



嵌入式技术

邢超

计算机程序设
计语言

混合语言编程
思考

嵌入式技术

混合语言程序设计

邢超

西北工业大学航天学院



- 纸带
- 汇编
- 高级语言
 - 函数式语言 (Functional language):
 - Lisp (LIsT Processing)
 - Haskell
 - Caml (Objective Caml)
 - 命令式语言 (Imperative language):
 - Fortran
 - C
 - Pascal
 - 脚本语言 (Descript language):
 - HTML
 - Javascript
 - Postscript



嵌入式技术

邢超

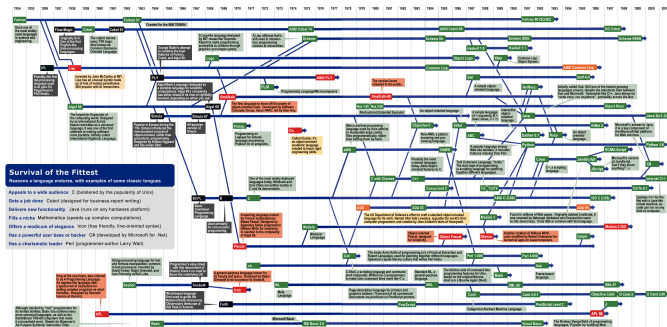
Mother Tongues

Tracing the roots of computer languages through the ages

Just like half of the world's spoken tongues, most of the 2,358-plus computer programming languages are either endangered or extinct. As powerhouse C/C++, Visual Basic, C#, Java and other modern source codes dominate our systems, hundreds of older languages are running out of life. As for less collection of engineers/electronic linguists, if you will, are to save, or at least document the legacy of classic software. They're combing the globe's 3 billion developers in search of coders still fluent in these nearly forgotten lingua francae. Among the most endangered are Ada, APL, B (the predecessor of C), Lisp, Oberon, Smalltalk, and Simula.

Code-slayer Gusty Bouch, National Software's chief scientist, is working with the Computer History Museum in Silicon Valley to record and, in some cases, maintain languages by writing new compilers so our ever-changing hardware can speak the code. Why bother? "They tell us about the state of software practice, the needs of their inventors, and the technical, social, and economic forces that shaped history at the time," Bouch explains. "They provide the raw material for software archaeologists, historians, and developers to learn what worked, what was brilliant, and what was an afterthought." Here's a peek at the strongest branches of programming's family tree. For a nearly exhaustive narrative, check out the Language List at http://www.infoworld.com/feature/04/04/040401lang_list.html - Michael Redmond

Key
 New technology
 Active accounts of users
 Potential input or conversion, complex
 Active
 Endangered, steep drop-off
 Extinct, no known active users or up-to-date compilers
 Language continues



Sources: Paul Bouch, Brent Halpern, associate director of computer science at IBM Research; The Retrocomputing Museum; Todd Probsting, senior researcher at Microsoft; Gusty Bouch, computer scientist, Stanford University

计算机程序设计语言

混合语言编程

思考



嵌入式技术

邢超

计算机程序设计语言

混合语言编程

思考

- 组合方式
 - 多个程序
 - 单个程序
- 通信方式
 - 文件
 - 管道
 - 网络
 - 共享库



嵌入式技术

邢超

计算机程序设计语言

混合语言编程

思考

- IEEE 754 float
- Big/Little Endian
- 数组



嵌入式技术

邢超

计算机程序设
计语言

混合语言编程

思考

- 基础
 - C/C++
 - Java
- 扩展/嵌入
 - 汇编
 - Fortran
 - MATLAB/SIMULINK
 - Scilab/Scicos
 - Lua, Python, Perl
 - Lisp, Scheme, Ocaml



嵌入式技术

邢超

计算机程序设计语言

混合语言编程

思考



- Java Platform
 - Java
 - JNI
- .Net Framework
 - C#
 - CLR



嵌入式技术

邢超

计算机程序设
计语言

混合语言编程

思考

- C 调用 Fortran 过程
- Fortran 调用 C 函数



```

program main
  implicit none
  integer i,j,M,N;
  integer , external :: r2;
  real A(3,3),B(3,3),C(5,3);
  real , allocatable ,dimension (:,::D;

A=1;  B=2;  B(1:3:2,1:3:2)=3;
B(3:2:-1,2:1:-1)=A(1:2,2:3);
B(int(A(:,1)),1)=(/4,5,6/);
where(B==3)
  B=5
endwhere
C= reshape((/ ((sin(PI/M)*cos(PI/N),
  M=1,5), N=1,3) /),(/5,3/));
allocate (D(size(C,1),size(C,2)));
D= reshape((/ ((M+N*10, M=1,5), N=1,3) /),(/5,3/));
print *, D;
write(*,*) r2(1)
end program main

```



```
recursive integer function r2(i) result(r)
integer i
integer r
write(*,*) 'recurseive ... ',i;
r=r2(i+1);
end function
```

```
subroutine s(a)
integer a
a=a+1;
end subroutine
```



嵌入式技术

邢超

计算机程序设
计语言

混合语言编程

思考

```
PROGRAM MAIN
  use iso_c_binding
  INTERFACE
    subroutine fact(n) bind(C,name="Fact")
      INTEGER(4) n(2,2)
    END subroutine
  END INTERFACE
  INTEGER(4) n(2,2)
  write(*,*) 'before□:□' , N
  call fact(N)
  write(*,*) 'after□:□' , N
END
```



```
void Fact(int a[2][2])
{
    int i,j;
    /*
    for (i=0;i<4;i++) *(a[0]+i)=i;
    */
    for (i=0;i<2;i++)for (j=0;j<2;j++) a[i][j]=i*2+j;
    /*
    for (i=0;i<2;i++)for (j=0;j<2;j++) *(a[0]+i*2+j)=1;
    */
}
```



嵌入式技术

邢超

计算机程序设
计语言

混合语言编程

思考

- 扩展
 - 速度
 - 系统调用
- 嵌入
 - 灵活
 - 方便



嵌入式技术

邢超

计算机程序设
计语言

混合语言编程

思考

- C 调用 Lua 函数
- 回调函数的实现



```
int main (void){
    char buff[256];
    int width,height;
    lua_State *L = lua_open();
    luaL_openlibs(L);
    luaL_loadfile(L, "lua.txt");
    lua_getglobal(L, "width");
    lua_getglobal(L, "height");
    width = (int)lua_tonumber(L, -2);
    height = (int)lua_tonumber(L, -1);
    printf("width is %d,height is %d.\n",width,height);
    lua_getglobal(L, "f"); /* function to be called */
    lua_pushnumber(L, 1.0);
    lua_pushnumber(L, 2.0);
    lua_pcall(L, 2, 2, 0);
    printf("double: %f%f\n",
        lua_tonumber(L, -2), lua_tonumber(L, -1));
    lua_close(L);
    return 0;
}
```

Lua 程序 lua.txt



嵌入式技术

邢超

计算机程序设
计语言

混合语言编程

思考

```
width = 200  
height = 300
```

```
function f(x1,x2)  
return x1,x2  
end
```




嵌入式技术

邢超

计算机程序设计语言

混合语言编程

思考

```
static int f(lua_State *L){
    double a = lua_tonumber(L, -1);
    lua_pushnumber(L, a);
    return 0;
}

int main(void){
    lua_State *L=lua_open();
    lua_register(L, "f", f);
    double a = 1;
    char *p = "f(a)";
    lua_pushnumber(L, a);
    lua_setglobal(L, "a");
    luaL_loadstring(L, p);
    lua_pcall(L, 1, 1, 0);
    lua_close(L);
    return 0;
}
```



嵌入式技术

邢超

计算机程序设
计语言

混合语言编程

思考

- C 调用 Scheme 函数
- 回调函数的实现



```
#include <stdio.h>
#include <stdlib.h>
#include "dynload.h"

int main(int argc, char *argv[])
{
    scheme scmenv;
    scheme_init(&scmenv);
    scheme_set_output_port_file(&scmenv, stdout);
    scheme_load_string(&scmenv,
        "(display(+123456))(newline)" );
    scheme_deinit(&scmenv);
    exit(EXIT_SUCCESS);
}
```



嵌入式技术

邢超

计算机程序设计语言

混合语言编程

思考

```
pointer square(scheme *sc, pointer args) {
    if (args!=sc->NIL) {
        if (sc->isnumber(sc->pair_car(args))) {
            double v=sc->rvalue(sc->pair_car(args));
            return sc->mk_real(sc, v*v);
        }
    }
    return sc->NIL;
}
```

Foreign functions are defined as closures:

```
sc->interface->scheme_define(
    sc,
    sc->global_env,
    sc->interface->mk_symbol(sc, "square"),
    sc->interface->mk_foreign_func(sc, square));
```



嵌入式技术

邢超

计算机程序设计语言

混合语言编程

思考

- Building more powerful C/C++ programs.
- Rapid prototyping and debugging.
- Systems integration.
- Construction of scripting language extension modules.

Compared with interface definition language (IDL)



嵌入式技术

邢超

计算机程序设
计语言

混合语言编程

思考

- ANSI C/C++ syntax.
- SWIG is not a stub generator.
- SWIG does not define a protocol nor is it a component framework.
- Designed to work with existing C/C++ code.
- Extensibility.



```
/* File : example.c */
#include <time.h>
double My_variable = 3.0;
int fact(int n) {
    if (n <= 1) return 1;
    else return n*fact(n-1);
}
int my_mod(int x, int y) {
    return (x%y);
}
char *get_time()
{
    time_t ltime;
    time(&ltime);
    return ctime(&ltime);
}
```



嵌入式技术

邢超

计算机程序设
计语言

混合语言编程

思考

```
/* example.i */
%module example
%{
/*Put header files here or function declarations*/
extern double My_variable;
extern int fact(int n);
extern int my_mod(int x, int y);
extern char *get_time();
%}

extern double My_variable;
extern int fact(int n);
extern int my_mod(int x, int y);
extern char *get_time();
```


Building a Tcl module



嵌入式技术

邢超

计算机程序设计语言

混合语言编程

思考

```
unix % swig -tcl example.i
unix % gcc -fpic -c example.c example_wrap.c \
      -I/usr/local/include
unix % gcc -shared example.o example_wrap.o \
      -o example.so
unix % tclsh
% load ./example.so example
% puts $My_variable
3.0
% fact 5
120
% my_mod 7 3
1
% get_time
Sun Feb 11 23:01:07 1996
%
```

Building a Python module



嵌入式技术

邢超

计算机程序设计语言

混合语言编程

思考

```
unix % swig -python example.i
unix % gcc -c example.c example_wrap.c \
      -I/usr/local/include/python2.1
unix % ld -shared example.o example_wrap.o \
      -o _example.so
```

We can now use the Python module as follows :

```
>>> import example
>>> example.fact(5)
120
>>> example.my_mod(7,3)
1
>>> example.get_time()
'Sun_Feb_11_23:01:07_1996'
>>>
```



嵌入式技术

邢超

计算机程序设计语言

混合语言编程

思考

```
unix % swig -perl5 example.i
unix % gcc -c example.c example_wrap.c \
    'perl -MExtUtils::Embed -e ccopts'
unix % ld -G example.o example_wrap.o \
    -o example.so
unix % perl
use example;
print $example::My_variable, "\n";
print example::fact(5), "\n";
print example::get_time(), "\n";
<ctrl-d>
3.0
120
Sun Feb 11 23:01:07 1996
unix %
```



嵌入式技术

邢超

计算机程序设
计语言

混合语言编程

思考

- 混合语言程序设计有哪些方法？
- 混合语言程序设计有哪些优缺点？