```
**********************************************************
    5653 Fri Dec 18 13:15:18 2015
new/usr/src/lib/common/i386/crt1.s
6507 i386 makecontext(3c) needs to 16-byte align the stack
**********************************************************
```

```
   1 /*
   2  * CDDL HEADER START
   3  *
   4  * The contents of this file are subject to the terms of the
   5  * Common Development and Distribution License (the "License").
   6  * You may not use this file except in compliance with the License.
   7  *
   8  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
   9  * or http://www.opensolaris.org/os/licensing.
  10  * See the License for the specific language governing permissions
  11  * and limitations under the License.
  12  *
  13  * When distributing Covered Code, include this CDDL HEADER in each
  14  * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
  15  * If applicable, add the following below this CDDL HEADER, with the
  16  * fields enclosed by brackets "[]" replaced with your own identifying
  17  * information: Portions Copyright [yyyy] [name of copyright owner]
  18  *
  19  * CDDL HEADER END
  20  */

  22 /*
  23  * Copyright 2009 Sun Microsystems, Inc.  All rights reserved.
  24  * Use is subject to license terms.
  25  */

  27 /*
  28  * This crt1.o module is provided as the bare minimum required to build
  29  * a 32-bit executable with gcc.  It is installed in /usr/lib
  30  * where it will be picked up by gcc, along with crti.o and crtn.o
  31  */

  33         .file   "crt1.s"

  35         .globl  _start

  37 /* global entities defined elsewhere but used here */
  38         .globl  main
  39         .globl  __fpstart
  40         .globl  exit
  41         .globl  _exit
  42         .weak   _DYNAMIC

  44         .section        .data

  46         .weak   environ
  47         .set    environ,_environ
  48         .globl  _environ
  49         .type   _environ,@object
  50         .size   _environ,4
  51         .align  4
  52 _environ:
  53         .4byte  0x0

  55         .globl  __environ_lock
  56         .type   __environ_lock,@object
  57         .size   __environ_lock,24
  58         .align  8
  59 __environ_lock:
  60         .zero   24
```



```
  62         .globl  ___Argv
  63         .type   ___Argv,@object
  64         .size   ___Argv,4
  65         .align  4
  66 ___Argv:
  67         .4byte  0x0

  69         .section        .text
  70         .align  4

  72 /*
  73  * C language startup routine.
  74  * Assume that exec code has cleared the direction flag in the TSS.
  75  * Assume that %esp is set to the addr after the last word pushed.
  76  * The stack contains (in order): argc, argv[],envp[],...
  77  * Assume that all of the segment registers are initialized.
  78  *
  79  * Allocate a NULL return address and a NULL previous %ebp as if
  80  * there was a genuine call to _start.
  81  * sdb stack trace shows _start(argc,argv[0],argv[1],...,envp[0],...)
  82  */
  83         .type   _start,@function
  84 _start:
  85         pushl   $0
  86         pushl   $0
  87         movl    %esp,%ebp               /* The first stack frame */

  89         movl    $_DYNAMIC,%eax
  90         testl   %eax,%eax
  91         jz      1f
  92         pushl   %edx                    /* register rt_do_exit */
  93         call    atexit
  94         addl    $4,%esp
  95 1:
  96         pushl   $_fini
  97         call    atexit
  98         addl    $4,%esp

 100 /*
 101  * The following code provides almost standard static destructor handling
 102  * for systems that do not have the modified atexit processing in their
 103  * system libraries.  It checks for the existence of the new routine
 104  * "_get_exit_frame_monitor()", which is in libc.so when the new exit-handling
 105  * code is there.  It then check for the existence of "__Crun::do_exit_code()"
 106  * which will be in libCrun.so whenever the code was linked with the C++
 107  * compiler.  If there is no enhanced atexit, and we do have do_exit_code,
 108  * we register the latter with atexit.  There are 5 extra slots in
 109  * atexit, so this will still be standard conforming.  Since the code
 110  * is registered after the .fini section, it runs before the library
 111  * cleanup code, leaving nothing for the calls to _do_exit_code_in_range
 112  * to handle.
 113  *
 114  * Remove this code and the associated code in libCrun when the earliest
 115  * system to be supported is Solaris 8.
 116  */
 117         .weak   _get_exit_frame_monitor
 118         .weak   __1cG__CrunMdo_exit_code6F_v_

 120         .section        .data
 121         .align  4
 122 __get_exit_frame_monitor_ptr:
 123         .4byte  _get_exit_frame_monitor
 124         .type   __get_exit_frame_monitor_ptr,@object
 125         .size   __get_exit_frame_monitor_ptr,4

 127         .align  4
```

```
__do_exit_code_ptr:
        .4byte  __1cG__CrunMdo_exit_code6F_v_
        .type   __do_exit_code_ptr,@object
        .size   __do_exit_code_ptr,4

        .section        .text

        lea     __get_exit_frame_monitor_ptr, %eax
        movl    (%eax), %eax
        testl   %eax,%eax
        jz      1f
        lea     __do_exit_code_ptr, %eax
        movl    (%eax), %eax
        testl   %eax, %eax
        jz      1f
        pushl   %eax
        call    atexit          /* atexit(__Crun::do_exit_code()) */
        addl    $4,%esp
1:

/*
 * End of destructor handling code
 */

/*
 * Calculate the location of the envp array by adding the size of
 * the argv array to the start of the argv array.
 */

        movl    8(%ebp),%eax            /* argc */
        movl    _environ, %edx          /* fixed bug 4302802 */
        testl   %edx, %edx              /* check if _enviorn==0 */
        jne     1f                      /* fixed bug 4203802 */
        leal    16(%ebp,%eax,4),%edx    /* envp */
        movl    %edx,_environ           /* copy to _environ */
1:
        /*
         * The stack needs to be 16-byte aligned with a 4-byte bias.  See
         * comment in lib/libc/i386/gen/makectxt.c.
         *
         * Note: If you change it, you need to change it in the following
         * files as well:
         *
         *  - lib/libc/i386/threads/machdep.c
         *  - lib/libc/i386/gen/makectxt.c
         *  - lib/common/i386/crti.s
         */
#endif /* ! codereview */
        andl    $-16,%esp       /* make main() and exit() be called with */
        subl    $4,%esp         /* a properly aligned stack pointer */
        subl    $4,%esp         /* a 16-byte aligned stack pointer */
        pushl   %edx
        leal    12(%ebp),%edx   /* argv */
        movl    %edx,___Argv
        pushl   %edx
        pushl   %eax            /* argc */
        call    __fpstart
        call    __fsr           /* support for ftrap/fround/fprecision  */
        call    _init
        call    main            /* main(argc,argv,envp) */
        movl    %eax,(%esp)     /* return value from main, for exit() */
        movl    %eax,4(%esp)    /* remember it for _exit(), below */
        call    exit
        movl    4(%esp),%eax    /* if user redefined exit, call _exit */
        movl    %eax,(%esp)
        call    _exit
```



```
        hlt
        .size   _start, .-_start

#include "fsr.s"

/*
 * The following is here in case any object module compiled with cc -p
 *      was linked into this module.
 */
        .section        .text
        .align  4
        .globl  _mcount
        .type   _mcount,@function
_mcount:
        ret
        .size   _mcount, .-_mcount

        .section        .data

        .globl  __longdouble_used
        .type   __longdouble_used,@object
        .size   __longdouble_used,4
        .align  4
__longdouble_used:
        .4byte  0x0
```

****
2221 Fri Dec 18 13:15:18 2015
new/usr/src/lib/common/i386/crti.s
6507 i386 makecontext(3c) needs to 16-byte align the stack
****

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License, Version 1.0 only
 * (the "License").  You may not use this file except in compliance
 * with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */
/*
 * Copyright (c) 2001 by Sun Microsystems, Inc.
 * All rights reserved.
 */
/*
 * Copyright (c) 2013, Joyent, Inc.  All rights reserved.
 */

/*
 * These crt*.o modules are provided as the bare minimum required
 * from a crt*.o for inclusion in building low level system
 * libraries.  The are only be to included in libraries which
 * contain *no* C++ code and want to avoid the startup code
 * that the C++ runtime has introduced into the crt*.o modules.
 *
 * For further details - see bug#4433015
 */

        .file   "crti.s"

/*
 * Note that when _init and _fini are called the stack needs to be 16-byte
 * aligned with a 4-byte bias.  See comment in lib/libc/i386/gen/makectxt.c.
 *
 * Note: If you change it, you need to change it in the following files as
 * well:
 *
 *  - lib/libc/i386/threads/machdep.c
 *  - lib/libc/i386/gen/makectxt.c
 *  - lib/common/i386/crt1.s
 * Note that when _init and _fini are called we have 16-byte alignment per the
 * ABI. We need to make sure that our asm leaves it such that subsequent calls
 * will be aligned. gcc expects stack alignment before the call instruction is
 * executed. Specifically if we call function foo(), the stack pointer will be
 * 0xc aligned after executing the call instruction and before executing foo's
 * prologue. Note that because 16-byte alignment also ensures 4-byte alignment
 * we will not be breaking compatibility with older applications.
 */

/*
```



```
 * _init function prologue
 */
        .section        .init,"ax"
        .globl  _init
        .type   _init,@function
        .align  16
_init:
        pushl   %ebp
        movl    %esp, %ebp
        andl    $-16,%esp
        subl    $12,%esp
        pushl   %ebx
        call    .L1
.L1:    popl    %ebx
        addl    $_GLOBAL_OFFSET_TABLE_+[.-.L1], %ebx

/*
 * _fini function prologue
 */
        .section        .fini,"ax"
        .globl  _fini
        .type   _fini,@function
        .align  16
_fini:
        pushl   %ebp
        movl    %esp, %ebp
        andl    $-16,%esp
        subl    $12,%esp
        pushl   %ebx
        call    .L2
.L2:    popl    %ebx
        addl    $_GLOBAL_OFFSET_TABLE_+[.-.L2], %ebx
```

```
**********************************************************
    3629 Fri Dec 18 13:15:18 2015
new/usr/src/lib/libc/i386/gen/makectxt.c
patch feedback
6507 i386 makecontext(3c) needs to 16-byte align the stack
**********************************************************
```

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */

/*
 * Copyright 2008 Sun Microsystems, Inc.  All rights reserved.
 * Use is subject to license terms.
 */

/*      Copyright (c) 1988 AT&T */
/*        All Rights Reserved   */

#pragma ident   "%Z%%M% %I%     %E% SMI"

#pragma weak _makecontext = makecontext

#include "lint.h"
#include <stdarg.h>
#include <ucontext.h>
#include <sys/stack.h>

/*
 * The ucontext_t that the user passes in must have been primed with a
 * call to getcontext(2), have the uc_stack member set to reflect the
 * stack which this context will use, and have the uc_link member set
 * to the context which should be resumed when this context returns.
 * When makecontext() returns, the ucontext_t will be set to run the
 * given function with the given parameters on the stack specified by
 * uc_stack, and which will return to the ucontext_t specified by uc_link.
 */

/*
 * The original i386 ABI said that the stack pointer need be only 4-byte
 * aligned before a function call (STACK_ALIGN == 4).  The ABI supplement
 * version 1.0 changed the required alignment to 16-byte for the benefit of
 * floating point code compiled using sse2.  The compiler assumes this
 * alignment and maintains it for calls it generates.  If the stack is
 * initially properly aligned, it will continue to be so aligned.  If it is
 * not initially so aligned, it will never become so aligned.
 *
 * One slightly confusing detail to keep in mind is that the 16-byte
 * alignment (%esp & 0xf == 0) is true just *before* the call instruction.
 * The call instruction will then push a return value, decrementing %esp by
```



```
 * 4.  Therefore, if one dumps %esp at the at the very first instruction in
 * a function, it will end with a 0xc.  The compiler expects this and
 * compensates for it properly.
 *
 * Note: If you change this value, you need to change it in the following
 * files as well:
 *
 *  - lib/libc/i386/threads/machdep.c
 *  - lib/common/i386/crti.s
 *  - lib/common/i386/crt1.s
 */
#undef  STACK_ALIGN
#define STACK_ALIGN     16

#endif /* ! codereview */
static void resumecontext(void);

void
makecontext(ucontext_t *ucp, void (*func)(), int argc, ...)
{
        long *sp;
        long *tsp;
        va_list ap;
        size_t size;

        ucp->uc_mcontext.gregs[EIP] = (greg_t)func;

        size = sizeof (long) * (argc + 1);

        tsp = (long *)(((uintptr_t)ucp->uc_stack.ss_sp +
        sp = (long *)(((uintptr_t)ucp->uc_stack.ss_sp +
            ucp->uc_stack.ss_size - size) & ~(STACK_ALIGN - 1));

        /*
         * Since we're emulating the call instruction, we must push the
         * return address (which involves adjusting the stack pointer to
         * have the proper 4-byte bias).
         */
        sp = tsp - 1;

        *sp = (long)resumecontext;              /* return address */

        ucp->uc_mcontext.gregs[UESP] = (greg_t)sp;
        tsp = sp + 1;

        /*
         * "push" all the arguments
         */
#endif /* ! codereview */
        va_start(ap, argc);
        while (argc-- > 0)

        while (argc-- > 0) {
                *tsp++ = va_arg(ap, long);
        }

        va_end(ap);

        *sp = (long)resumecontext;              /* return address */

        ucp->uc_mcontext.gregs[UESP] = (greg_t)sp;
}
```

_____*unchanged_portion_omitted_*

****
7123 Fri Dec 18 13:15:18 2015
new/usr/src/lib/libc/i386/threads/machdep.c
6507 i386 makecontext(3c) needs to 16-byte align the stack
****

```
/*
 * CDDL HEADER START
 *
 * The contents of this file are subject to the terms of the
 * Common Development and Distribution License (the "License").
 * You may not use this file except in compliance with the License.
 *
 * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
 * or http://www.opensolaris.org/os/licensing.
 * See the License for the specific language governing permissions
 * and limitations under the License.
 *
 * When distributing Covered Code, include this CDDL HEADER in each
 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
 * If applicable, add the following below this CDDL HEADER, with the
 * fields enclosed by brackets "[]" replaced with your own identifying
 * information: Portions Copyright [yyyy] [name of copyright owner]
 *
 * CDDL HEADER END
 */

/*
 * Copyright (c) 1999, 2010, Oracle and/or its affiliates. All rights reserved.
 */

#include "thr_uberdata.h"
#include <procfs.h>
#include <ucontext.h>
#include <setjmp.h>

/*
 * The stack needs to be 16-byte aligned with a 4-byte bias.  See comment in
 * lib/libc/i386/gen/makectxt.c.
 *
 * Note: If you change it, you need to change it in the following files as
 * well:
 *
 *  - lib/libc/i386/gen/makectxt.c
 *  - lib/common/i386/crti.s
 *  - lib/common/i386/crt1.s
 * The i386 ABI says that the stack pointer need be only 4-byte aligned
 * before a function call (STACK_ALIGN == 4).  We use a 16-byte stack
 * alignment for the benefit of floating point code compiled using sse2.
 * Even though the i386 ABI doesn't require it, both cc and gcc
 * assume this alignment on entry to a function and maintain it
 * for calls made from that function.  If the stack is initially
 * aligned on a 16-byte boundary, it will continue to be so aligned.
 * If it is not initially so aligned, it will never become so aligned.
 */
#undef  STACK_ALIGN
#define STACK_ALIGN     16

extern int getlwpstatus(thread_t, lwpstatus_t *);
extern int putlwpregs(thread_t, prgregset_t);

void *
setup_top_frame(void *stk, size_t stksize, ulwp_t *ulwp)
{
        uint32_t *stack;
        struct {
                uint32_t        rpc;
```



```
                uint32_t        arg;
                uint32_t        pad;
                uint32_t        fp;
                uint32_t        pc;
        } frame;

        /*
         * Top-of-stack must be rounded down to STACK_ALIGN and
         * there must be a minimum frame.  Note: 'frame' is not a true
         * stack frame (see <sys/frame.h>) but a construction made here to
         * make it look like _lwp_start called the thread start function
         * with a 16-byte aligned stack pointer (the address of frame.arg
         * is the address that muet be aligned on a 16-byte boundary).
         */
        stack = (uint32_t *)(((uintptr_t)stk + stksize) & ~(STACK_ALIGN-1));

        /*
         * This will return NULL if the kernel cannot allocate
         * a page for the top page of the stack.  This will cause
         * thr_create(), pthread_create() or pthread_attr_setstack()
         * to fail, passing the problem up to the application.
         */
        stack -= 5;     /* make the address of frame.arg be 16-byte aligned */
        frame.pc = 0;
        frame.fp = 0;   /* initial address for %ebp (see EBP below) */
        frame.pad = 0;
        frame.arg = (uint32_t)ulwp;
        frame.rpc = (uint32_t)_lwp_start;
        if (uucopy(&frame, (void *)stack, sizeof (frame)) == 0)
                return (stack);
        return (NULL);
}
```

_____***unchanged_portion_omitted_***