Sample Programs A

This appendix provides sample programs that show you how to:

- Check profile calibration (`kcms_update.c`)
- Test loading in a scanner profile and a monitor profile, and color correct the image data (`kcms_test.c`)
- Print header attributes in a profile (`print_header.c`)

Also provided in this appendix are the following files. These files are located on-line and are used in the sample programs.

- `kcms_create.c`
- `kcms_utils.c`
- `kcms_utils.h`

A

Check Profile Calibration

This sample code shows you how measurement data updates a profile. First, the data file, `lum_out` (monitor luminance output data) is read into the KCMS measurement structure, `KcsCalibrationData` (defined in `kcstypes.h`). Next, the input profile (argument1) is opened, its white point is changed with `KcsSetAttribute`, the profile is updated with the new luminance measurement data tables, and then the profile is saved.

Code Example A-1 `kcms_update.c`

```
#pragma ident "@(#)kcms_update.c1.16 13 Dec 1994"
/* Copyright (c) 1994, by Sun Microsystems, Inc.
 * All rights reserved.
 */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <math.h>
#include <sys/types.h>
#include <time.h>
#include <X11/X.h>
#include <kcms/kcs.h>
#include <kcms/icc.h>
#include <kcms/kcstypes.h>
#include "kcms_utils.h"

/* #define FILE_DESC */
/* #define X11_SAVE */

float      Luminance_float_out[3][256];

/* test code to check profile calibration */
main(int argc, char **argv)
{
    KcsCalibrationData    *calData;
    KcsProfileDesc        x_desc, desc;
    KcsProfileId          profileid;
    KcsStatusId           status;
    KcsAttributeValue     *attrValuePtr;
    KcsErrDesc            errDesc;
    int                   levels = 256, channels = 3;
    int                   sizemeas, nvalues, i, j, fd, sfd;
    FILE                  *simfile;
```

Code Example A-1 `kcms_update.c` *(Continued)*

```
    float                   input_val;
    size_t                  rc;
    time_t                  clocktime;
    struct tm               *datetime;
    icXYZNumber             *XYZval;
    Display                 *display;
    void                    print_curves(int, int);
    double                  icfixed2double(long val, long type);

    display = NULL;
#ifdef undef
    /* Read  the measured calibration data from a file */
    if ((simfile = fopen("lum_out", "r")) == NULL) {
        fprintf(stderr,"cannot open output luminance file\n");
        exit(1);
    }
    for (i=0; i<channels; i++)
        for (j=0; j<levels;  j++)
            Luminance_float_out[i][j] = 0.0;
    nvalues = levels * channels;
    rc = fread(Luminance_float_out, sizeof(float), nvalues, simfile);
    fclose(simfile);
#endif undef

    for (j=0; j<levels; j++) {
        input_val = j * (1.0/255.0);
    Luminance_float_out[0][j] = pow(input_val, 2.22);
    Luminance_float_out[1][j] = pow(input_val, 2.22);
    Luminance_float_out[2][j] = pow(input_val, 2.22);
    }

    /* This routine prints the curves to a file */
    print_curves(channels, levels);

    /* Fill out the measurement structures */
    sizemeas = (int) (sizeof(KcsMeasurementBase) + sizeof(long) +
                sizeof(KcsMeasurementSample) * levels);
    calData = (KcsCalibrationData *) malloc(sizemeas);
    calData->base.countSupplied = levels;
    calData->base.numInComp = 3;
    calData->base.numOutComp = 3;
    calData->base.inputSpace = KcsRGB;
    calData->base.outputSpace = KcsRGB;
```

Code Example A-1 `kcms_update.c` *(Continued)*

```
    for (i=0; i< levels; i++) {
        calData->val.patch[i].weight = 1.0;
        calData->val.patch[i].standardDeviation = 0.0;
        calData->val.patch[i].sampleType = KcsChromatic;
        calData->val.patch[i].input[KcsRGB_R] = (float)i/255;
        calData->val.patch[i].input[KcsRGB_G] = (float)i/255;
        calData->val.patch[i].input[KcsRGB_B] = (float)i/255;
        calData->val.patch[i].input[3] = 0.0;

        calData->val.patch[i].output[KcsRGB_R] = Luminance_float_out[0][i];
        calData->val.patch[i].output[KcsRGB_G] = Luminance_float_out[1][i];
        calData->val.patch[i].output[KcsRGB_B] = Luminance_float_out[2][i];
        calData->val.patch[i].output[3] = 0.0;
    }

    calData->val.patch[0].sampleType = KcsBlack;
    calData->val.patch[255].sampleType = KcsWhite;

    if (!argv[1]) {
        fprintf(stderr, "Usage kcs_update profile_in [profile_out]\n");
        exit(1);
    }

#ifdef FILE_DESC
    /* Open the file and use a file descriptor */
    if ((fd = open(argv[1], O_RDWR)) == -1) {
        perror ("open failed");
        exit (1);
    }
    x_desc.type = KcsFileProfile;
    x_desc.desc.desc.file.openFileId = fd;
    x_desc.desc.file.offset = 0;
#else
    /* Let the library open the file */
    x_desc.type = KcsSolarisProfile;
    x_desc.desc.solarisFile.fileName = argv[1];
    x_desc.desc.solarisFile.hostName = NULL;
    x_desc.desc.solarisFile.oflag = O_RDWR;
    x_desc.desc.solarisFile.mode = 0;
#endif FILE_DESC

    status = KcsLoadProfile(&profileid, &x_desc, KcsLoadAllNow);
    if(status != KCS_SUCCESS) {
        status = KcsGetLastError(&errDesc);
```

Code Example A-1 `kcms_update.c` *(Continued)*

```
        printf("LoadProfile error: %s\n", errDesc.desc);
        exit(1);
    }

#ifdef undef
    /* Save to an X11 profile */
    if ((display = XOpenDisplay(NULL)) == NULL) {
        fprintf(stderr, "Couldn't open display\n");
        KcsFreeProfile(profileid);
    }
    desc.type = KcsWindowProfile;
    desc.desc.xwin.dpy = display;
    desc.desc.xwin.visual = DefaultVisual(display, DefaultScreen(display));
    desc.desc.xwin.screen = DefaultScreen(display);
    status = KcsSaveProfile(profileid, &desc);
    if(status != KCS_SUCCESS) {
        status = KcsGetLastError(&errDesc);
        printf("LoadProfile error: %s\n", errDesc.desc);
        exit(1);
    }
    KcsFreeProfile(profileid);
    status = KcsLoadProfile(&profileid, &desc, KcsLoadAllNow);
    if(status != KCS_SUCCESS) {
        status = KcsGetLastError(&errDesc);
        printf("LoadProfile error: %s\n", errDesc.desc);
        exit(1);
    }
#endif undef

    /* See if we can read back the original gamma values */
    attrValuePtr = (KcsAttributeValue *)malloc(sizeof(KcsAttributeBase)
                + sizeof(icUInt32Number) + (256 * sizeof(icUInt16Number)));
    attrValuePtr->base.countSupplied = 1;

    status = KcsGetAttribute(profileid, icSigRedTRCTag, attrValuePtr);
#ifdef undef
    if (status == KCS_SUCCESS) {
        for (i=0; i<attrValuePtr->val.icCurve.count; i++)
            printf("%d\n", attrValuePtr->val.icCurve.data[i]);
    } else
        printf("Original Monitor calibration LUT Not present\n");
#endif undef
    free(attrValuePtr);
    attrValuePtr = NULL;
```

Code Example A-1 `kcms_update.c` *(Continued)*

```
    attrValuePtr = (KcsAttributeValue *)malloc(sizeof(KcsAttributeBase)
                      + sizeof(icXYZNumber) );
    attrValuePtr->base.type = icSigXYZArrayType;
    attrValuePtr->base.countSupplied = 1;
    attrValuePtr->base.sizeOfType = sizeof(icXYZNumber);
    status = KcsGetAttribute(profileid, icSigMediaWhitePointTag, attrValuePtr);
    if(status != KCS_SUCCESS) {
        status = KcsGetLastError(&errDesc);
        printf("GetAttribute error icSigMediaWhitePoint : %s\n", errDesc.desc);
    } else {
        XYZval = (icXYZNumber *)attrValuePtr->val.icXYZ.data;
        printf("Media White Point X=%f Y=%f Z=%f\n",
                icfixed2double(XYZval->X, icSigS15Fixed16ArrayType),
                icfixed2double(XYZval->Y, icSigS15Fixed16ArrayType),
                icfixed2double(XYZval->Z, icSigS15Fixed16ArrayType));
        free(attrValuePtr);
        attrValuePtr = NULL;
    }

    /* Get the colorants */
    /* Red */
    attrValuePtr = (KcsAttributeValue *)malloc(sizeof(KcsAttributeBase) +
                      sizeof(icXYZNumber) );
    attrValuePtr->base.type = icSigXYZArrayType;
    attrValuePtr->base.countSupplied = 1;
    status = KcsGetAttribute(profileid, icSigRedColorantTag, attrValuePtr);
    if(status != KCS_SUCCESS) {
        status = KcsGetLastError(&errDesc);
        printf("GetAttribute error: %s\n", errDesc.desc);
        KcsFreeProfile(profileid);
        exit(1);
    }

    XYZval = (icXYZNumber *)attrValuePtr->val.icXYZ.data;
    printf("Red X=%f Y=%f Z=%f\n", icfixed2double(XYZval->X,
            icSigS15Fixed16ArrayType), icfixed2double(XYZval->Y,
            icSigS15Fixed16ArrayType), icfixed2double(XYZval->Z,
            icSigS15Fixed16ArrayType));

    /* Green */
    status = KcsGetAttribute(profileid, icSigGreenColorantTag, attrValuePtr);
    if(status != KCS_SUCCESS) {
        status = KcsGetLastError(&errDesc);
```

Code Example A-1 `kcms_update.c` *(Continued)*

```
        printf("SetAttribute error: %s\n", errDesc.desc);
        KcsFreeProfile(profileid);
        exit(1);
    }

    XYZval = (icXYZNumber *)attrValuePtr->val.icXYZ.data;
    printf("Green X=%f Y=%f Z=%f\n", icfixed2double(XYZval->X,
            icSigS15Fixed16ArrayType), icfixed2double(XYZval->Y,
            icSigS15Fixed16ArrayType), icfixed2double(XYZval->Z,
            icSigS15Fixed16ArrayType));

    /* Blue */
    status = KcsGetAttribute(profileid, icSigBlueColorantTag, attrValuePtr);
    if(status != KCS_SUCCESS) {
        status = KcsGetLastError(&errDesc);
        printf("SetAttribute error: %s\n", errDesc.desc);
        KcsFreeProfile(profileid);
        exit(1);
    }

    XYZval = (icXYZNumber *)attrValuePtr->val.icXYZ.data;
    printf("Blue X=%f Y=%f Z=%f\n", icfixed2double(XYZval->X,
            icSigS15Fixed16ArrayType), icfixed2double(XYZval->Y,
            icSigS15Fixed16ArrayType), icfixed2double(XYZval->Z,
            icSigS15Fixed16ArrayType));
    free(attrValuePtr);

    status = KcsUpdateProfile(profileid, NULL, calData, NULL);
    if(status != KCS_SUCCESS) {
        status = KcsGetLastError(&errDesc);
        printf("UpdateProfile error: %s\n", errDesc.desc);
        KcsFreeProfile(profileid);
        exit(1);
    }
    free(calData);
    /* update calibrate time stamp - save this example - not easy! */
    /* Get the time from the system */
    clocktime = time(NULL);
    datetime = localtime(&clocktime);

    /* Create space for the tag */
    attrValuePtr = (KcsAttributeValue *)malloc(sizeof(KcsAttributeBase) +
                   sizeof(icDateTimeNumber) );
    attrValuePtr->base.type = icSigDateTimeType;
```

Code Example A-1 `kcms_update.c` *(Continued)*

```
    attrValuePtr->base.countSupplied = 1;

    attrValuePtr->val.icDateTime.seconds = (icUInt16Number)datetime->tm_sec;
    attrValuePtr->val.icDateTime.minutes = (icUInt16Number)datetime->tm_min;
    attrValuePtr->val.icDateTime.hours = (icUInt16Number)datetime->tm_hour;
    attrValuePtr->val.icDateTime.day = (icUInt16Number)datetime->tm_mday;
    attrValuePtr->val.icDateTime.month = (icUInt16Number)datetime->tm_mon;
    attrValuePtr->val.icDateTime.year = (icUInt16Number)datetime->tm_year;
    status = KcsSetAttribute(profileid, icSigCalibrationDateTimeTag, attrValuePtr);
    if(status != KCS_SUCCESS) {
        status = KcsGetLastError(&errDesc);
        printf("SetAttribute error: %s\n", errDesc.desc);
        KcsFreeProfile(profileid);
        exit(1);
    }

    /* Test to see if we can read back the new gamma values */
    attrValuePtr = (KcsAttributeValue *)malloc(sizeof(KcsAttributeBase) +
                   sizeof(icUInt32Number) + (256 * sizeof(icUInt16Number)));
    attrValuePtr->base.countSupplied = 1;

    status = KcsGetAttribute(profileid, icSigRedTRCTag, attrValuePtr);
#ifdef undef
    if (status == KCS_SUCCESS) {
        for (i=0; i<attrValuePtr->val.icCurve.count; i++)
            printf("%d\n", attrValuePtr->val.icCurve.data[i]);
    } else
        printf("Monitor calibration LUT Not present\n");
#endif undef
    free(attrValuePtr);

    if (argv[2]) {     is this extra??????
        /* Save to an output file */
#ifdef FILE_DESC
    if ((sfd = open(argv[2], O_RDWR|O_CREAT, 0666)) == -1) {
        perror ("save open failed");
        exit (1);
    }
    desc.type = KcsFileProfile;
    desc.desc.file.openFileId = sfd;
    desc.desc.file.offset = 0;
#else
#ifdef X11_SAVE
    if ((display = XOpenDisplay(NULL)) == NULL) {
```

Code Example A-1 `kcms_update.c` *(Continued)*

```
            fprintf(stderr, "Couldn't open display\n");
            KcsFreeProfile(profileid);
    }
    desc.type = KcsWindowProfile;
    desc.desc.xwin.dpy = display;
    desc.desc.xwin.visual = DefaultVisual(display, DefaultScreen(display));
    desc.desc.xwin.screen = DefaultScreen(display);
#else
    desc.type = KcsSolarisProfile;
    desc.desc.solarisFile.fileName = argv[2];
    desc.desc.solarisFile.hostName = NULL;
    desc.desc.solarisFile.oflag = O_RDWR|O_CREAT|O_TRUNC;
    desc.desc.solarisFile.mode = 0666;
#endif X11_SAVE
#endif FILE_DESC

    status = KcsSaveProfile(profileid, &desc);
    if(status != KCS_SUCCESS) {
        status = KcsGetLastError(&errDesc);
        printf("SaveProfile error: %s\n", errDesc.desc);
    }
    KcsFreeProfile(profileid);
#ifdef FILE_DESC
    close (sfd);
#endif FILE_DESC

    } else {
        /* Just save to the same description */
        status = KcsSaveProfile(profileid, NULL);
        if(status != KCS_SUCCESS) {
            status = KcsGetLastError(&errDesc);
            printf("SaveProfile error: %s\n", errDesc.desc);
        }
        KcsFreeProfile(profileid);
    }

#ifdef FILE_DESC
    close (fd);
#endif FILE_DESC
            exit(0);
    }
```

Code Example A-1 kcms_update.c *(Continued)*

```
void
print_curves(int channels, int levels)
{
    FILE*   simfile;
    int     i, j;

    if ((simfile = fopen("graph_lum", "w+")) == NULL) {
        fprintf(stderr,"cannot open output luminance file\n");
        exit(1);
    }

    for (i=0; i<levels; i++) {
        fprintf(simfile, "%f %f %f\n", Luminance_float_out[0][i],
        Luminance_float_out[1][i], Luminance_float_out[2][i]);
    }

    fclose(simfile);
}
```

Test Loading a Profile and Color Correction

This sample code shows you how to load two profiles, connect the profiles and then save the result. The first profile (argument1) is opened, then the second profile (argument2) is opened, then the two profiles are connected and the result is saved into a third profile (argument3). For example, a monitor profile is connected to a printer profile and stored in a resultant profile used to convert numerous files that need to be printed.

Code Example A-2 kcms_test.c

```
#pragma ident "@(#)kcstest.c1.2 11 Apr 1994"
/*
 * Program tests loading in a scanner profile and monitor profile and then
* color correcting the image data. */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <memory.h>
#include <unistd.h>
#include <fcntl.h>
#include <errno.h>

#include "kcs.h" <kcms/kcs.h>
```

Code Example A-2 `kcms_test.c` *(Continued)*

```
#include "kcsattrb.h" <kcms/kcs.h>

#define FILE_NAME

main(int argc, char **argv)
{
KcsFileId               scannerFd, monitorFd;
KcsProfileDesc          scannerDesc, monitorDesc, completeDesc;
KcsProfileId            scannerProfile, monitorProfile;
KcsProfileId            profileSequence[2], completeProfile;
KcsStatusId             status;
KcsAttributeValue       attrValue;
KcsAttributeName        i;
KcsOperationType        op = (KcsOpForward+KcsContImage);
KcsErrDesc              errDesc;
u_long                  failedProfileNum;

if (argc > 4) {
    fprintf(stderr,"Usage : kcstest profile_1 profile_2 [save_profile]\n");
    exit(1);
}

#ifdef FILE_DESC
    /* Open up the files from disk */
    scannerDesc.type = KcsFileProfile;
    scannerFd = open(argv[1], O_RDONLY);
    if (scannerFd == -1) {
        perror("Failed to open scanner profile");
        exit(1);
    }
    scannerDesc.desc.file.openFileId = scannerFd;

    monitorDesc.type = KcsFileProfile;
    monitorFd = open(argv[2], O_RDONLY);
    if (monitorFd == -1) {
        perror("Failed to open monitor profile");
        exit(1);
    }
    monitorDesc.desc.file.openFileId = monitorFd;
    monitorDesc.desc.file.offset = 0;
#endif

#ifdef FILE_NAME
    scannerDesc.type = KcsSolarisProfile;
```

Code Example A-2 `kcms_test.c` *(Continued)*

```
    scannerDesc.desc.solarisFile.fileName = argv[1];
    scannerDesc.desc.solarisFile.hostName = NULL;
    scannerDesc.desc.solarisFile.oflag = O_RDONLY;
    scannerDesc.desc.solarisFile.mode = 0;

    monitorDesc.type = KcsSolarisProfile;
    monitorDesc.desc.solarisFile.fileName = argv[2];
    monitorDesc.desc.solarisFile.hostName = NULL;
    monitorDesc.desc.solarisFile.oflag = O_RDONLY;
    monitorDesc.desc.solarisFile.mode = 0;
#endif

/* Load the profiles */
printf("Load scanner profile\n");
status = KcsLoadProfile(&scannerProfile, &scannerDesc, KcsLoadAllNow);
 if (status != KCS_SUCCESS) {
    KcsGetLastError(&errDesc);
    printf("Scanner LoadProfile error: %s\n", errDesc.desc);

#ifdef FILE_DESC
    close(scannerFd);
    close(monitorFd);
#endif
    exit(1);
}

printf("Load monitor profile\n");
status = KcsLoadProfile(&monitorProfile, &monitorDesc, KcsLoadAllNow);
if (status != KCS_SUCCESS) {
    KcsGetLastError(&errDesc);
    printf("Monitor LoadProfile error: %s\n", errDesc.desc);

#ifdef FILE_DESC
    close(scannerFd);
    close(monitorFd);
#endif
    exit(1);
}

/* See if we can combine them */
profileSequence[0] = scannerProfile;
profileSequence[1] = monitorProfile;
printf("Make complete profile\n");
status = KcsConnectProfiles(&completeProfile, 2, profileSequence, op,
```

Code Example A-2 `kcms_test.c` *(Continued)*

```
            &failedProfileNum);
if (status != KCS_SUCCESS) {
    KcsGetLastError(&errDesc);
    printf("ConnectProfile error: %s\n", errDesc.desc);
    fprintf(stderr, "Failed in profile number %d\n", failedProfileNum);

#ifdef FILE_DESC
    close(scannerFd);
    close(monitorFd);
#endif
    exit(1);
}

/* Try to save the profile we just created if their is a 4th arg*/
if (argv[3]) {
    completeDesc.type = KcsSolarisProfile;
    completeDesc.desc.solarisFile.fileName = argv[3];
    completeDesc.desc.solarisFile.hostName = NULL;
    completeDesc.desc.solarisFile.oflag = O_RDWR|O_CREAT|O_TRUNC;
    completeDesc.desc.solarisFile.mode = 0666;
    status = KcsSaveProfile(completeProfile, &completeDesc);
    if (status != KCS_SUCCESS) {
           KcsGetLastError(&errDesc);
           printf("SaveProfile error: %s\n", errDesc.desc);

#ifdef FILE_DESC
     close(scannerFd);
     close(monitorFd);
#endif
     exit(1);
    }
}
KcsFreeProfile(completeProfile);
KcsFreeProfile(scannerProfile);
KcsFreeProfile(monitorProfile);

#ifdef FILE_DESC
    close(scannerFd);
    close(monitorFd);
#endif FILE_DESC
}
```

A

Print File Header

This sample code prints the ICC header in a profile. The arguments are defined as:

`-x` Library looks for the default visual profile. User does not need to supply a profile name.

`-n` Supply a partial filename. The library looks in the right directories for the profile.

`-d` Supply full path name of profile. User opens and closes file. It then opens and loads the profile and looping through all available attributes, prints out an attribute name and value.

Code Example A-3 `print_header.c`

```
/* Copyright (c) 1994, by Sun Microsystems, Inc.
 * All rights reserved.
 */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <math.h>
#include <kcms/kcs.h>
#include <kcms/kcstypes.h>
#include <kcms/icc.h>
#include "./kcms_utils.h"

/* Change a profiles description
 */
main(int argc, char **argv)
{
    KcsProfileDesc        x_desc;
    KcsProfileId          profileid;
    KcsStatusId           status;
    KcsAttributeValue     *attrValue;
    KcsErrDesc            errDesc;
    int                   size;

    void print_header(icHeader *hdr);

    if (!argv[1]) {
            fprintf(stderr, "Usage : print_header profile\n");
            exit(1);
    }
```

Code Example A-3 `print_header.c` *(Continued)*

```
    /* Let the library open the file */
    x_desc.type = KcsSolarisProfile;
    x_desc.desc.solarisFile.fileName = argv[1];
    x_desc.desc.solarisFile.hostName = NULL;
    x_desc.desc.solarisFile.oflag = O_RDONLY;
    x_desc.desc.solarisFile.mode = 0;

    status = KcsLoadProfile(&profileid, &x_desc, KcsLoadAllNow);
    if(status != KCS_SUCCESS) {
            status = KcsGetLastError(&errDesc);
            printf("LoadProfile error: %s\n", errDesc.desc);
            exit(1);
    }

    size = sizeof(KcsAttributeBase) + sizeof(icHeader);
    attrValue = (KcsAttributeValue *)malloc(size);

    /* Get the header */
    attrValue->base.type = icSigHeaderType;
    attrValue->base.sizeOfType = sizeof(icHeader);
    attrValue->base.countSupplied = 1;
    KcsGetAttribute(profileid, icSigHeaderTag, attrValue);

    print_header(&attrValue->val.icHeader);
    free(attrValue);
    KcsFreeProfile(profileid);

    exit(0);
}

void
print_header(icHeader *hdr)
{
    char       charstring[5];

    printf("Size in bytes = %d\n", hdr->size);
    printf("CMM Id = 0x%x\n", hdr->cmmId);
    printf("Major version number = 0x%x\n", hdr->version>>24);
    printf("Minor version number = 0x%x\n",
               (hdr->version&0x00FF0000)>>16);

    switch(hdr->deviceClass) {
    case icSigInputClass :
```

Code Example A-3 `print_header.c` *(Continued)*

```
            printf("deviceClass = input\n");
            break;
    case icSigDisplayClass :
            printf("deviceClass = display\n");
            break;
    case icSigOutputClass :
            printf("deviceClass = output\n");
            break;
    case icSigLinkClass :
            printf("deviceClass = link\n");
            break;
    case icSigAbstractClass :
            printf("deviceClass = abstract\n");
            break;
    case icSigColorSpaceClass :
            printf("deviceClass = colorspace\n");
            break;
    default :
            printf("Unknown\n");
            break;
    }

    memset(charstring, 0 ,5);
    memcpy(charstring, &hdr->colorSpace, 4);
    printf("colorspace = %s\n", charstring);

    memset(charstring, 0 ,5);
    memcpy(charstring, &hdr->pcs, 4);
    printf("profile connection space = %s\n", charstring);

    printf("date = %d/%d/%d,  ", hdr->date.day,hdr->date.month,
                hdr->date.year);
    printf("time = %d:%d:%d\n", hdr->date.hours,
                hdr->date.minutes,hdr->date.seconds);

    memset(charstring, 0 ,5);
    memcpy(charstring, &hdr->magic, 4);
    printf("magic number = %s\n", charstring);

    switch(hdr->platform) {
    case icSigMacintosh :
            printf("platform = Macintosh\n");
            break;
    case icSigMicrosoft :
```

Code Example A-3 `print_header.c` *(Continued)*

```
        printf("platform = Microsoft\n");
        break;
case icSigSolaris :
        printf("platform = Solaris\n");
        break;
case icSigSGI :
        printf("platform = SGI\n");
        break;
case icSigTaligent :
        printf("platform = Taligent\n");
        break;
default :
        printf("Unknown\n");
        break;
}

if(hdr->flags && icEmbeddedProfileTrue)
        printf("Embedded profile.\n");
else
        printf("Non-embedded profile\n");

if(hdr->flags && icUseWithEmbeddedDataOnly)
    printf("If this profile is embedded, it is not allowed to strip
                it out and use it independently.\n");
else
        printf("OK to strip embedded profile out and use
                independently\n");

memset(charstring, 0 ,5);
memcpy(charstring, &hdr->manufacturer, 4);
printf("manufacturer = %s\n", charstring);

printf("model number = %d\n", hdr->model);

printf("device attributes = %d%d\n", hdr->attributes[0],
                hdr->attributes[1]);


switch (hdr->renderingIntent) {
case 0 :
        printf("rendering intent = Perceptual\n");
        break;
case 1 :
        printf("rendering intent = Relative Colorimetric\n");
```

Code Example A-3 `print_header.c` *(Continued)*

```
            break;
    case 2 :
            printf("rendering intent = Saturation\n");
            break;
    case 3 :
            printf("rendering intent = Absolute Colorimetric\n");
            break;
    default :
            printf("Unknown\n");
            break;
    }

    printf("Illuminat X=%f Y=%f X=%f\n",
            icfixed2double(hdr->illuminant.X,
icSigS15Fixed16ArrayType),
            icfixed2double(hdr->illuminant.Y,
icSigS15Fixed16ArrayType),
            icfixed2double(hdr->illuminant.Z,
                        icSigS15Fixed16ArrayType));
}
```

Miscellaneous Include Files

The following `.c` and `.h` files are used in the previous sample programs. They are located online also.

A

kcms_create.c

Code Example A-4 kcms_create.c

```
#pragma ident "@(#)kcms_create.c1.12  09 Jan 1995"
/* Copyright (c) 1994, by Sun Microsystems, Inc.
 * All rights reserved.
 */
#include <stdio.h>
#include <fcntl.h>
#include <sys/isa_defs.h>
#include <kcms/kcs.h>
#include <kcms/kcsstats.h>
#include <kcms/kcstypes.h>
#include <kcms/kcsattrb.h>
#include <kcms/icc.h>
#include "kcms_utils.h"

#define SAMPLE_WORD"A profile created using kcms_create"

/* test code to check profile creation */

void print_header(icHeader *hdr);

main(int argc, char **argv)
{
    KcsCalibrationData    *calData;
    KcsProfileDesc        desc;
    KcsCreationDesc       c_desc;
    KcsProfileId          profileid;
    KcsStatusId           status;
    KcsAttributeValue     attrValue;
    KcsAttributeValue     *attrValue2;
     KcsAttributeValue    *attrValuePtr;
     KcsErrDesc           errDesc;
     int                  levels = 256, channels = 3;
     int                  sizemeas, size, nvalues, i, j;
     time_t               clocktime;
     struct tm            *datetime;
     FILE                 *simfile;
     float                Luminance_float_out[3][256];
     size_t               rc;
     char                 *description;
     char                 attr[256];
     KcsAttrSpace         spacetype;
     KcsAttributeIllumType illumtype;
```

Code Example A-4 `kcms_create.c` *(Continued)*

```
double                  test_double[3];

/* Read in the measured calibration data from a file */
if ((simfile = fopen("lum_out", "r")) == NULL) {
fprintf(stderr,"cannot open output luminance file\n");
exit(1);
}

for (i=0; i<channels; i++)
for (j=0; j<levels;  j++)
    Luminance_float_out[i][j] = 0.0;

nvalues = levels * channels;
rc = fread(Luminance_float_out, sizeof(float), nvalues, simfile);
fclose(simfile);

/* Fill out the measurement structures - The illuminant must be D50 */
test_double[0] = 0.9642;
test_double[1] = 1.0;
test_double[2] = 0.8249;
sizemeas = (int) (sizeof(KcsMeasurementBase) + sizeof(long) +
      sizeof(KcsMeasurementSample) * levels);
calData = (KcsCalibrationData *) malloc(sizemeas);

calData->base.countSupplied = levels;
calData->base.numInComp = 3;
calData->base.numOutComp = 3;
calData->base.inputSpace = KcsRGB;
calData->base.outputSpace = KcsRGB;
for (i=0; i< levels; i++) {
calData->val.patch[i].weight = 1.0;
calData->val.patch[i].standardDeviation = 0.0;
calData->val.patch[i].sampleType = KcsChromatic;

calData->val.patch[i].input[KcsRGB_R] = (float)i/255;
calData->val.patch[i].input[KcsRGB_G] = (float)i/255;
calData->val.patch[i].input[KcsRGB_B] = (float)i/255;
calData->val.patch[i].input[3] = 0.0;

calData->val.patch[i].output[KcsRGB_R] = Luminance_float_out[0][i];
calData->val.patch[i].output[KcsRGB_G] = Luminance_float_out[1][i];
calData->val.patch[i].output[KcsRGB_B] = Luminance_float_out[2][i];
calData->val.patch[i].output[3] = 0.0;
}
```

Code Example A-4 `kcms_create.c` *(Continued)*

```
calData->val.patch[0].sampleType = KcsBlack;
calData->val.patch[255].sampleType = KcsWhite;

if (!argv[1]) {
fprintf(stderr, "Usage kcms_create profile_out\n");
exit(1);
}

desc.type = KcsSolarisProfile;
desc.desc.solarisFile.fileName = argv[1];
desc.desc.solarisFile.hostName = NULL;
desc.desc.solarisFile.oflag = O_RDWR|O_CREAT|O_TRUNC;
desc.desc.solarisFile.mode = 0666;
c_desc.profileDesc = &desc;
c_desc.desc.id.cmmId = 0;
c_desc.desc.id.cmmVersionId = 0;
c_desc.desc.id.profileId = 0;
c_desc.desc.id.profileVersionId = 0;
status = KcsCreateProfile(&profileid, &c_desc);
if(status != KCS_SUCCESS) {
    KcsGetLastError(&errDesc);
    printf("CreateProfile error: %s\n", errDesc.desc);
    exit(1);
}

if ((description = (char *)malloc(strlen(SAMPLE_WORDS) + 1)) == NULL) {
    perror("malloc failed : ");
    KcsFreeProfile(profileid);
    exit(1);
}
memset(description, 0, strlen(SAMPLE_WORDS) + 1);
strcpy(description, SAMPLE_WORDS);
if ((attrValue2 = string2icTextAttrValue(description)) == NULL) {
    fprintf(stderr, "conversion to AttrValue failed \n");
    KcsFreeProfile(profileid);
    exit(1);
}
if (KcsSetAttribute(profileid, icSigProfileDescriptionTag,
            attrValue2) != KCS_SUCCESS) {
    KcsGetLastError(&errDesc);
    printf("Set Atrribute error: %s\n", errDesc.desc);
    exit(1);
}
```

Code Example A-4 `kcms_create.c` *(Continued)*

```
    free(attrValue2);
    free(description);
    size = sizeof(KcsAttributeBase) + sizeof(icHeader);
    attrValuePtr = (KcsAttributeValue *)malloc(size);

    /* Build the header */
    attrValuePtr->base.type = icSigHeaderType;
    attrValuePtr->base.sizeOfType = sizeof(icHeader);
    attrValuePtr->base.countSupplied = 1;
    KcsGetAttribute(profileid, icSigHeaderTag, attrValuePtr);
    attrValuePtr->val.icHeader.size = 0;

/* The following three values do not have to be set if you do a GetAttribute
 * on the header, since the Create should set them for you.  If you do not do
 * a GetAttribute of the header, you must set these.
 *   attrValuePtr->val.icHeader.cmmId = 0x4b434d53;
 *   attrValuePtr->val.icHeader.version =icVersionNumber;
 *   attrValuePtr->val.icHeader.magic = icMagicNumber;
*/
    attrValuePtr->val.icHeader.deviceClass = icSigDisplayClass;
    attrValuePtr->val.icHeader.colorSpace = icSigRgbData;
    attrValuePtr->val.icHeader.pcs = icSigXYZData;

    /* Get the time from the system */
    clocktime = time(NULL);
    datetime = localtime(&clocktime);

    /* Create space for the tag */

    attrValuePtr->val.icHeader.date.seconds = (icUInt16Number)datetime->tm_sec;
    attrValuePtr->val.icHeader.date.minutes = (icUInt16Number)datetime->tm_min;
    attrValuePtr->val.icHeader.date.hours = (icUInt16Number)datetime->tm_hour;
    attrValuePtr->val.icHeader.date.day = (icUInt16Number)datetime->tm_mday;
    attrValuePtr->val.icHeader.date.month = (icUInt16Number)datetime->tm_mon + 1;
    attrValuePtr->val.icHeader.date.year = (icUInt16Number)datetime->tm_year;

    attrValuePtr->val.icHeader.platform = icSigSolaris;

    attrValuePtr->val.icHeader.flags = icEmbeddedProfileFalse || icUseAnywhere;

    strcpy(description,"SUNW ");
    memcpy(&attrValuePtr->val.icHeader.manufacturer, description, 4);
    attrValuePtr->val.icHeader.model = 0;
    attrValuePtr->val.icHeader.attributes[0] = 0;
```

Code Example A-4 `kcms_create.c` *(Continued)*

```
    attrValuePtr->val.icHeader.attributes[1] = 0;
    attrValuePtr->val.icHeader.renderingIntent = icPerceptual;

    attrValuePtr->val.icHeader.illuminant.X = double2icfixed(test_double[0],
           icSigS15Fixed16ArrayType);
    attrValuePtr->val.icHeader.illuminant.Y = double2icfixed(test_double[1],
           icSigS15Fixed16ArrayType);
    attrValuePtr->val.icHeader.illuminant.Z = double2icfixed(test_double[2],
           icSigS15Fixed16ArrayType);
    rc = KcsSetAttribute(profileid, icSigHeaderTag, attrValuePtr);
    if(rc != KCS_SUCCESS) {
        rc =KcsGetLastError(&errDesc);
        fprintf(stderr, "unable to set header: %s\n", errDesc.desc);
        KcsFreeProfile(profileid);
        return(-1);
    }
    free(attrValuePtr);

/*  Not necessary to set this - for example only */
    attrValue.base.type = icSigMeasurementType;
    attrValue.base.sizeOfType = sizeof(icMeasurement);
    attrValue.base.countSupplied = 1;
    attrValue.val.icMeasurement.illuminant = icIlluminantD50;
    attrValue.val.icMeasurement.stdObserver = icStdObsUnknown;
    attrValue.val.icMeasurement.geometry = icGeometryUnknown;
    attrValue.val.icMeasurement.flare = icFlare0;
    attrValue.val.icMeasurement.backing.X = 0;
    attrValue.val.icMeasurement.backing.Y = 0;
    attrValue.val.icMeasurement.backing.Z = 0;
    rc = KcsSetAttribute(profileid, icSigMeasurementTag, &attrValue);
    if(rc != KCS_SUCCESS) {
        KcsGetLastError(&errDesc);
        fprintf(stderr, "unable to set icMeasurent: %s\n", errDesc.desc);
        KcsFreeProfile(profileid);
        return(-1);
    }

    /* set white point and colorants with dummy values just to show it works*/

    attrValue.base.countSupplied = 1;
    attrValue.base.type = icSigXYZType;
    attrValue.base.sizeOfType = sizeof(icXYZNumber);
    attrValue.val.icXYZ.data[0].X = double2icfixed(test_double[0],
           icSigS15Fixed16ArrayType);
```

Code Example A-4 `kcms_create.c` *(Continued)*

```
    attrValue.val.icXYZ.data[0].Y = double2icfixed(test_double[1],
            icSigS15Fixed16ArrayType);
    attrValue.val.icXYZ.data[0].Z = double2icfixed(test_double[2],
            icSigS15Fixed16ArrayType);
    rc = KcsSetAttribute(profileid, icSigMediaWhitePointTag, &attrValue);
    if(rc != KCS_SUCCESS) {
        KcsGetLastError(&errDesc);
        fprintf(stderr, "unable to set whitepoint: %s\n", errDesc.desc);
        KcsFreeProfile(profileid);
        return(-1);
    }
    test_double[0] = 0.572586;
    test_double[1] = 0.337198;
    test_double[2] = 0.026291;
    attrValue.val.icXYZ.data[0].X = double2icfixed(test_double[0],
            icSigS15Fixed16ArrayType);
    attrValue.val.icXYZ.data[0].Y =double2icfixed(test_double[1],
            icSigS15Fixed16ArrayType);
    attrValue.val.icXYZ.data[0].Z =double2icfixed(test_double[2],
            icSigS15Fixed16ArrayType);
    rc = KcsSetAttribute(profileid, icSigRedColorantTag, &attrValue);
    if(rc != KCS_SUCCESS) {
        KcsGetLastError(&errDesc);
        fprintf(stderr, "unable to set red primaries: %s\n", errDesc.desc);
        KcsFreeProfile(profileid);
        return(-1);
    }
    test_double[0] = 0.2556;
    test_double[1] = 0.600189;
    test_double[2] = 0.097794;
    attrValue.val.icXYZ.data[0].X = double2icfixed(test_double[0],
            icSigS15Fixed16ArrayType);
    attrValue.val.icXYZ.data[0].Y = double2icfixed(test_double[1],
            icSigS15Fixed16ArrayType);
    attrValue.val.icXYZ.data[0].Z = double2icfixed(test_double[2],
            icSigS15Fixed16ArrayType);
    rc = KcsSetAttribute(profileid, icSigGreenColorantTag, &attrValue);
    if(rc != KCS_SUCCESS) {
        KcsGetLastError(&errDesc);
        fprintf(stderr, "unable to set green primaries: %s\n", errDesc.desc);
        KcsFreeProfile(profileid);
        return(-1);
    }
    test_double[0] = 0.135986;
```

Code Example A-4 `kcms_create.c` *(Continued)*

```
    test_double[1] = 0.062592;
    test_double[2] = 0.70079;
    attrValue.val.icXYZ.data[0].X = double2icfixed(test_double[0],
            icSigS15Fixed16ArrayType);
    attrValue.val.icXYZ.data[0].Y = double2icfixed(test_double[1],
            icSigS15Fixed16ArrayType);
    attrValue.val.icXYZ.data[0].Z = double2icfixed(test_double[2],
            icSigS15Fixed16ArrayType);
    rc = KcsSetAttribute(profileid, icSigBlueColorantTag, &attrValue);
    if(rc != KCS_SUCCESS) {
        KcsGetLastError(&errDesc);
        fprintf(stderr, "unable to set blue primaries: %s\n", errDesc.desc);
        KcsFreeProfile(profileid);
        return(-1);
    }

    rc = KcsUpdateProfile(profileid, NULL, calData, NULL);
    if(rc != KCS_SUCCESS) {
        KcsGetLastError(&errDesc);
        fprintf(stderr, "unable to update profile: %s\n", errDesc.desc);
        KcsFreeProfile(profileid);
        return(-1);
    }
    rc = KcsSaveProfile(profileid, &desc);
    if(rc != KCS_SUCCESS) {
        KcsGetLastError(&errDesc);
        fprintf(stderr, "unable to save profile: %s\n", errDesc.desc);
        KcsFreeProfile(profileid);
        return(-1);
    }
    KcsFreeProfile(profileid);

    /* Reload to to verify that a readable profile was built */
    desc.type = KcsSolarisProfile;
    desc.desc.solarisFile.fileName = argv[1];
    desc.desc.solarisFile.hostName = NULL;
    desc.desc.solarisFile.oflag = O_RDWR;
    desc.desc.solarisFile.mode = 0;
    rc = KcsLoadProfile(&profileid, &desc, KcsLoadAttributesNow);
    if(rc != KCS_SUCCESS) {
        KcsGetLastError(&errDesc);
        fprintf(stderr, "unable to load saved profile: %s\n", errDesc.desc);
        KcsFreeProfile(profileid);
        return(-1);
```

Code Example A-4 kcms_create.c *(Continued)*

```
    }
    KcsFreeProfile(profileid);
    exit(0);
}
```

kcms_utils.c

Code Example A-5 kcms_utils.c

```
#pragma ident "@(#)kcms_utils.c1.2  04 Jan 1995"
/* Copyright (c) 1994, by Sun Microsystems, Inc.
 * All rights reserved.
 */

/* A set of utility functions useful for ICC attributes
 */

#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <kcms/kcs.h>
#include <kcms/icc.h>
#include <kcms/kcstypes.h>

KcsAttributeValue *
string2icTextAttrValue(char *description)
{
    KcsAttributeValue  *attrValue;
    int                size;
    icUInt32Number     ucCode, ucCount;
    icUInt16Number     scCount;
    icUInt8Number      scCode;
    icInt8Number       *ptr8;

    /* Make space for the attribute and null it out, AttributeBase +
     * ascii count + data, UniCode code + count scriptCode code + count
     */
    size = sizeof(KcsAttributeBase) + sizeof(icUInt32Number) +
           (strlen(description) + 1) + 2*sizeof(icUInt32Number) +
           sizeof(icUInt16Number) + sizeof(icUInt8Number);
    attrValue = (KcsAttributeValue *)malloc(size);
    if (attrValue == NULL)
        return(attrValue);
```

Code Example A-5 `kcms_utils.c` *(Continued)*

```
    memset(attrValue, 0, size);

    attrValue->base.type = icSigTextDescriptionType;
    attrValue->base.countSupplied = 1;
    attrValue->base.sizeOfType = size - sizeof(KcsAttributeBase);
    attrValue->val.icTextDescription.count = strlen(description) + 1;
    memcpy(attrValue->val.icTextDescription.data, description,
           strlen(description));
    ptr8 = attrValue->val.icTextDescription.data +
           strlen(description) + 1;
    ucCode = 0;
    ucCount = 0;
    scCode = 0;
    scCount = 0;

    /* Write in the UniCode data */
    memcpy(ptr8, &ucCode, sizeof(ucCode));
    ptr8 += sizeof(ucCode);
    memcpy(ptr8, &ucCount, sizeof(ucCount));
    ptr8 += sizeof(ucCount);

    /* Write in the ScriptCode data */
    memcpy(ptr8, &scCode, sizeof(scCode));
    ptr8 += sizeof(scCode);
    memcpy(ptr8, &scCount, sizeof(scCount));
    return (attrValue);
}


/* Convert IC fixed point to a double, float is not accurate enough
 * for boundary conditions.
 */
double
icfixed2double(long val, long type)
{
    double         retval;
    short          a;
    unsigned short b, c;

    switch(type) {
    case icSigS15Fixed16ArrayType :
        a = (short)(val>>16);
        b = (unsigned short)(val&0x0000FFFF);
```

Code Example A-5 `kcms_utils.c` *(Continued)*

```
        retval = (double)a + ((double)b)/65536.0;
        break;
    case icSigU16Fixed16ArrayType :
        c = (unsigned short)(val>>16);
        b = (unsigned short)(val&0x0000FFFF);
        retval = (double)c + ((double)b)/65536.0;
        break;
    default :
        retval  = -1.0;
        break;
    }
    return(retval);
}

/* Convert double to an IC, fixed point number is not accurate enough
 * for boundary conditions.
 */
long
double2icfixed(double val, long type)
{
    long           retval;
    short          a;
    unsigned short b, c;

    switch(type) {
    case icSigS15Fixed16ArrayType :
        a = (short)(val);
        b = (unsigned short)((val - a) * 65536.0);
        retval = (a <<16) | b;
        break;
    case icSigU16Fixed16ArrayType :
        c = (unsigned short)(val);
        b = (unsigned short)((val - c) * 65536.0);
        retval = (c <<16) | b;
        break;
    default :
        retval  = -1.0;
        break;
    }
    return(retval);
}

/* Return the number of channels in a profile from
 * the header colorspace field
```

Code Example A-5 kcms_utils.c *(Continued)*

```
 */
int
icchannels(icColorSpaceSignature cspace)
{
    int nchannels;

    switch (cspace) {
    case icSigCmykData :/* CMYK need 4 channels */
        nchannels = 4;
        break;
    case icSigGrayData :/* Grayscale only one */
        nchannels = 1;
        break;
    case icSigXYZData :
    case icSigLabData :
    case icSigLuvData :
    case icSigYCbCrData :
    case icSigYxyData :
    case icSigRgbData :
    case icSigHsvData :
    case icSigHlsData :
    case icSigCmyData :
        nchannels = 3;
        break;
    default :              /* Everything else is 3 */
        nchannels = 3;
        break;
    }
    return (nchannels);
}
```

kcms_utils.h

Code Example A-6 kcms_utils.h

```
#ifndef _KCMS_UTIL_H_
#define _KCMS_UTIL_H_

#pragma ident "@(#)kcms_utils.h1.3  06 Jan 1995"
/* Copyright (c) 1994, by Sun Microsystems, Inc.
 * All rights reserved.
 */

#include <kcms/icc.h>
```

Code Example A-6 `kcms_utils.h`

```
#if defined (__cplusplus)
extern "C" {
#endif

#if defined (__STDC__)
extern KcsAttributeValue *string2icTextAttrValue(char *description);
extern double             icfixed2double(long val, long type);
extern long               double2icfixed(double val, long type);
extern int                icchannels(icColorSpaceSignature cspace);
#else
extern KcsAttributeValue *string2icTextAttrValue();
extern double             icfixed2double();
extern long               double2icfixed();
extern int                icchannels();
#endif __STDC__

#if defined (__cplusplus)
}
#endif

#endif _KCMS_UTIL_H_
```

A