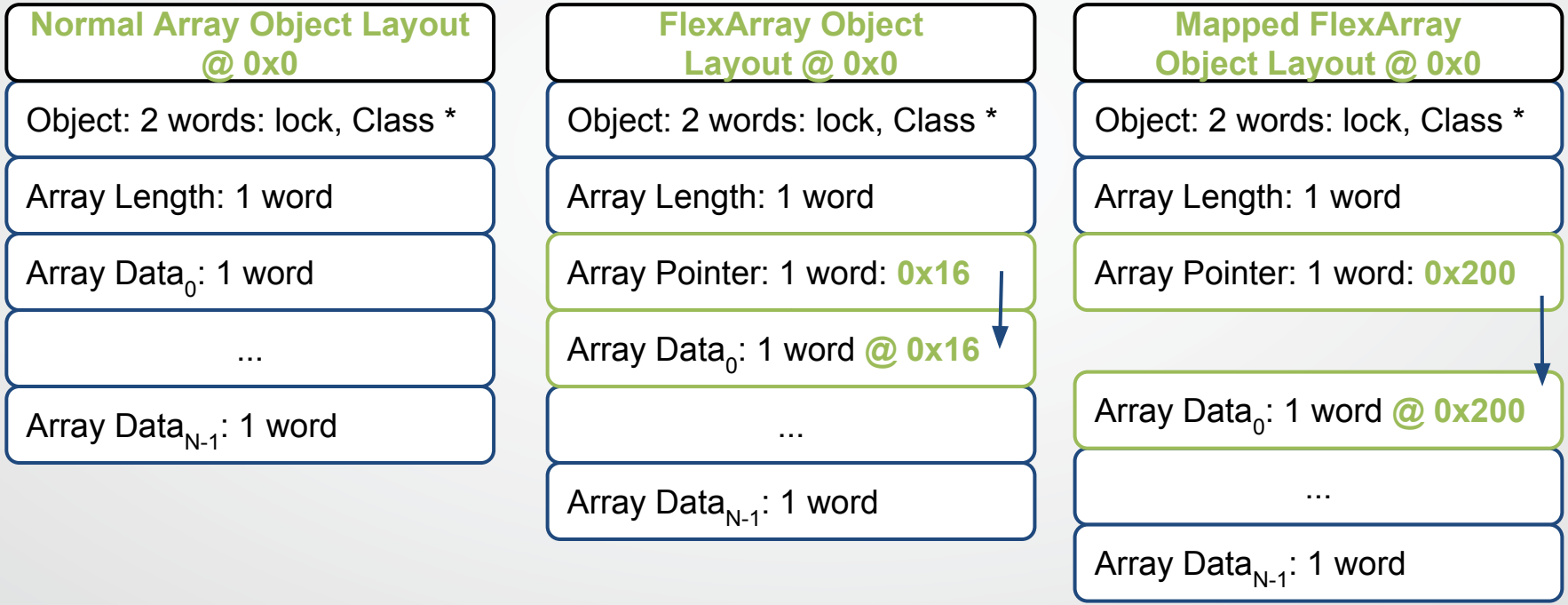
```
@@ -658,15 +658,28 @@ uintptr_t *executeJava() {
 #define ARRAY_LOAD_ARY *--ostack
 #endif

-#define ARRAY_LOAD(TYPE)                                \
-{                                                       \
-    int idx = ARRAY_LOAD_IDX;                           \
-    Object *array = (Object *)ARRAY_LOAD_ARY;           \
-                                                        \
-    NULL_POINTER_CHECK(array);                          \
-    ARRAY_BOUNDS_CHECK(array, idx);                     \
-    PUSH_0(ARRAY_DATA(array, TYPE)[idx], 1);            \
+#ifdef VM_FLEXARRAY
+#define ARRAY_LOAD(TYPE)                                \
+{                                                       \
+    int idx = ARRAY_LOAD_IDX;                           \
+    Object *array = (Object *)ARRAY_LOAD_ARY;           \
+                                                        \
+    NULL_POINTER_CHECK(array);                          \
+    NULL_POINTER_CHECK(ARRAY_DATA(array, TYPE));        \
+    ARRAY_BOUNDS_CHECK(array, idx);                     \
+    PUSH_0(ARRAY_DATA(array, TYPE)[idx], 1);            \
 }
+#else
+#define ARRAY_LOAD(TYPE)                                \
+{                                                       \
+    int idx = ARRAY_LOAD_IDX;                           \
+    Object *array = (Object *)ARRAY_LOAD_ARY;           \
+                                                        \
+    NULL_POINTER_CHECK(array);                          \
+    ARRAY_BOUNDS_CHECK(array, idx);                     \
+    PUSH_0(ARRAY_DATA(array, TYPE)[idx], 1);            \
+}
+#endif

     DEF_OPC_012_2(
             OPC_IALOAD,
```

MMB Networks
Simply Connected

Methods: VMFlexArray Example: int[]

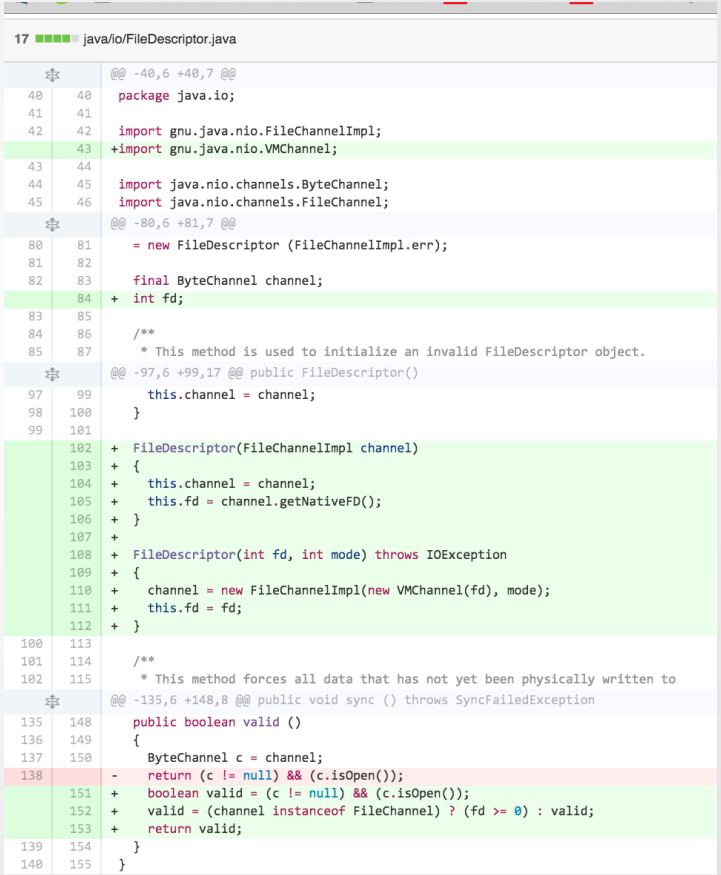


MMB Networks
Simply Connected

Methods: VMFlexArray

- Similarly, array Objects simply wrap the Object structure
- Previously, JamVM was always responsible for allocating arrays contiguously
- 1 additional dereference is performed
- A pointer is reserved directly before the array data. If array is contiguous, pointer points to next word. Otherwise, points elsewhere (e.g. mmap'd data)

https://github.com/cfriedt/jamvm/compare/array-object-modifications



17 java/io/FileDescriptor.java

```
@@ -40,6 +40,7 @@
 package java.io;

 import gnu.java.nio.FileChannelImpl;
+import gnu.java.nio.VMChannel;

 import java.nio.channels.ByteChannel;
 import java.nio.channels.FileChannel;
@@ -80,6 +81,7 @@
   = new FileDescriptor (FileChannelImpl.err);

   final ByteChannel channel;
+  int fd;

   /**
    * This method is used to initialize an invalid FileDescriptor object.
@@ -97,6 +99,17 @@ public FileDescriptor()
     this.channel = channel;
   }

+  FileDescriptor(FileChannelImpl channel)
+  {
+    this.channel = channel;
+    this.fd = channel.getNativeFD();
+  }
+
+  FileDescriptor(int fd, int mode) throws IOException
+  {
+    channel = new FileChannelImpl(new VMChannel(fd), mode);
+    this.fd = fd;
+  }

   /**
    * This method forces all data that has not yet been physically written to
@@ -135,6 +148,8 @@ public void sync () throws SyncFailedException
   public boolean valid ()
   {
     ByteChannel c = channel;
-    return (c != null) && (c.isOpen());
+    boolean valid = (c != null) && (c.isOpen());
+    valid = (channel instanceof FileChannel) ? (fd >= 0) : valid;
+    return valid;
   }
 }
```

MMB Networks
Simply Connected

Methods: Build Configuration & Running

JamVM

- ./configure --enable-vm-flexarray …

Classpath

- ./configure --enable-vm-flexarray …

Linux

- Patch 1/2 of VMWare patchset is upstream
- I believe it rolls out with current Ubuntu
- Patch 2/2 (page flip) applied manually
- Waiting for feedback from lkml

VMWare

- Download VM image
- The root password is empty
- SSH in from terminal
- modprobe vmwgfx
- fbset
  –xres 640 –yres 480
  –vxres 640 –vyres 960
- . ~/.bashrc; testfb4j

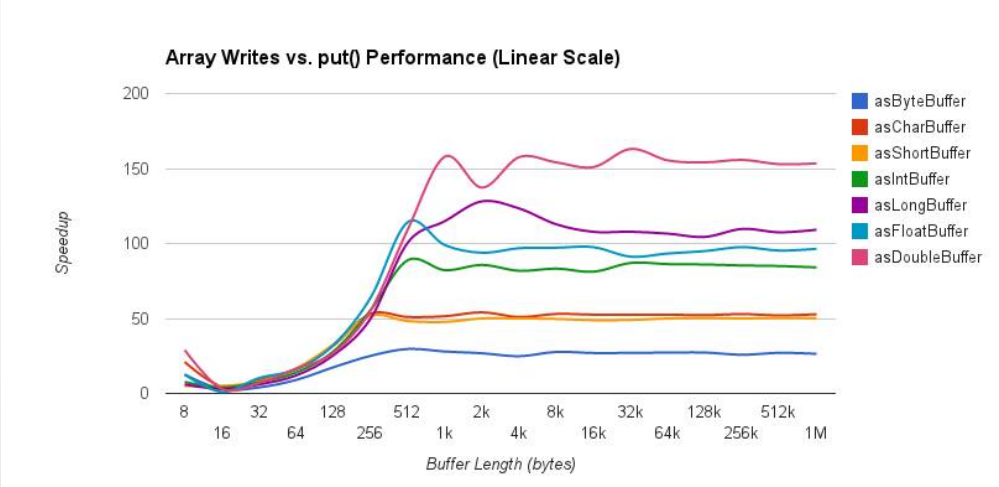
MMB Networks
Simply Connected

- History
- Hypothesis
- Apparatus
- Methods
- Observations
- Conclusions
- Questions
- Thanks

MMB Networks
Simply Connected

Observations

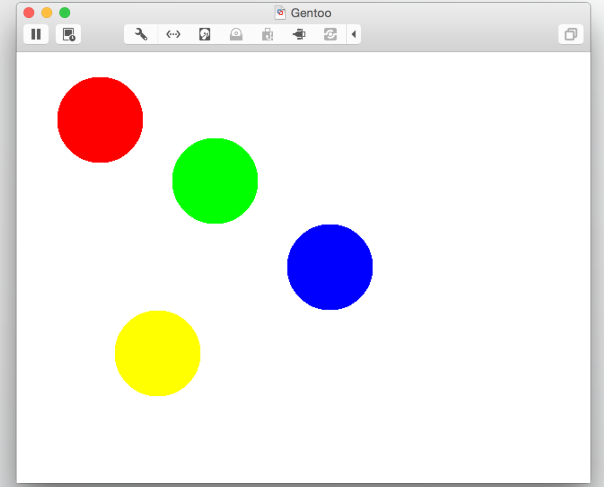
- Compared to respective Buffer.put(), using byte[] foo = MappedByteBuffer.array() and performing regular Java operations array elements speeds up Java code by a factor of up to **150x** !~1!

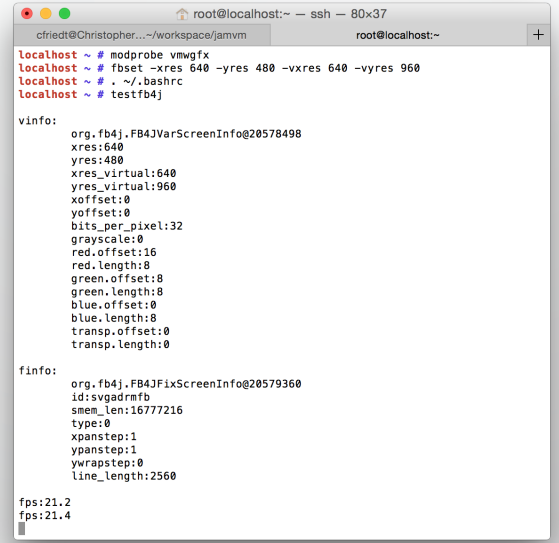


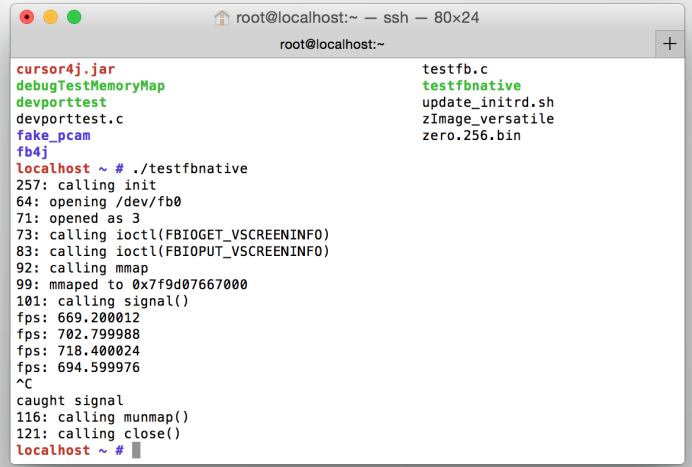
MMB Networks
Simply Connected

Observations

- With **JamVM**, the bouncing ball demo achieves **21** up to **48 fps**
- In **C**, the bouncing ball demo achieves up to **700 fps** :-/







MMB Networks
Simply Connected

History
Hypothesis
Apparatus
Methods
Observations
Conclusions
Questions
Thanks

Conclusions

- Code works end-to-end with JamVM and GNU Classpath
- **Very** interested to try VMFlexArray with e.g. other class libraries, other VMs
    - would be useful to have JDWP and Java Profiling Agent support
- Large difference between Java and C fps indicates that JamVM needs performance optimizations
    - backport Dalvik's JIT?
- Demo anyone?

MMB Networks
Simply Connected

History
Hypothesis
Apparatus
Methods
Observations
Conclusions
Thanks
Questions

Thanks!

- MMB Networks, for
    - having our monthly HackDay
    - encouraging me to do this sort of thing for fun
- Robert Lougher for writing JamVM, the Classpath developer community
- David Airlie (RedHat), and Thomas Hellström (VMWare) for
    - reviewing my patches to go upstream in the Linux Kernel
- Mario Torre an Roman Kennke
    - for organizing the Java DevRoom, speakers, etc
    - for starting the Caciocavallo project
- The Audience!

History
Hypothesis
Apparatus
Methods
Observations
Conclusions
Thanks
Questions