Ada & C

Martyn Pike

university.adacore.com

Introduction

- Language Standard (ISO/IEC 8652:2012)
- Interfacing to C is defined in the Language Standard
- Uses Ada 2012 Aspects
  - Convention, Import, Export, Link_Name, External_Name
- Child Packages of Interfaces are available
  - Interfaces.C
  - Interfaces.C.Pointers
  - Interfaces.C.Strings
- Possible to have mixed language Main programs
  - C Main program with exported with Ada Subprograms
  - Ada Main program with imported C Subprograms

Importing C Subprograms

```
with Interfaces.C; with Ada.Text_IO;
procedure Main is

   function Get_Length return Interfaces.C.size_t with
      Convention    => C,
      Import        => True,
      External_Name => "getLength";

begin
   Ada.Text_IO.Put_Line(Get_Length'Img);
end Main;
```

```
#include <stdlib.h>

extern size_t getLength(void)
{
   return (size_t)10;
}
```

Importing C Memory Objects

```
with Interfaces.C; with Ada.Text_IO;
procedure Main is

   The_Length : Interfaces.C.size_t with
      Convention    => C,
      Import        => True,
      External_Name => "theLength";

begin
   Ada.Text_IO.Put_Line(The_Length'Img);
end Main;
```

```
#include <stdlib.h>

const size_t theLength = 20;
```

Exporting Ada Subprograms

```
with Interfaces.C;
package ALib is

   function Get_Length return Interfaces.C.size_t with
      Convention    => C,
      Export        => True,
      External_Name => "getLength";

end ALib;
```

```
package body ALib is

   function Get_Length return Interfaces.C.size_t is
   begin
      return 60;
   end Get_Length;

end ALib;
```

```
#include <stdio.h>

extern size_t getLength(void);

[ … ]

printf("%d\n", getLength());
```

Exporting Ada Memory Objects

```
with Interfaces.C;
package ALib is

   The_Length : constant Interfaces.C.size_t := 80 with
      Convention    => C,
      Export        => True,
      External_Name => "theLength";

end ALib;
```

```
#include <stdio.h>

extern size_t theLength;

[ … ]

printf("%d\n", theLength);
```

Interfaces.C

- **Provides types**
  - int, short, long, signed_char, unsigned, char
  - ptrdiff_t, size_t
  - C_Float, double, long_double
- **Provides constants**
  - nul, wide_nul, char16_nul, char32_nul
- **Provides subprograms**
  - To_C for strings and variants of characters
  - To_Ada for strings and variants of characters
- **Forms the foundation for other C related packages**

Importing C arrays

```
with Interfaces.C;
with Ada.Text_IO;

procedure Main is

   type C_Array_Type is array(1..4) of Interfaces.C.size_t;
   package Size_T_IO is new Ada.Text_IO.Modular_IO(Num => Interfaces.C.size_t);

   C_Array : C_Array_Type with
      Convention    => C,
      Import        => True,
      External_Name => "c_array";

begin
   for I in C_Array'Range loop
       Size_T_IO.Put(
         Item  => C_Array(I),
         Base  => 16
       );
      Ada.Text_IO.New_Line;
   end loop;
end Main;
```

```
#include <stdlib.h>

const size_t c_array[4] = { 0x10, 0x20, 0x30, 0x40 };
```

Importing C Strings

```
char *theString = "Hello World";

char* getString(void)
{
   return theString;
}
```

```
with Interfaces.C.Strings; use Interfaces.C.Strings;
with Ada.Text_IO; use Ada.Text_IO;

procedure Main is

   function Get_String return chars_ptr with
      Convention    => C,
      Import        => True,
      External_Name => "getString";

begin
   Put_Line(Interfaces.C.Strings.Value(Get_String));
end Main;
```

Exporting Ada Strings

```
with Interfaces.C.Strings;

package ALib is

   My_Strings : constant
      Interfaces.C.Strings.chars_ptr_array(1..2) := (
         1 => Interfaces.C.Strings.New_String("Hello World"),
         2 => Interfaces.C.Strings.Null_Ptr
      ) with
      Convention    => C,
      Export        => True,
      External_Name => "someStrings";

end ALib;
```

```
#include <stdio.h>

extern char* someStrings[];

[ … ]

int i = 0;
while (NULL != someStrings[i])
{
   printf("%s ", someStrings[i]);
   i++;
}
```

Interfaces.C.Pointers

```
with Interfaces.C.Pointers; with Ada.Text_IO;

procedure Main is

   type Index is range 1..30;
   type Element_Array is array(Index range <>) of aliased Interfaces.C.unsigned;

   package Obj_Ptr is new
      Interfaces.C.Pointers(
         Index              => Index,
         Element            => Interfaces.C.unsigned,
         Element_Array      => Element_Array,
         Default_Terminator => Interfaces.C.unsigned'(9999)
      );

   C_Array : aliased Interfaces.C.unsigned with
      Convention    => C,
      Import        => True,
      External_Name => "cArray";

   My_Array : Element_Array := Obj_Ptr.Value(C_Array'Access);

begin
   for I in My_Array'Range loop
      exit when Interfaces.C."="(9999,My_Array(I));
      Ada.Text_IO.Put_Line(My_Array(I)'Img);
   end loop;
end Main;
```

```
const unsigned myArray[] = { 30, 60, 70, 90, 100, 9999 };
```

Access to Subprograms

```
with Interfaces.C;

procedure Main is

   type Size_T_Ptr is access all Interfaces.C.size_t;
   type C_Function_Ptr is access procedure (X : Size_T_Ptr);

   A_Size_T : Size_T_Ptr := new Interfaces.C.size_t'(10);

   A_C_Func : C_Function_Ptr with
      Convention    => C,
      Import        => True,
      External_Name => "func";

begin

   A_C_func.all(X => A_Size_T);

end Main;
```

```
#include <stdio.h>

void c_function(size_t *x)
{
   printf("%d\n", *x);
}

void (*func)(size_t *x) = c_function;
```

The Issue of C main and Ada Elaboration

```
with Interfaces.C;

package Alib is

   function F return Interfaces.C.int is (55);

   V : Interfaces.C.int := F with
      Convention    => C,
      Export        => True,
      External_Name => "val";

end Alib;
```

```
#include <stdio.h>

extern int val;

extern void adainit (void);
extern void adafinal(void);

int main (int argc, char ** argv) {
  adainit ();
  printf ("val %i", val);
  adafinal ();
}
```

Processing C Header Files

```
/* clib.h */
#include <stdio.h>

void function1(void* x);

void* function2(void);

char* function3(char** j);
```

```
$ gcc -fdump-ada-spec -C clib.h
```

```
pragma Ada_2005;
pragma Style_Checks (Off);

with Interfaces.C; use Interfaces.C;
with System;
with Interfaces.C.Strings;

package clib_h is

   procedure function1 (arg1 : System.Address);  -- clib.h:3
   pragma Import (C, function1, "function1");

   function function2 return System.Address;  -- clib.h:5
   pragma Import (C, function2, "function2");

   function function3 (arg1 : System.Address) return Interfaces.C.Strings.chars_ptr;  -- clib.h:7
   pragma Import (C, function3, "function3");

end clib_h;
```

Quiz

Is this correct? (1/10)

YES
(click on the check icon)

NO
(click on the error location(s))

```
with Interfaces.C; with Ada.Text_IO;
procedure Main is

   function Get_Length return Interfaces.C.size_t with
      Convention    => C,
      Import        => True,
      External_Name => "getLength";

begin
   Ada.Text_IO.Put_Line(Get_Length'Img);
end Main;
```

```
#include <stdlib.h>

extern size_t getlength(void)
{
   return (size_t)10;
}
```

Is this correct? (1/10)

NO

The External_Name string must match the foreign language entity name.

While Ada is case insensitive, C is case sensitive

```
with Interfaces.C; with Ada.Text_IO;
procedure Main is

   function Get_Length return Interfaces.C.size_t with
      Convention    => C,
      Import        => True,
      External_Name => "getLength";

begin
   Ada.Text_IO.Put_Line(Get_Length'Img);
end Main;
```

```
#include <stdlib.h>

extern size_t getlength(void)
{
   return (size_t)10;
}
```

Is this correct? (2/10)

YES
(click on the check icon)

NO
(click on the error location(s))

```
with Interfaces.C;
package ALib is

   function Get_Length return Interfaces.C.size_t with
      Convention    => C,
      Export        => True,
      External_Name => "getLength";

end ALib;
```

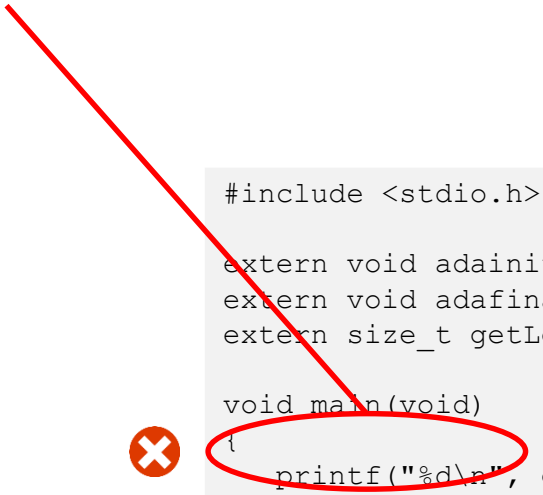
```
package body ALib is

   function Get_Length return Interfaces.C.size_t is
   begin
      return 60;
   end Get_Length;

end ALib;
```

```
#include <stdio.h>

extern void adainit(void);
extern void adafinal(void);
extern size_t getLength(void);

void main(void)
{
   printf("%d\n", getLength());
   adafinal();
}
```

Is this correct? (2/10)

NO

Missing call to adainit()




```
#include <stdio.h>

extern void adainit(void);
extern void adafinal(void);
extern size_t getLength(void);

void main(void)
{
   printf("%d\n", getLength());
   adafinal();
}
```

Is this correct? (3/10)

YES
(click on the check icon)

NO
(click on the error location(s))

```
with Interfaces.C; with Ada.Text_IO;
procedure Main is

   The_Length : Interfaces.C.size_t with
      Convention    => C,
      Import        => True,
      External_Name => "theLength";

begin
   Ada.Text_IO.Put_Line(The_Length'Img);
end Main;
```

```
#include <stdlib.h>

const size_t theLength = 20;
```

Is this correct? (3/10)


```
with Interfaces.C; with Ada.Text_IO;
procedure Main is

   The_Length : Interfaces.C.size_t with
      Convention    => C,
      Import        => True,
      External_Name => "theLength";

begin
   Ada.Text_IO.Put_Line(The_Length'Img);
end Main;
```

```
#include <stdlib.h>

const size_t theLength = 20;
```

Is this correct? (4/10)

YES
(click on the check icon)

NO
(click on the error location(s))

```
with Interfaces.C;
package ALib is

  Exported_Object : aliased Interfaces.C.unsigned with
     Convention    => C,
     Export        => True,
     External_Name => "objFromAda";

  type ObjPtr is access all Interfaces.C.unsigned with
     Convention    => C;

  procedure Output_Object(Obj : ObjPtr) with
     Convention    => C,
     Export        => True,
     External_Name => "procFromAda";

end ALib;
```

```
with Ada.Text_IO;
package body ALib is

  procedure Output_Object(Obj : ObjPtr) is
  begin
     Ada.Text_IO.Put_Line(Obj.all'Img);
  end Output_Object;

end ALib;
```

```
#include <stdio.h>

extern void adainit(void);
extern void adafinal(void);
extern void procFromAda(unsigned* x);
extern unsigned objFromAda;

void main(void)
{
   adainit();
   objFromAda = 50;
   procFromAda(&objFromAda);
   adafinal();
}
```

Is this correct? (4/10)


```
with Interfaces.C;
package ALib is

  Exported_Object : aliased Interfaces.C.unsigned with
     Convention    => C,
     Export        => True,
     External_Name => "objFromAda";

  type ObjPtr is access all Interfaces.C.unsigned with
     Convention    => C;

  procedure Output_Object(Obj : ObjPtr) with
     Convention    => C,
     Export        => True,
     External_Name => "procFromAda";

end ALib;
```

```
with Ada.Text_IO;
package body ALib is

  procedure Output_Object(Obj : ObjPtr) is
  begin
     Ada.Text_IO.Put_Line(Obj.all'Img);
  end Output_Object;

end ALib;
```

```
#include <stdio.h>

extern void adainit(void);
extern void adafinal(void);
extern void procFromAda(unsigned* x);
extern unsigned objFromAda;

void main(void)
{
   adainit();
   objFromAda = 50;
   procFromAda(&objFromAda);
   adafinal();
}
```

Is this correct? (5/10)

YES
(click on the check icon)

NO
(click on the error location(s))

```
with Interfaces.C; with Ada.Text_IO;
procedure Main is

   type ObjPtr is access all Interfaces.C.unsigned;
   type ObjPtr_Array is array(Positive range <>) of aliased Interfaces.C.unsigned;

   My_Array : aliased ObjPtr_Array(1..30) with
      Convention    => C,
      Import        => True,
      External_Name => "myArray";

   My_Ptr : ObjPtr := My_Array(1)'Access;

begin

   for I in My_Array'Range loop

      My_Ptr := My_Array(I)'Access;

      exit when Interfaces.C."="(9999,My_Ptr.all);

      Ada.Text_IO.Put_Line(My_Ptr.all'Img);

   end loop;

end Main;
```

```
const unsigned myArray[] = { 30, 60, 70, 90, 100, 9999 };
```

Is this correct? (5/10)


```
with Interfaces.C.Pointers; with Ada.Text_IO;

procedure Main is

   type Index is range 1..30;
   type Element_Array is array(Index range <>) of aliased Interfaces.C.unsigned;

   package Obj_Ptr is new
      Interfaces.C.Pointers(
         Index              => Index,
         Element            => Interfaces.C.unsigned,
         Element_Array      => Element_Array,
         Default_Terminator => Interfaces.C.unsigned'(9999)
      );

   C_Array : aliased Interfaces.C.unsigned with
      Convention    => C,
      Import        => True,
      External_Name => "cArray";

   My_Array : Element_Array := Obj_Ptr.Value(C_Array'Access);

begin

   for I in My_Array'Range loop
      exit when Interfaces.C."="(9999,My_Array(I));
      Ada.Text_IO.Put_Line(My_Array(I)'Img);
   end loop;

end Main;
```

(7/10)

```
with Interfaces.C.Strings;
with Ada.Text_IO;

procedure Main is

  C_String : Interfaces.C.Strings.chars_ptr with
     Convention    => C,
     Import        => True,
     External_Name => "cString";

begin
  Ada.Text_IO.Put_Line(Interfaces.C.Strings.Value(C_String));
end Main;
```

```
const char *cString = "Hello World";
```

(7/10)

```
with Interfaces.C.Strings;
with Ada.Text_IO;

procedure Main is

  C_String : Interfaces.C.Strings.chars_ptr with
     Convention    => C,
     Import        => True,
     External_Name => "cString";

begin
  Ada.Text_IO.Put_Line(Interfaces.C.Strings.Value(C_String));
end Main;
```

```
const char *cString = "Hello World";
```

(8/10)

```
package ALib is

   Ada_String : constant String := "Hello" & " World" with
      Convention    => C,
      Export        => True,
      External_Name => "adaString";

end ALib;
```

```
#include <stdio.h>

extern void adainit(void);
extern void adafinal(void);
extern char* adaString;

void main(void)
{
   adainit();
   printf("%s\n", &adaString);
   adafinal();
};
```

(8/10)

```
package ALib is

   Ada_String : constant String := "Hello" & " World" with
      Convention    => C,
      Export        => True,
      External_Name => "adaString";

end ALib;
```

```
#include <stdio.h>

extern void adainit(void);
extern void adafinal(void);
extern char* adaString;

void main(void)
{
   adainit();
   printf("%s\n", &adaString);
   adafinal();
};
```

(9/10)

```
with Interfaces.C.Strings; use Interfaces.C.Strings;
with Ada.Text_IO; use Ada.Text_IO;

procedure Main is

   function Get_String return char_array_access with
      Convention    => C,
      Import        => True,
      External_Name => "getString";

begin

   Put_Line(
      Interfaces.C.Strings.Value(
         Interfaces.C.Strings.To_Chars_Ptr(Get_String)
      )
   );

end Main;
```

```
char *theString = "Hello World";

char* getString(void)
{
   return theString;
}
```

(9/10)

```
with Interfaces.C.Strings; use Interfaces.C.Strings;
with Ada.Text_IO; use Ada.Text_IO;

procedure Main is

   function Get_String return char_array_access with
      Convention    => C,
      Import        => True,
      External_Name => "getString";

begin

   Put_Line(
      Interfaces.C.Strings.Value(
         Interfaces.C.Strings.To_Chars_Ptr(Get_String)
      )
   );

end Main;
```

```
char *theString = "Hello World";

char* getString(void)
{
   return theString;
}
```

(10/10)

```
with Interfaces.C.Strings;

package ALib is

   My_Strings : constant
      Interfaces.C.Strings.chars_ptr_array(1..4) := (
         1 => Interfaces.C.Strings.New_String("Hello"),
         2 => Interfaces.C.Strings.New_String("World"),
         3 => Interfaces.C.Strings.Null_Ptr,
         4 => Interfaces.C.Strings.New_String("End")
      ) with
      Convention    => C,
      Export        => True,
      External_Name => "someStrings";

end ALib;
```

```
#include <stdio.h>

extern void adafinal(void);
extern void adainit(void);
extern char* someStrings[];

void main(void)
{
   adainit();
   int i = 0;
   while (NULL != someStrings[i])
   {
      printf("%s ", someStrings[i]);
      i++;
   }
   adafinal();
};
```

(10/10)

```
with Interfaces.C.Strings;

package ALib is

   My_Strings : constant
      Interfaces.C.Strings.chars_ptr_array(1..4) := (
         1 => Interfaces.C.Strings.New_String("Hello"),
         2 => Interfaces.C.Strings.New_String("World"),
         3 => Interfaces.C.Strings.Null_Ptr,
         4 => Interfaces.C.Strings.New_String("End")
      ) with
      Convention    => C,
      Export        => True,
      External_Name => "someStrings";

end ALib;
```

```
#include <stdio.h>

extern void adafinal(void);
extern void adainit(void);
extern char* someStrings[];

void main(void)
{
   adainit();
   int i = 0;
   while (NULL != someStrings[i])
   {
      printf("%s ", someStrings[i]);
      i++;
   }
   adafinal();
};
```

university.adacore.com