2012 年 12 月 北京航空航天大学学报 December 2012
第38卷 第12期 Journal of Beijing University of Aeronautics and Astronautics Vol. 38 No. 12

基于仿真器的嵌入式软件单元测试方法

许 福 金茂忠 刘 辉 陈 琦

(北京林业大学 信息学院, 北京 100083) (北京航空航天大学 计算机学院, 北京 100191)

摘 要: 介绍了一种基于仿真器的嵌入式软件测试方法,通过构造链接器分析目标文件,生成内存映像,加载到仿真器执行测试. 该方法特别适合嵌入式软件的单元测试,支持对不完整的代码进行测试,同时引入了交互式脚本执行机制,支持采用高级脚本语言而不是嵌入式专用汇编语言设计桩函数和测试用例,降低了测试人员的培训代价. 航天某院等单位的应用结果显示,该方法能有效提升嵌入式软件的单元测试效率,有较高的实际推广价值.

关 键 词: 嵌入式软件; 软件测试; 仿真器; 链接器; 单元测试

中图分类号: TP 311

文献标识码: A **文 章 编 号**: 1001-5965(2012)12-1686-05

Emulator-based embedded software unit testing method

Xu Fu

(School of Information Science and Technology, Beijing Forestry University, Beijing 100083, China)

Jin Maozhong Liu Hui Chen Qi

(School of Computer Science and Technology, Beijing University of Aeronautics and Astronautics, Beijing 100191, China)

Abstract: A unit testing method for embedded software was presented, which used customized linker to parse object files and generates memory image. The memory image can be loaded into a software simulator to generate testing results. This method is especially useful for the unit testing of embedded software, which can test incomplete code and support designing stub functions and test cases through advanced script programming language instead of embedded assembly programming language. Then the training cost of testing engineers can be reduced. The application result from one institute of China Aerospace Science and Industry Group shows that it can improve the efficiency of embedded software unit testing significantly.

Key words: embedded software; software testing; simulator; linker; unit testing

嵌入式软件广泛应用于航空航天、工业控制、交通管控等领域. 可靠性是这类软件的重要指标,因为一旦发生故障,往往会造成严重损失,2011年"7·23"甬温线特别重大铁路交通事故就是典型例子[1]. 因此,必须对这类软件执行严格的、有效的测试. 按实现方式不同,可将嵌入式测试工具分为3类.

1) 纯硬件形式. 常见的有在线仿真器[2]和逻辑分析仪[3]. 在线仿真器可以完全仿真目标芯片的行为,准确度高,运行速度快,但工艺设计复杂,价格昂贵. 逻辑分析仪通过监控和采样总线数据执行测试,完备性受限于采样频率,而提高采样频率则会大幅提高成本. 基于硬件的测试工具通常用于计算机硬件的设计和测试,很少用于嵌入式软件的测试.

2) 纯软件形式. 包括插桩[4]和软件仿真[5]两种典型方法. 插桩向被测程序的特定位置插入探针,据此获得程序的运行信息. 优点是实现简单,缺点是改变了被测程序,无法得到被测程序的精确执行轨迹和执行时间. 软件仿真通过构建目标机的虚拟仿真环境,在宿主机上加载待测程序执行测试(目标机指嵌入式软件的运行环境,宿主

收稿日期: 2012-07-30; **网络出版时间**: 2012-11-29 11:04

网络出版地址: www.cnki.net/kcms/detail/11.2625.V.20121129.1104.013.html

基金项目: 北京林业大学青年科技基金资助项目(2010BLX14);国家自然科学基金资助项目(60573084)

作者简介: 许 福(1979-),男,山东乳山人,讲师,xufu@buaa.edu.cn.

机指嵌入式软件的开发环境,下同). 优点是可获得待测程序的精确执行轨迹,缺点是速度较慢.

3) 软硬结合形式. 综合了纯硬件和纯软件形式的特点,代表工具是 CodeTEST[6]. 测试环境接近于真实环境,准确性较高. 缺点是硬件依赖性强,灵活性及可移植性差,成本昂贵.

理论上讲,嵌入式软件测试应在目标机进行,毕竟软件最终运行在目标机上. 但嵌入式系统多采用软硬件协同开发,在软件开发过程中,目标机环境很有限,甚至不可用,无法满足软件测试的需要. 此外,由于目标机的专用性,目标机下的测试工具比宿主机下的同类工具往往昂贵得多. 考虑到上述因素,应将尽可能多的测试工作放在宿主机进行. 但由于体系结构的差异,目标机软件一般不能在宿主机直接运行,这就需要建立目标机的仿真环境,即通过软件仿真进行测试.

软件仿真是一种经济灵活的测试策略,可在没有目标机硬件的情况下测试,准确性高,可以获取精确的运行时间,能灵活控制程序的运行状态. 采用软件仿真测试也可以避免将程序烧录到目标机 ROM 造成的测试困难,降低开发成本.

1　当前嵌入式测试工具的不足

不少嵌入式测试工具采用软件仿真测试[7-8]. 传统仿真工具的测试流程是:编译获得被测程序的可执行程序,加载到仿真器运行,获得测试结果. 这些工具对集成测试和系统测试支持较好,对单元测试的支持不佳.

例 1　对图 1 中 AD21060 芯片[9]的汇编代码进行单元测试时,若 sub1(第 7 行)和 sub2(第 9 行)未定义,必须用 AD21060 汇编语言编写桩函数[10]. 类似地,若被测单元引用了外部变量,也需要在被测源程序中声明. 否则,编译器会报错,不能生成可执行程序,从而无法加载到仿真器测试.

```
1   .SECTION/PM seg_pmco;
2   .global start,end;
3   function:
4           R1=5;
5           R2=6;
6           R3=7;
7           call sub1;
8           R1=R1+1;
9           call sub2;
10 end:
11          idle;
```

图 1　AD21060 汇编代码片段

通过例 1 可得到以下结论:①单元测试时很多模块尚未编写完毕,若采用传统的嵌入式软件测试方法,需要编写大量的桩函数,工作量大. ②测试者需熟悉嵌入式软件的开发语言,这样才能编写桩函数. 软件开发者通常会参与单元测试,这类人员可满足要求. 但为提高软件质量,一般需安排专门的测试人员测试. 嵌入式开发通常采用汇编等底层语言,种类繁多且较难学习,对测试者要求较高. ③修改桩函数和外部变量后,需重新编译生成可执行程序,重新加载到仿真器测试. 每做一点修改,均需重复上述步骤,测试过程繁琐. ④测试的针对性不强,只能加载整个可执行程序测试,无法针对特定的关注单元测试.

以航天某院为例,尽管购置了 CodeTEST[6] 和 Logiscope[11] 等昂贵的测试工具,但上述测试问题仍非常突出,其它工具也普遍存在类似的局限性. 本文介绍了一种新的嵌入式软件测试方法,可以弥补上述不足. 其特色和创新体现在 3 方面:

1) 通过自行构建链接器,而不采用缺省的链接器完成链接,生成“内存映像”(此处的“内存映像”与可执行程序类似,但并不完全一致,本文采用该术语以区分二者),可在程序代码不完备的情况下执行测试,能更好地支持单元测试.

2) 构建软件仿真器,加载内存映像测试. 除具备传统仿真测试的优点外,还增加了交互式脚本处理能力,运行过程中可直接修改桩函数和变量,无需频繁地编译、加载,测试更加便捷. 普通测试人员可以通过高级脚本语言编写测试用例,无需精通嵌入式平台的专用开发语言.

3) 能对指定的关注单元进行测试,支持设定源代码断点,支持设定仿真的起止范围,测试更有针对性,测试效率较高.

2　基于仿真器的嵌入式测试平台

图 2 给出了本文开发的测试平台 SimAD 的测试流程图. 其测试流程如下:

1) 编译被测程序生成目标文件(x86 平台的. obj 文件就属于典型的目标文件)和链接描述文件(LDF,Link Description File).

2) 利用自行构建的链接器分析目标文件和库文件,提取符号表、数据段、代码段、可重定位段、行号信息等,按照 LDF 文件的说明进行链接,生成内存映像、总符号表和行号信息表.

3) 将内存映像加载到仿真器测试. 测试过程

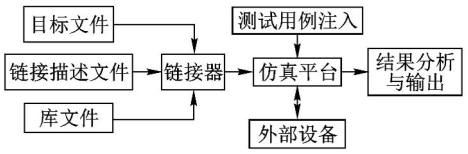


图 2　SimAD 测试流程图

中,可以修改内存变量、寄存器值、桩函数等. 结合测试用例注入,可实现单元测试和动态测试.

2.1 链接器构造

链接器根据 LDF 文件的说明,提取并拼装目标文件中的各个段,得到内存映像、符号表和行号信息表(图 3). 其中,内存映像和可执行程序功能类似;符号表定义了各个符号(包括函数、变量、标号等)的名称、地址、文件、作用域等信息;行号信息定义了源文件中行号与内存映像中地址的映射关系. 关于行号信息的更多介绍,请参考 2.3 节.

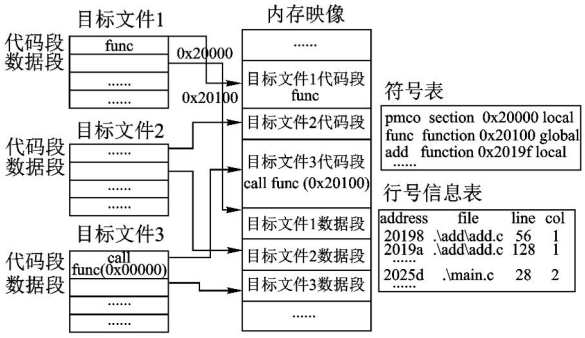


图 3 链接过程示意图

目标文件中的未定义符号在链接时会以 UNDEF 标记,只分析本文件无法解析,需要进一步分析其它文件. 如果在所有文件中都没有找到该符号,该符号就无法解析,使用标准的链接器无法链接,不能生成可执行程序进行仿真测试.

SimAD 自行设计了链接器,可以有效控制链接过程. 对于未定义符号,自动分配特殊的内存地址,使得这些符号可以通过“解析”,进而构造可被仿真器执行的内存映像. 这是实现本文测试方法的关键所在. 这样的设计思路有以下优点:

1) 单元测试时很多模块不完备,可避免使用嵌入式开发语言编写大量桩函数. 即使存在未定义符号,也可以正常链接,提升了测试效率.

2) 仿真器执行时可以检测到未定义符号,并检查是否定义了桩函数或者进行了变量赋值. 这样,就把桩函数和外部变量的绑定由编译期调整到了运行期. 这带来了 2 个好处:①可通过脚本语言设计桩函数,进行变量赋值,避免了学习嵌入式开发语言的代价. ②可实现交互式的桩函数声明和变量赋值,避免了传统的“修改桩函数和外部变量→编译→加载运行”的繁琐测试流程.

2.2 仿真器实现

仿真器完成对目标机的模拟,其基本特征是行为等价,注重对目标机行为功能的仿真,对内部结构的仿真要求不高. 重点关注:真实性、全面性、运行速度、通用性和可扩展性等指标.

SimAD 仿真器实现了以下模拟:①译码,包括运算指令译码、移位指令译码、地址寄存器译码、通用寄存器译码等;②运算模拟,包括加法器模拟、乘法器模拟及移位器模拟;③控制流处理,包括中断、循环、跳转、调用等;④地址产生模拟;⑤内存访问模拟;⑥I/O 模拟,包括外部内存方式和串口 DMA 方式;⑦定时器模拟.

在实现功能模拟的同时,仿真器还提供了测试接口和外部交互接口,包括内存映像的加载接口、内存和寄存器的查看/修改接口、桩函数的设定/修改接口、数据的输入/输出接口等.

对单元测试的良好支持是 SimAD 仿真器的特色之一. 单元测试中不可避免会用到桩函数,仿真器提供了 setstub 桩函数设置接口. 设置桩函数后,仿真器查询其内存地址,连同函数名一并保存到桩函数表中. 在运行过程中,仿真器检查调用的函数名是否存在于桩函数表中,若存在则不执行内存映像中的函数代码,而转去执行测试脚本中设定的桩函数. 为了便于理解上述过程,请参考例 2.

例 2 图 4 是一个 SimAD 测试脚本示例. 其中,第 11 ~ 13 行代码将函数 set1,set2 和 add3 设为桩函数,第 19 行启动仿真器. 仿真器遇到桩函数时会暂停运行,并返回一个特定值. 第 20 行将返回值保存在 result 中. 第 22 ~ 33 行通过仿真器的 getstub 命令查询遇到了哪个桩函数,并执行桩函数对应的测试脚本. 假设仿真器遇到了 set1 桩函数,则执行第 24 ~ 25 行,即输出“set1”字符串,并将寄存器 R1 的值设为 6.

```
10  # 指定桩
11  simcore setstub set1
12  simcore setstub set2
13  simcore setstub add3
14
15  # 设定起始结束地址
16  set startaddr [simcore getsymaddr start]
17  set endaddr [simcore getsymaddr end]
18
19  set result [simcore execute $startaddr $endaddr]
20  set result [simcore continue]
21  while {$result} {
22      switch  [simcore getstub] {
23        stub_set1 {
24          puts "set1"
25          simcore setreg R1 6
26        }
27        stub_set2 {
28          puts "set1"
29          simcore setreg R2 7
30        }
31        stub_add3 {
32          puts "add1"
33          simcore setreg R3 [expr [simcore getreg R1]*[simcore getreg R2]]
34        }
35      }
36      set result [simcore continue]
37  }
```

图 4 桩函数实例

2.3 对关注单元的测试

对关注单元的测试是 SimAD 仿真器的另一个特色,通过 2 种手段实现.

1) 利用“set startaddr”和“set endaddr”设置

仿真执行的起止地址. 考虑到直接输入地址不方便,可通过“getsymaddr”查询符号的实际地址. 比如,图 4 中的第 16 ~ 17 行,将标号为“start”和“end”间的代码设置为仿真运行代码.

2) 为仿真器添加断点运行机制. 实现该功能需要获取源程序中语句行与内存映像中指令地址的对应关系,这种对应关系称为行号信息.

SimAD 仿真 AD21060[9] 芯片,其目标文件格式是 ELF[12],行号格式是 DWARF[13],保存在目标文件的. debug_line 段中. 通过分析该段,可解析出源程序行号和指令地址的映射关系. 仿真器运行时,检查执行的指令地址处是否设置了断点. 若设置了断点,暂停仿真,显示被测单元的详细信息,如寄存器值、内存变量值等.

借助行号信息,还可以支持动态测试. 仿真器可以记录执行过的指令地址,通过查询地址与源码行的对应关系,可计算出覆盖了哪些代码行,进而计算出语句覆盖率. 语句覆盖率是单元测试的重要指标. 对于嵌入式应用,尤其是高可靠性的嵌入式应用,一般要求语句覆盖率达到 100%.

图 5 为某次仿真后的语句覆盖率着色情况,其中已覆盖的语句和未覆盖的语句在软件中会分别着色为绿色和红色. 为便于使用,SimAD 可对单次仿真结果着色,也可以对多次仿真结果合并着色.

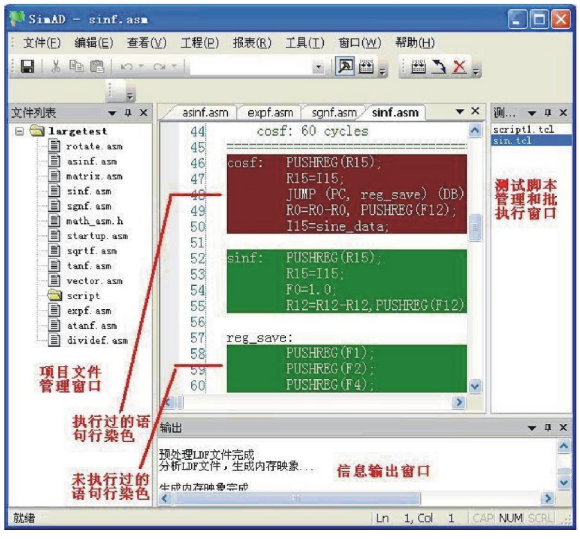


图 5　SimAD 语句覆盖率测试

3　SimAD 测试工具介绍

本文方法已在 SimAD 测试工具中实现. 该工具目前支持 AD21060 和 AD21020[9] 2 类芯片,可实现静态分析和动态测试. 已实现的功能包括集成测试环境(图 6)、链接器、仿真器、测试脚本解释执行、测试报表生成等.

对测试脚本的良好支持是 SimAD 的特色. 支持新建和保存测试脚本,双击运行测试脚本(图 5 中的右侧窗口),批量执行测试脚本,从而提高了回归测试[10]的执行效率. SimAD 还提供了控制台交互窗口,支持在仿真过程中输入交互式的测试命令,即时控制测试过程(图 7).

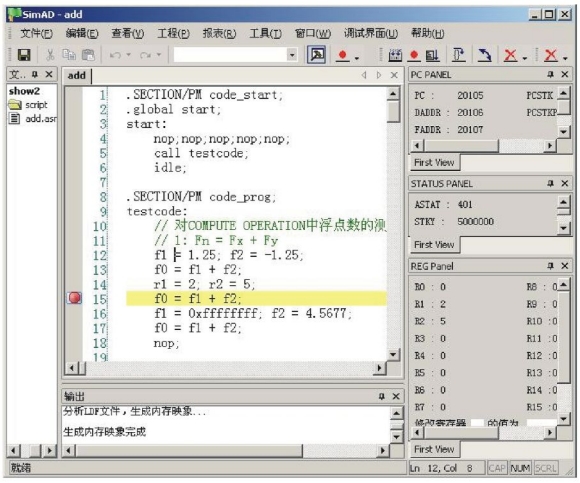


图 6　SimAD 集成测试环境

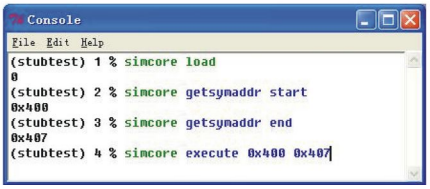


图 7　交互式执行测试命令

SimAD 还支持其它测试功能,如语句覆盖率报表、控制流着色、函数调用图生成等,这些措施对提高测试效果也有裨益.

4　相关系统比较

在嵌入式软件测试领域,国内外使用较多的测试工具是 CodeTEST[6] 和 Logiscope[11]. 它们与 SimAD 的比较见表 1.

3 种工具中 CodeTEST 功能最强. 动态内存分析和性能分析是其主要特点,采用软硬件结合形式实现,准确度较高,缺点是价格昂贵. 使用 CodeTEST 必须具备目标机硬件,必须生成执行程序,对单元测试的支持不佳.

Logiscope 主要用于软件的质量分析和测试,其突出特点是对软件全生命周期的支持. 采用插装方式测试,对执行速度和运行结果会造成一定影响,无法获得精确的执行时间和运行轨迹.

SimAD 通过软件仿真测试,不依赖目标机硬件,自行构造的链接器可处理未定义符号,可通过高级语言脚本编写测试用例和桩函数,对单元测试的支持较好.

表 1 常用嵌入式软件测试工具与 SimAD 的比较

条目	CodeTEST	Logiscope	SimAD
主要功能	①性能分析:最大、最小及平均执行时间,执行次数,函数调用次数; ②逻辑覆盖:语句覆盖、判定覆盖,判定/条件覆盖; ③存储器分配分析; ④追踪分析:分函数级、控制块级、源码级显示追踪情况,显示内存的分配情况.	①静态分析:根据编程规则检查源码; ②逻辑覆盖:语句覆盖、判定覆盖,判定/条件覆盖; ③综合测试.	①静态分析:可统计函数信息,生成函数调用关系图、控制流图; ②性能分析:程序执行时间的精确统计; ③逻辑覆盖:语句覆盖.
实现方式	软硬件结合	插装	软件仿真
突出特点	性能分析,内存泄漏分析	静态分析,覆盖分析	单元测试,交互式脚本测试
适用阶段	单元、集成、系统测试	软件开发各阶段	单元、集成、系统测试
精确程度	较高	中	高
灵活性	差	好	好
可移植性	差	好	好
成本	高	中	中
支持 CPU	支持所有 64/32/16bit CPU 和 MCU	语言编译器支持的处理器	目前只支持 AD21060,AD21020

5 结束语

目前的嵌入式软件测试工具对单元测试的支持不佳,本文对此进行了研究,提出了相应的解决思路,相关方法已在 SimAD 测试工具中实现,并在航天某院的项目测试中得到实际应用. 与成熟的嵌入式软件测试工具相比,SimAD 仍有很多地方需要完善. 未来的主要工作包括:①完善仿真器和测试框架的接口,方便第三方仿真器接入,以支持更多的嵌入式平台软件测试;②针对 COFF[14] 和 PE 等常用目标文件开发链接器,使之能分析链接多种平台的目标文件;③增加自动生成测试脚本功能,目前纯手工编写脚本,使用不太方便;④增强覆盖率分析,目前只支持语句覆盖,需进一步支持判定覆盖、条件覆盖等.

参考文献(References)

[1] 国家安全生产监督管理总局. "7·23"甬温线特别重大铁路交通事故调查报告[EB/OL]. 北京:国家安全生产监督管理总局,2011[2012-06-15]. http://www.chinasafety.gov.cn/newpage/Contents/Channel_5498/2011/1228/160577/content_160577.htm
State Administration of Work Safety of China. The investigation report of 7·23 Yong-wen major rail accident [EB/OL]. Beijing: State Administration of Work Safety of China Official Website, 2011[2012-06-15]. http://www.chinasafety.gov.cn/newpage/Contents/Channel_5498/2011/1228/160577/content_160577.htm (in Chinese)

[2] 薛伟. 利用在线仿真器(ICE)设计微机故障自动诊断系统[J]. 小型微型计算机系统,1988,9(6):40-44
Xue Wei. The microcomputer automatic fault diagnosis system based on circuit emulator[J]. Mini-Micro Systems, 1988, 9(6): 40-44 (in Chinese)

[3] 师奕兵,王厚军. 高速逻辑分析仪产品化设计的关键技术[J]. 仪器仪表学报,2002,23(5):38-40
Shi Yibing, Wang Houjun. Key techniques in product design of high-speed logic analyzer[J]. Chinese Journal of Scientific Instrument, 2002, 23(5): 38-40 (in Chinese)

[4] 李跃飞,郭君红,白成刚,等. 飞行控制软件测试中的插桩技术[J]. 北京航空航天大学学报,2009,35(5):580-583
Li Yuefei, Guo Junhong, Bai Chenggang, et al. Instrumentation in flight-control software testing[J]. Journal of Beijing University of Aeronautics and Astronautics, 2009, 35(5): 580-583 (in Chinese)

[5] 周庆,刘斌,余正伟,等. 综合模块化航电软件仿真测试环境研究[J]. 航空学报,2012,33(4):722-733
Zhou Qing, Liu Bin, Yu Zhengwei, et al. A framework of simulation testing environment for integrated modular avionics software [J]. Acta Aeronautica et Astronautica Sinica, 2012, 33(4): 722-733 (in Chinese)

[6] 吴晓葵. 基于 CodeTEST 的嵌入式软件测试技术[J]. 电子设计工程,2010,18(9):74-76
Wu Xiaokui. Embedded software testing technology based on CodeTEST[J]. Electronic Design Engineering, 2010, 18(9): 74-76 (in Chinese)

[7] Rihar Marjan. Software simulator as an effective tool for testing control algorithms[J]. Simulation, 1994, 63(1): 6-14

[8] Anuradha P S, Kumar K S S, Arindam C. Commercial dynamic software satellite simulator[J]. Journal of Spacecraft Technology, 2010, 20(2): 1-8

[9] Analog Devices Corporation. ADSP-21XX processor architectural overview[EB/OL]. Norwood: Analog Devices Corporation, 2012 [2012-05-20]. http://www.analog.com/en/processors-dsp/products/adsp21xx_processor_architectural_overview/fca.html

[10] 路晓丽,葛伟,龚晓庆,等. 软件测试技术[M]. 北京:机械工业出版社,2007:32-46
Lu Xiaoli, Ge Wei, Gong Xiaoqing, et al. Software testing technology[M]. Beijing: China Machine Press, 2007: 32-46 (in Chinese)

[11] IBM Corporation. Rational logiscope[EB/OL]. New York: IBM Corporation, 2012 [2012-06-20]. http://www-01.ibm.com/software/awdtools/logiscope/

[12] Wikipedia. Executable and linkable format[EB/OL]. London: Wikipedia, 2012 [2012-06-25]. http://en.wikipedia.org/wiki/Executable_and_Linkable_Format

[13] Wikipedia. DWARF [EB/OL]. London: Wikipedia, 2012 [2012-06-25]. http://en.wikipedia.org/wiki/DWARF

[14] Wikipedia. The common object file format[EB/OL]. London: Wikipedia, 2012 [2012-06-30]. http://en.wikipedia.org/wiki/COFF

(编 辑:文丽芳)