第 47 卷第 4 期
2015 年 8 月

南 京 航 空 航 天 大 学 学 报
Journal of Nanjing University of Aeronautics & Astronautics

Vol. 47 No. 4
Aug. 2015

DOI:10.16356/j.1005-2615.2015.04.005

# 基于 AADL 的综合模块化航空电子通信调度分析与建模

孙毅刚[1] 李 浩[1] 赵 珍[1] 樊智勇[2]

(1. 中国民航大学航空自动化学院,天津,300300;
2. 中国民航大学工程训练中心,天津,300300)

**摘要**:针对具有最终配置信息的综合模块化航空电子(Integtated modular avionics, IMA)系统的通信调度问题,分析了 IMA 系统网络中通信调度流的产生、特性以及表示方式,并通过实例分析通信调度的生成表示过程。在此基础上,基于架构分析与设计语言(Architecture analysis and design language, AADL),对实例中的通信流进行建模,并进行实例化,得到端口连接一致性检测报告,验证了文中通信流模型的可行性和有效性。

**关键词**:综合模块化航空电子;通信调度;通信流;架构分析与设计语言

**中图分类号**: V556.7 **文献标识码**: A **文章编号**:1005-2615(2015)03-0497-11

# Analysis and Modeling of Communication Schedule for Integrated Modular Avionics Based on AADL

*Sun Yigang*[1], *Li Hao*[1], *Zhao Zhen*[1], *Fan Zhiyong*[2]

(1. College of Aerospace Automation, Civil Aviation University of China, Tianjin, 300300, China;
2. Engineering Training Center, Civil Aviation University of China, Tianjin, 300300, China)

**Abstract**: Aiming at the communication schedule problem of integrated modular avionics(IMA) with its final configuration, the generation, characteristics, and representation of communication schedule flows are analyzed, and an instance is given to show the implementation process of communication schedule. Based on the architecture analysis and design language (AADL), a model for communication flows is built for the previous instance. Finally, the model is instantiated, and a check-port connection consistency report is obtained for consistency verification of modeling communication flows. Simulation result illustrates the feasibility and validation of the communication flow model.

**Key words**: integrated modular avionics; communication schedule; communication flows; architecture analysis and design language

综合模块化航空电子(Integrated modular avionics, IMA)系统的通信调度是符合 ARINC653 标准的 IMA 分区模块中任务调度的细化,它能够表现 IMA 系统调度的底层调度行为,以及 IMA 系统底层的配置信息。ARINC653 标准中定义的空间分区、时间分区以及分区调度和任务调度,是 IMA 系统中的一项重要技术,为综合化航空电子系统的安全性及可靠性提供了技术保障。但是,由于 IMA 系统调度的复杂性,增加了对其进行测试与验证的难度。针对 IMA 系统通信调度进行分析,是从 IMA 调度的底层调度着手进行分析,为 IMA 系统通信的测试与验证提供技术支持。

**基金项目**:国家重点基础研究发展计划("九七三"计划)(2014CB744904)资助项目。

**收稿日期**:2015-06-15;**修订日期**:2015-07-15

**作者简介**:孙毅刚,男,教授,主要研究方向:航空电子系统测试与适航验证及机场运行控制与安全保障技术。

**通信作者**:孙毅刚,E-mail: ygsun@cauc.edu.cn。

目前,针对IMA系统的通信调度、测试与验证问题已经展开了大量研究。2004年,德国Bremen大学的Tsiolakis最先提出了一种通信流的生成方法[1],他将IMA系统通信调度理解为包含WRITE_PORT、READ_PORT和TIMED_WAIT集合的时间痕迹。2007年,德国Bremen大学的Ott在其博士论文中研究了IMA网络通信流数据的生成和表示方法[2]。2011年,Efkemann针对IMA测试开发了一种特定领域的建模语言[3],这种语言能够为测试自动建立不同层级的模型。

中国国内对IMA平台网络测试也做了一些相关研究工作,多数是针对IMA分区调度的研究或优化,如:航空电子分区隔离的建模与设计、实时任务的双层调度以及核心处理安全分区的优化设计[3-5]、合理利用空闲时间片提高系统的时间利用率[6-8],以及基于分区的航空电子系统调度分析工具的实现[9]等。但是,对于IMA调度的基础性工作——IMA系统通信调度的过程分析与描述尚缺乏一些深入研究。

本文主要针对IMA通信调度进行分析与建模,研究了通信调度的生成、特性以及通信流的表示方法,通过将IMA系统通信调度的配置信息、性能与架构分析与设计语言(Architecture analysis and design language, AADL)的构件及属性相映射,得到IMA系统通信调度流的描述模型,试验结果验证了该模型与实际通信调度的一致性。该模型的建立能够为后续IMA系统通信调度测试与验证提供模型基础。

## 1 IMA通信

### 1.1 IMA平台概述

IMA模块可以称为核心处理以及I/O资源(Core processing and I/O module, CPIOM),能够为多个应用程序提供通用计算资源和各种接口。作为共享资源,IMA模块能够驻留多个应用,每个应用使用一个或多个分区来执行其任务。

分区是IMA模块的核心逻辑单元,它们被指派具体的应用,且有着各自的存储空间和专门的处理时间片。IMA平台操作系统通过提供分区间或模块间的通信方式来支持驻留在同一个或不同模块上分区间的协作。资源共享(CPU时间、内存、通信方法)的前提是:时间和空间上的分离能够保证功能的实现相互独立,即一个功能的实现并不会干扰到其他功能的实现。

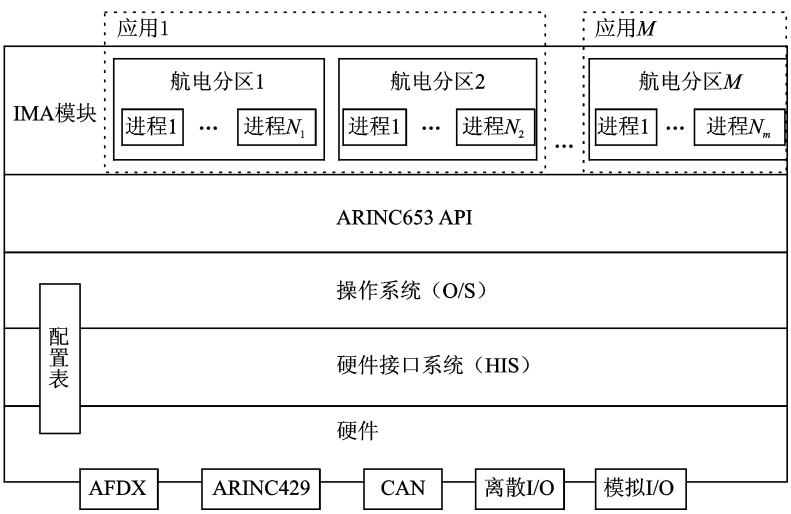


图1 IMA平台示意图

Fig. 1 Sketch map of IMA platform

图1为IMA平台示意图。由图1可见,IMA平台结构由以下几个部分组成:(1)应用软件包含在相应的IMA模块的航空电子分区中,用于执行应用程序。(2)操作系统中标准ARINC653 API提供的服务,被航空电子应用程序用来执行任务以及通信。(3)操作系统(Operating system, O/S),管理分区间以及分区内的通信。在分区层,操作系统管理着分区进程;在通信、调度、内存管理、时间以及健康监测方面,操作系统与硬件接口系统相连接。(4)硬件接口系统(Hardware interface system, HIS),由包含硬件接口的接口驱动器集合组成,提供访问存储器管理单元(Memory management unit, MMU)和时钟的方法。(5)IMA模块的硬件,包括硬件接口、处理器、硬件时钟、内

存管理单元以及内存。(6)配置表，被操作系统以及硬件接口系统用于配置内存的访问、调度和通信。

### 1.2　IMA 调度理论

航空电子标准 ARINC653 中使用分区作为调度、资源分配及对应用进行隔离保护的单位。分区包括空间分区和时间分区。空间分区即每个分区都有独立的地址空间，利用 MMU 为每个分区建立不同的虚拟地址到物理地址的映射，使每个分区都有独立的、确定的物理存储空间，并且该物理空间仅为该分区所有，使分区在空间上相互独立，从而在空间上对分区进行保护。时间分区即各个分区都按照一个确定的周期被调度，由操作系统维护一个固定时间长度的主时间框架(Major time frame，MAF)，各分区间没有优先级高低之分。MAF 在系统运行过程中周期重复，每个 MAF 可以划分为若干个时间窗口，每个分区在 MAF 内至少拥有一个时间窗口，各分区只能在其自身的时间窗口中被调度，所以，即使一个分区发生错误，基于 ARINC653 的操作系统根据 MAF 仍能调度其他分区继续执行。

ARINC653 标准定义的 IMA 构架中，采用时间分区的双层调度策略来执行分区的应用程序，即操作系统采用使用时间片轮转的方式激活每个分区，之后根据分区内定义的策略调度任务进程。ARINC653 航空电子分区操作系统含有两级调度：分区级调度和任务级调度。分区级调度是指系统为分区分派时间片的过程；任务级调度是指分区下的任务集合在所属分区获取时间片时的执行过程。

### 1.3　IMA 通信

某个分区的进程能够和相同模块上的其他分区上的进程、驻留在其他 IMA 模块上的分区进程及非 IMA 控制器以及外部设备进行通信。对于模块间通信来说，信息的接收与发送是通过操作系统与相应的接口驱动之间进行交互来实现的。对于分区间通信来说，操作系统管理着相应的内存区间，且这些内存区间映射到相应的通信接口上。然而，操作系统不能区分模块间通信和分区间通信，而是提供一种统一的访问机制。

分区通信(或者更精确地说，它们的进程)是由操作系统提供的用于接收和发送信息的服务来实现的。为了支持应用的可移植性，采用了一种统一的端口概念，将端口内部映射于接口通道。如此，用于分区间通信的 I/O 接口对于应用来说是透明的，并且由配置表来决定。在应用程序编程接口(Application programming interface，API)级的信息仅仅包含有效负载，并不包含像接收方、发送方一类的路由信息。如果这类信息被一个特殊应用需要，则需要在应用级的协议上保证发送方附着此类信息。

下面通过一个实例来说明 IMA 模块间的通信过程，如图 2 所示。图 2 为 IMA 通信中位于不同模块 A 及 B 上的不同分区的两个端口之间的通信过程。

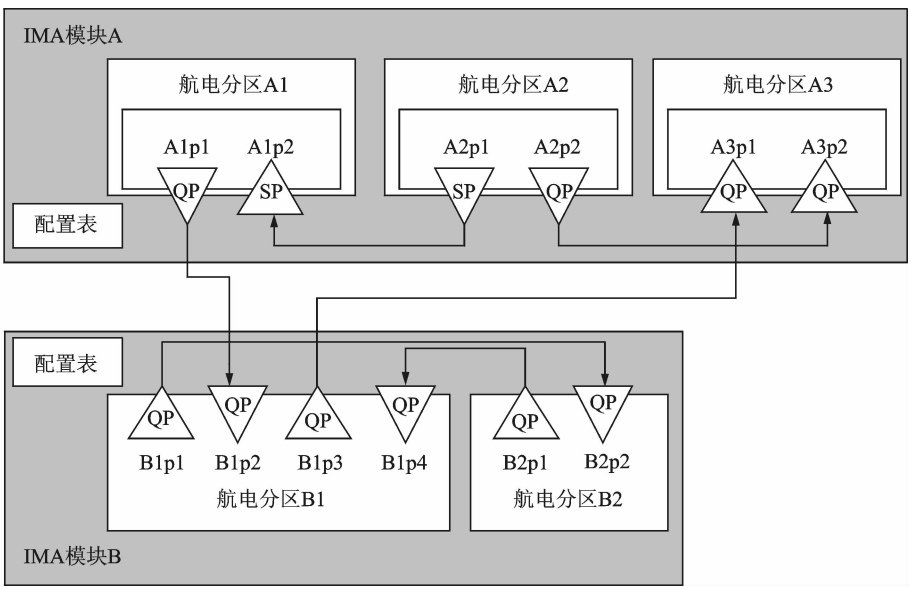


图 2　IMA 模块间通信过程

Fig. 2　Communication process between two IMA modules

已知条件如表 1 所示。模块内分区上的端口名称用“A1p1，A1p2，…，B2p1，B2p2”表示，这些端口类型可能是采样的或是队列的，其中采样端口用“SP”表示，队列端口用“QP”表示，在图 2 中，端口名称标注在端口下的三角形内，其中，最差情况下的软件延时(Worst-case software latency，WCSL)，对应于 API 调用的执行时间；最差情况下的硬件延时(Worst-case hardware latency，WCHL)，对应于 I/O 驱动用于接收和发送数据所需时间；最差情况下传输延时(Worst-case transmission time，WCT)，对应于信息在传输线路上所用时间。使用 WRITE_PORT 及 READ_PORT 来分别命名写、读 API 调用。

**表 1 示例分区、端口信息**

**Tab. 1 Partition and port information for the above instance**

| (A，B)模块配置信息 | (A，B)模块性能信息 |
|---|---|
| A1p1 属于 A 模块分区 | WCSL(A1p1)=2 |
| A1p1 端口为源端口 | WCSL(B1p2)=3 |
| B1p2 属于 B 模块分区 | WCHL(A1p1)=1 |
| B1p2 端口为目标端口 | WCHL(B1p2)=1 |
| | WCT(A1p1，B1p2)=2 |

端口之间通信的流程如图 3 所示。首先，由处于调度状态下的分区选择被调度的源端口，接着分区执行相应的 API 调用，这些 API 调用是用于发送信息的，此时的分区为 BUSY 状态。信息经由 I/O 驱动，通过通信连接送往目标端口，目标端口相应的 I/O 驱动接收信息。最后，等待目标端口的分区被调度，执行相应的用于接收信息的 API 调用。

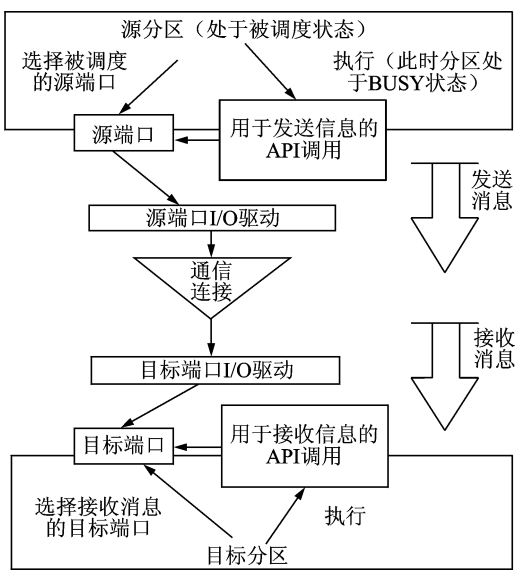


图 3 端口之间的通信流程

Fig. 3 Inter-port communication process

通信详细过程如图 4 所示。在对 IMA 平台网络进行调度分析的时候，给模块或是分区分配的调度时间单位均为 Time unit。

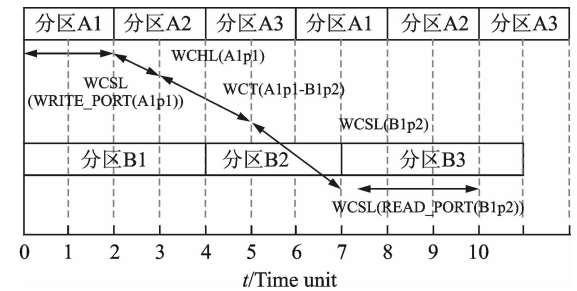


图 4 端口之间通信详细过程

Fig. 4 Detailed communication process between two ports

## 2 IMA 通信调度分析

### 2.1 IMA 通信调度的定义

对于特定分区来说，每个通信调度是一种接口触发的时间序列，其中，接口触发是用来发送和接收消息的 API 调用，这些接口触发执行时间的长短依赖于端口和消息的特性。整个 IMA 网络的通信调度是各个分立的通信调度的交叉，这些分立的通信调度具有相同的参考起始点。特别的，就网络中的 IMA 模块来说，通信调度应该服从端口配置、每个模块上配置的分区调度以及每个模块类别的性能信息。

### 2.2 IMA 通信调度的特点

首先，每个通信调度描述了对象网络中的一种可能的通信流。通信流可以理解为：当选定一条通信链路时，哪条通信是并行或是顺序执行、何种类型的信号或是信息被传输。

其次，通信调度能够明确所有网络部件的通信行为，尤其是那些有为通信流配置端口的配置区间。通信调度必须遵照网络配置以及每个网络部件、网络技术提供的性能信息。

最后，通信调度的时间标记是时间戳，时间戳是单调递增的。考虑到接口可能的并行触发，整个通信调度是能够发生的各个模块间通信调度的交叉。

### 2.3 IMA 通信调度分析

对图 2 所示的模块间通信调度进行分析，已知条件如表 2 所示。

模块内部通信中，相应端口的 WCHL 和 WCT 均为 0，这是因为在模块内部通信中，不论是 I/O 驱动还是网络都不涉及这种通信。两个 IMA 模块以并行的方式执行各自配置的调度，即有着相同的分区调度起始时间点。对象的通信调度过程如图 5 所示。

**表 2　A，B 模块的详细信息**

**Tab. 2　Detailed information of modules A and B**

| (A，B)模块的配置信息 | (A，B)模块的性能信息 |
|---|---|
| 模块 A：<br>分区：A1，A2，A3；<br>通信端口类型：QP，SP；<br>调度次序：A1→A2→A3(按时间轮转)；<br>分区分配的时间单元：A1＝2，A2＝2，A3＝2。 | WCSL(A1p1)＝2 ，WCHL(B1p2)＝3<br>WCT(A1p1，B1p2)＝2<br>WCSL(A1p2)＝1，WCHL(A2p1)＝0<br>WCT(A2p1，A1p2)＝0<br>WCSL(A2p2)＝2，WCHL(A3p2)＝0<br>WCT(A2p2，A3p2)＝0 |
| 模块 B：<br>分区：B1，B2；<br>通信端口类型：QP；<br>调度次序：B1→B2(按时间轮转)；<br>分区分配的时间单元：B1＝4，B2＝3。 | WCSL(B1p1)＝4，WCHL(B2p2)＝0<br>WCT(B1p1，B2p2)＝0<br>WCHL(B1p3)＝2，WCHL(A3p1)＝2<br>WCT(B1p3，A3p1)＝1<br>WCHL(B2p1)＝0，WCHL(B1p4)＝0<br>WCT(B2p1，B1p4)＝0 |

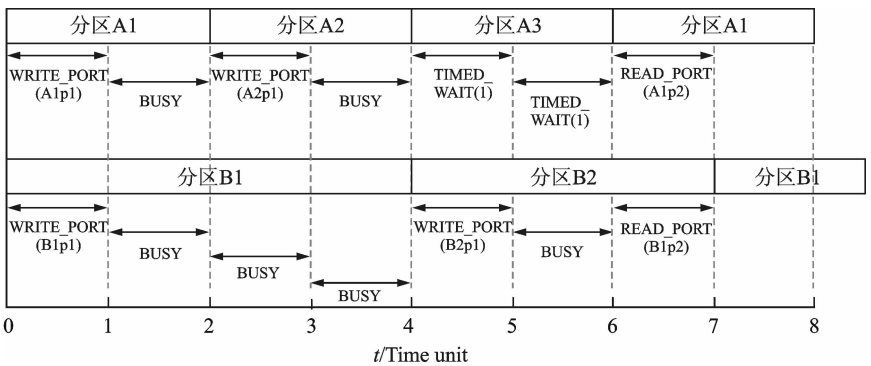


图 5　通信调度过程

Fig. 5　Communication schedule process

由图 5 可见，在 $t=0$ 时需要完成用于调度的 API 调用选择。对于模块 A 及 B，分别被选中的 API 调用是 WRITE_PORT(A1p1)及 WRITE_PORT(B1p1)。

$t=1$ 时，被选中的 WRITE_PORT (A1p1)及 WRITE_PORT(B1p1)处于执行状态。

$t=2$ 时，由于 WCSL(A1p1＝2)，即 A1 分区上执行的 API 调用已经完成，而且分区 A1 配置的时间单元为 2，因此，模块 A 在该时刻重新被调度。这时，对模块 A 的 A2 分区进行四步算法，选择 WRITE_PORT(A2p1)执行；对于模块 B 来说，由于 WCSL(A1p1＝4)，此时 WRITE_PORT(B1p1)仍在被执行，即为 BUSY 状态。

$t=3$ 时，由于 WSCL(A2p1)＝2 以及 WCSL(A1p1＝4)，给 B1 分区配置的时间单元为 4，那么此时两个均为 BUSY 状态。

$t=4$ 时，此时两个模块均需要被重新调度，因为 A2 及 B1 配置的时间以及 WSCL 均已满足。模块 A 中，按照正常调度顺序应该是对 A3 分区进行调度，但是考虑到 A3 分区上的端口均为目标端口，且 WRITE_PORT(B1p3)及 WRITE_PORT(A2p2)均还没被选择执行，那么，此时对于 A3 只能选择 TIMED_WAIT(1)；模块 B 中，B2 分区并没被限制，因此可以选 WRITE_PORT(B2p1)执行。

$t=5$ 时，模块 A 中，由于给 A3 分区配置的时间单元为 2，而这期间 A3 分区上的两个端口 API 调用还是不能被选择，只能继续执行 TIMED_WAIT(1)；模块 B 中，B2 分区仍忙于执行 WRITE_PORT(B2p1)，因此为 BUSY 状态。

$t=6$ 时，模块 A 中，按照顺序应该开始分区 A1 的调度了，由于在 $t=0$ 时已经选择过 WRITE_PORT(A1p1)了，而且考虑由 A1p1 发送的信息直到 B1p2 接收，信息的成功传送需要 WCSL(A1p1)＋WCHL(A1p1)＋ WCT(A1p1，B1p2)＋ WCHL(B1p2)＋ WCSL(B1p2)＝10，也就是说，$t=6$ 时，信息并没有成功传送，就不能再选择 WRITE_PORT(A1p1)；考虑 READ_PORT(A1p2)，在 $t=2$ 时分区 A2 上的经过端口 A2p1 发送的信息对于分区 A1 来说已经是可得的了：WCSL (A2p1) ＋

WCHL(A2p1)+WCT(A2p1, A1p2)=2,信息已经可以传输到A1分区了,所以READ_PORT(A1p2)是可以被选择的。而B模块中,分区B2配置的时间还剩1,而如果要选择执行WRITE_PORT(B2p1),由于WCSL(B2p1)+WCHL(B2p1)=2,已经超过了剩余调度时间,那么只能考虑READ_PORT(B2p2)了,由WCSL(B1p1)+WCHL(B1p1)+WCT(B1p1, B2p2)=4,即$t$=0时刻从端口B1p1发送的信息对于分区B2已经可得,而且WCSL(B2p2)+WCHL(B2p2)=1,与分区B2配置的时间剩余相等,那么,可以选择READ_PORT(B2p2)。

由以上分析,该实例的通信调度可描述如下:

<(0, {A: WRITE_PORT(A1p1), B: WRITE_PORT(B1p1)}),

(1, {A:BUSY(1), B:BUSY(3)}),

(2, {A: WRITE_PORT(A2p1), B: BUSY(2)}),

(3, {A:BUSY(1), B:BUSY(1)}),

(4, {A: TIMED_WAIT(1), B: WRITE_PORT(B2p1)}),

(5, {A: TIMED_WAIT(1), B: BUSY(1)}),

(6, {A:READ_PORT(A1p2), B:READ_PORT(B2p2)})>

## 3 IMA通信调度建模

### 3.1 AADL

AADL可用于设计和分析系统的软硬件结构,包括独立组件及其之间的交互,尤其适用于性能关键的实时嵌入式系统。该语言可用于分析和生成嵌入式实时系统语言和工具集的研究原型,由霍尼韦尔(Honeywell)技术中心开发,在20世纪90年代就已经成功地用于各种驾驶仪项目之中。

AADL是带有建模要素图形描绘的严谨的文本建模语言。它的重点在于物理系统体系架构及嵌入式应用软件运行时体系结构与计算机平台之间的交互。它提供了实现系统功能对软件影响的深入理解,并提供了对于应用运行时体系结构,诸如操作系统和通信协议一类的基础结构,以及在分布式计算机平台上的部署决策。

本文是在开源AADL工具环境(Open source AADL tool environment, OSATE)下进行建模与模型验证的。OSATE是由软件工程师协会基于平台Eclipse开发的AADL模型设计开发工具,它能够有效地编辑、调试AADL模型文件,并能图示化、实例化系统模型,且自带的插件能够对所建模型进行端口连接一致性检验。

### 3.2 基于AADL的IMA模块通信调度建模

使用AADL工具为IMA通信调度建模,可以分为两个部分。此处建模的对象是上部分中通信流分析的对象IMA(图2)。其中,配置信息和绩效信息已经在通信流分析中给出。

首先,对配置的IMA模块进行建模,包括模块调度及分区调度。文献[10,11]中详细介绍了模型语言AADL的构建、特征、关联及属性等基本建模元素。其中文献[12]中详细给出了配置信息转换为AADL模型的转换规则,并且给出了关于IMA系统建模的实例。SAE-5506/2(2011-01)标准的附件F(Annex F-ARINC653)[13]为ARINC653系统的建模、分析以及集成提供了向导,并且能够导出或得到相似的分区架构。本例中的IMA模块及分区调度的相关信息对应于AADL中的内容如表3所示。

**表3 ARINC653配置信息与AADL元素的转换规则**

**Tab.3 Transformation rule between ARINC653 configuration information and AADL elements**

| ARINC653标准中配置信息 | AADL中对应的组件与实现 |
|---|---|
| 模块A: | 组件:processor; |
| 模块总时间框架=12 ms; | Properties: ARINC653::Module_Major_Frame |
| 分区:A1, A2, A3; | 组件:vitual processor; |
| 通信端口类型:QP, SP; | Features: event data port; data port; |
| 调度次序:A1→A2→A3(按时间轮转); | Properties: ARINC653::Slots_Allocation; |
| 分区分配的时间单元:A1=2, A2=2, A3=2。 | Properties: ARINC653::Partition_Slots。 |

对模块B的分区调度建模也是一样的,只是参数有所改变,在这里不再给出。

其次,对IMA通信调度进行建模。通信调度的建模主要是对通信流的建模,即使用AADL工具将通信流在已建模的IMA系统中表示出来。

由于ARINC653API提供的服务能够被航空电子应用程序用来执行它们的任务及通信,也就是说,ARINC653API是服务于任务的执行和通信的,在AADL建模工具中,对应服务于任务的构件是线程,因此,本文在对ARINC653API建模时选择线程来对应API,并且将在通信流中具有性能信息的API特性,如WCSL及WCHL,作为线程属性添加到线程属性集中。具体的转换如表4所示。

**表 4　A，B 模块性能信息与 AADL 元素的转换规则**

**Tab. 4　Transformation rule between AADL elements and performance information of modules A and B**

| (A，B)模块的性能信息 | AADL 中对应的组件与实现 |
|---|---|
| WCSL(A1p1)=2；<br>WCHL(B1p2)=3；<br>WCT(A1p1,B1p2)=2；<br>调度协议：Periodic；<br>周期：2 ms；<br>计算执行时间：2 ms；<br>通信流表示：A1p1→B1p2。 | 组件：thread；<br>线程扩展属性：thread::WCSL；<br>线程扩展属性：thread::WCHL；<br>流属性：lantency；<br>线程周期属性：periodic；<br>线程属性：Compute_Execution_Time；<br>流特征：flow_path。 |

整个系统的实现如下：

system implementation IMA_system. impl——将两个 IMA 模块放在一个系统实现中整合

subcomponents

process_A1 : process process_A1. impl;

process_A2 : process process_A2. impl;

process_A3 : process process_A3. impl;

process_B1 : process process_B1. impl;

process_B2 : process process_B2. impl;

IMA_Module_A : processor IMA_Module_A. impl;——模块 A 的实现

IMA_Module_B : processor IMA_Module_B. impl;——模块 B 的实现

memory_A : memory ModuleA_memory. impl;——模块 A 的内存实现

memory_B : memory ModuleB_memory. impl;——模块 B 的内存实现

communication_channel : abstract communication_channel. impl;

connections——端口与端口之间的互连

cA1p1 : port process_A1. A1p1 → communication_channel. A1p1;

cB1p2 : port communication_channel. B1p2 → process_B1. B1p2;

cA2p1 : port process_A2. A2p1 → communication_channel. A2p1;

cA1p2 : port communication_channel. A1p2 → process_A1. A1p2;

cA3p1 : port communication_channel. A3p1 → process_A3. A3p1;

cA3p2 : port communication_channel. A3p2 → process_A3. A3p2;

cA2p2 : port process_A2. A2p2 → communication_channel. A2p2;

cB1p3 : port process_B1. B1p3 → communication_channel. B1p3;

B1p4 : port communication_channel. B1p4 → process_B1. B1p4;

B1p1 : port process_B1. B1p1 → communication_channel. B1p1;

B2p2 : port communication_channel. B2p2 → process_B2. B2p2;

B2p1 : port process_B2. B2p1 → communication_channel. B2p1;

properties——IMA 模块中进程与处理器的绑定，进程与内存的绑定，用于表示 IMA 模块中时间分区和空间分区的实现

Actual_Processor_Binding ⇒ (reference (IMA_Module_A. avionics_partition_A1)) applies to process_A1;

Actual_Processor_Binding ⇒ (reference (IMA_Module_A. avionics_partition_A2)) applies to process_A2;

Actual_Processor_Binding ⇒ (reference (IMA_Module_A. avionics_partition_A3)) applies to process_A3;

Actual_Processor_Binding ⇒ (reference (IMA_Module_B. avionics_partition_B1)) applies to process_B1;

Actual_Processor_Binding ⇒ (reference (IMA_Module_B. avionics_partition_B2)) applies to process_B2;

Actual_Memory_Binding ⇒ (reference (memory_A. memory_A1)) applies to process_A1;

Actual_Memory_Binding ⇒ (reference (memory_A. memory_A2)) applies to process_A2;

Actual_Memory_Binding ⇒ (reference (memory_A. memory_A3)) applies to process_A3;

Actual_Memory_Binding ⇒ (reference (memory_B. memory_B1)) applies to process_B1;

Actual_Memory_Binding ⇒ (reference (memory_B. memory_B2)) applies to process_B2;

end IMA_system. impl;

通过上述设置，整个系统的建模图形如图6所示。图中，带箭头的虚粗线分别表示进程与虚拟处理器和内存之间的绑定。当进程与内存之间进行绑定时，对应为ARINC653标准下的分区独立内存分配，表示分区空间上的独立；当进程与虚拟处理器之间进行绑定时，并给虚拟处理器附上调度时间，对应为ARINC653标准下的分区调度时间分配，表示分区时间上的独立。

### 3.3 模型验证

本文使用的OSATE版本为osate2-2.0.8-win32.win32.x86，且运行在win7系统下。此处建模的对象是上部分中通信流分析的对象——IMA平台网络(图2)。

模型检验步骤设计如下：

(1) 基于第1，2节部分的理论分析，从符合ARINC653标准的配置信息和IMA平台网络性能信息中获取建模所需要的建模信息；

(2) 依据AADL与ARINC653标准之间的转换规则对IMA平台网络进行建模，由于AADL属性集中对应于通信调度信息并没有定义，因此还需对其属性集进行扩展；

(3) 对建模后的模型进行端口一致性检验，此检验功能是由OSATE自带的。

IMA平台网络的建模信息在表5中给出，包括配置信息以及性能信息。

**表5 IMA平台网络的建模信息**

**Tab.5 Modeling information for IMA platform network**

| 配置信息 | 性能信息 |
|---|---|
| 模块A：<br>模块总时间框架=12 ms，<br>分区：A1，A2，A3；<br>调度次序：A1→A2→A3(按时间轮转)；<br>分区分配的时间单元：A1=2，A2=2，A3=2；<br>源端口：A1p1(QP)，A2p1(SP)，A2p2(QP)；<br>目标端口：A1p2(SP)，A3p1(QP)，A3p2(QP)；<br>Queue Size(A1p1，A2p2，A3p1，A3p2)=1，Timeout(A1p1，A2p2，A3p1，A3p2)=2 ms，<br>Queue processing protocol(A1p1，A2p2，A3p1，A3p2)=FIFO；Timeout(A1p2，A2p1)=2 ms。 | WCSL(A1p1)=2；<br>WCHL(B1p2)=1；<br>WCT(A1p1，B1p2)=2；<br>WCSL(A1p2)=1；<br>WCSL(A2p1)=2；<br>WCSL(A2p2)=2；<br>WCSL(A3p2)=1； |
| 模块B：<br>模块总时间框架=12 ms，<br>分区：B1，B2；<br>调度次序：B1→B2 (按时间轮转)；<br>分区分配的时间单元：B1=4，B2=3；<br>源端口：B1p1(QP)，B1p3(QP)，B2p1(QP)；<br>目标端口：B1p2(QP)，B1p4(QP)，B2p2(QP)；<br>Queue Size(B1p1，B1p2，B1p3，B1p4)=2，Timeout(B1p1，B1p2，B1p3，B1p4)=4 ms，<br>Queue processing protocol(B1p1，B1p2，B1p3，B1p4)=FIFO；<br>Queue Size(B2p1，B2p2)=1，Timeout(B1p1，B1p2)=3 ms，Queue processing protocol(B1p1，B1p2)=FIFO。 | WCSL(B1p1)=4；<br>WCSL(B2p2)=4；<br>WCSL(B1p4)=2；<br>WCSL(B2p1)=2；<br>WCSL(B1p3)=2；<br>WCSL(A3p1)=1；<br>WCHL(B1p3)=2；<br>WCHL(A3p1)=2；<br>WCT(B1p3，A3p1)=1。 |

建模后，通过OSATE工具中instantiate selected system.iml选项将模型实例化，这个主要是针对系统实现进行的，只有经过实例化的模型才能进行进一步得以分析和验证。实例化的过程就是将整个系统的进程、线程、处理器和总线等按层级关系组织成一个整体。经过实例化后，将得到一个格式为.aaxl的文档，模型的端口连接一致性将在此实例上进行。

“模型端口连接一致性”属于模型内部的一致性，它主要针对模型本身。在AADL建模中，当组件的特征组被定义了以后，它们将包含端口。对于每个端口来说，在模型实例化以后，会得到一个单独的连接实例，模型端口连接一致性检验主要是针对定义在特征组的每个端口属性的前后一致性。对于每个端口连接实例来说，端口连接一致性检查器将从源端口和目标端口提取属性参数进行比较，验证其参数是否一致。AADL属于层次化建模，对于一个有着“process”和“system”组件的模型来说，底层的连接实例是起源或者终止于“process”或“system”端口，那么，如果将模型细化到“thread”层级，则端口连接的实例将在“thread”端口之间建立，因此需要比较不同层级之间，源端口

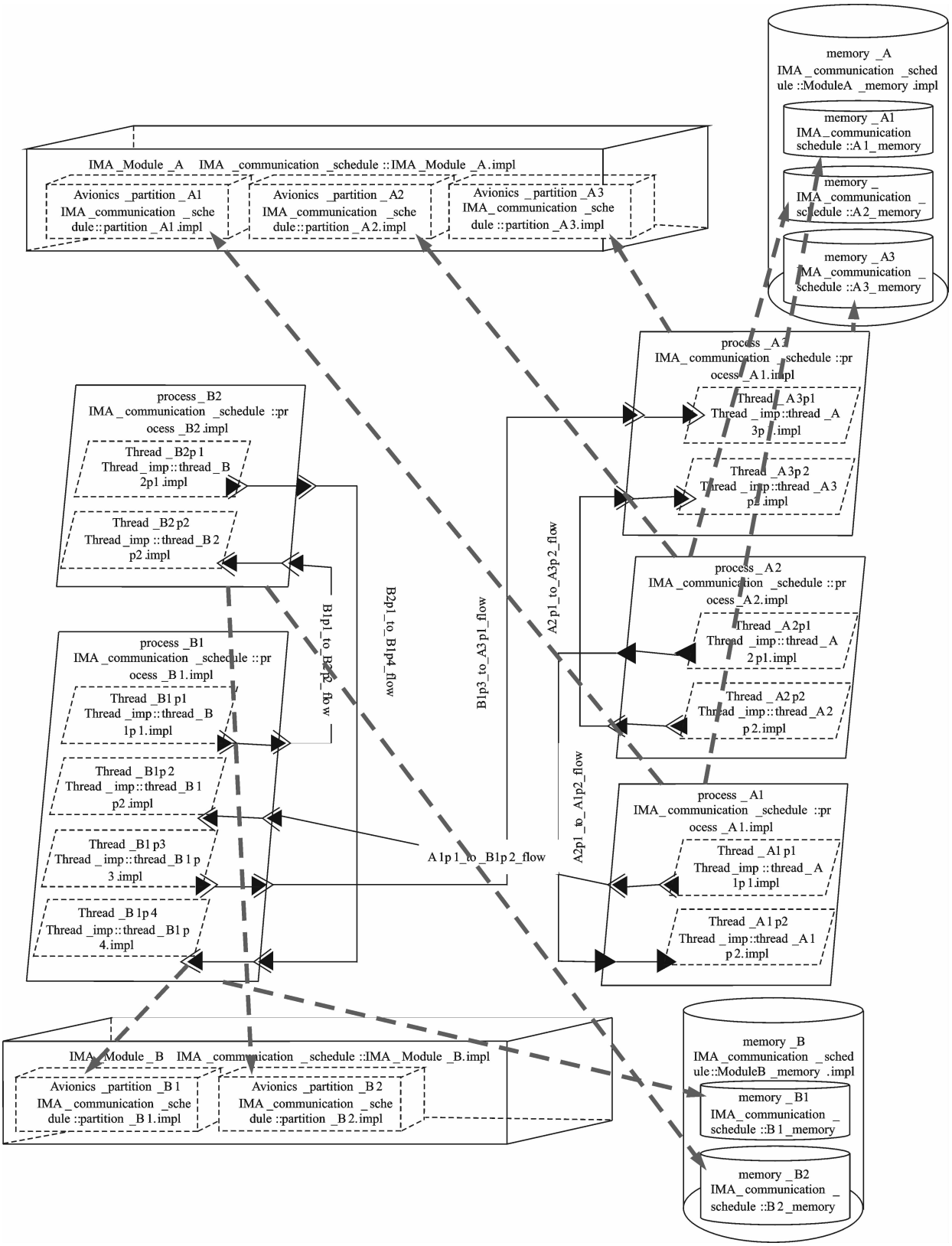


图 6　IMA 通信调度的 AADL 图形化模型

Fig. 6　AADL graphical model of IMA communication schedule

和目标端口的端口属性是否一致。用于比较的端口属性主要有以下几个：

（1）Source_Data_Size：该属性表示在端口之间进行通信的数据大小，主要与数据类型或是端口本身有关联；

（2）Input_Rate 和 Output_Rate：该属性表示

端口期望的输入/输出速率；

(3) Data_Model::Base_Type：该属性主要是针对“data classifier”，必须指定为相同的“classifier”。

由OSATE提供的“Analysis”插件功能，选择“Architecture”选项中的“Port Connection Consistency”选项，得到的模型端口连接一致性验证报告如表6所示。由于端口连接一致性报告是由Excel表格形式生成的，且内容较多，作者将其划分为两个表格，内容同属一份报告。

**表6 模型端口连接一致性验证报告**

**Tab.6 Check port connection consistency report**

| connection | source | destination |
|---|---|---|
| (process_A1.thread_A1p1.A1p1→process_B1.thread_B1p2.B1p2) | thread_A1p1.A1p1 | thread_B1p2.B1p2 |
| (process_A2.thread_A2p1.A2p1→process_A1.thread_A1p2.A1p2) | thread_A2p1.A2p1 | thread_A1p2.A1p2 |
| (process_A2.thread_A2p2.A2p2→process_A3.thread_A3p2.A3p2) | thread_A2p2.A2p2 | thread_A3p2.A3p2 |
| (process_B1.thread_B1p1.B1p1→process_B2.thread_B2p2.B2p2) | thread_B1p1.B1p1 | thread_B2p2.B2p2 |
| (process_B1.thread_B1p3.B1p3→process_A3.thread_A3p1.A3p1) | thread_B1p3.B1p3 | thread_A3p1.A3p1 |
| (process_B2.thread_B2p1.B2p1→process_B1.thread_B1p4.B1p4) | thread_B2p1.B2p1 | thread_B1p4.B1p4 |

| source Data_Size | destination Data_Size | source Output_Rate | destination Input_Rate | source Base_Type | destination Base_Type |
|---|---|---|---|---|---|
| 20 | 20 | 1.0..1.0 PerDispatch | 1.0..1.0 PerDispatch | integer | integer |
| 30 | 30 | 1.0..1.0 PerDispatch | 1.0..1.0 PerDispatch | integer | integer |
| 20 | 20 | 1.0..1.0 PerDispatch | 1.0..1.0 PerDispatch | integer | integer |
| 20 | 20 | 1.0..1.0 PerDispatch | 1.0..1.0 PerDispatch | integer | integer |
| 20 | 20 | 1.0..1.1 PerDispatch | 1.0..1.1 PerDispatch | integer | integer |
| 20 | 20 | 1.0..1.2 PerDispatch | 1.0..1.2 PerDispatch | integer | integer |

报告表格中不论是“process”或是“thread”组件中的端口作为源端口或是目标端口，其端口的属性Source_Data_Size，Input_Rate/Output_Rate，Data_Model::Base_Type都是一致的，因此，可以得到模型端口连接一致性的结论。在进程中，表示API的线程与其他进程内的线程之间的互连，这些进程可以属于同一模块，也可以属于不同模块。当进程属于同一模块时，表示通信连接为模块内分区间通信；当进程属于不同模块时，表示通信连接为模块间通信。每条连接表示端口之间的通信，如process_A1.thread_A1p1.A1p1→process_B1.thread_B1p2.B1p2表示了从端口A1p1到B1p2的通信连接，且端口A1p1是属于进程A1内A1p1线程的。表6中显示的每条通信连接与上部分分析的通信流吻合。

## 4 结束语

针对IMA系统的通信调度问题，首先分析了IMA系统的底层调度——通信调度，给出了通信调度的定义及特性等。然后，基于通信调度分析，利用AADL建模工具，将通信流的特性与AADL构件和属性相互映射，得到基于AADL的通信流表达形式。最后，将建好的模型进行实例化，由AADL自带的端口连接一致性检查功能对所建模型进行一致性检查。结果表明了在IMA系统通信调度中对通信流的理论描述与建模的通信流的一致性。

随着对有着ARINC653规范的IMA系统通信流调度分析的深入，下一步可以开展两方面工作：(1)通过引入第三方工具，对建模后的IMA通信调度模型进行可调度性测试，验证其配置信息的正确性；(2)将通信调度相关的配置信息用XML文件表示出来，通过将XML文件导入到AADL工具，自动生成AADL建模文件。

## 参考文献

[1] Tsiolakis A. Model-based test data generation for testing integrated modular avionics [C]//Dagstuhl

Seminar Perspectives of Model-Based Testing. Schloss Dagstuhl:[s. n.], 2004.

[2] Ott A. System testing in the avionics domain[D]. Bremen, Germany: University Bremen, 2007.

[3] Efkenmann C, Peleska J. Model-based testing for the second generation of integrated modular avionics [C]//Software Testing, Verification and Validation Workshops (ICSTW), 2011 IEEE Fourth International Conference on. S.l: IEEE Xplore, 2011.

[4] 何锋，宋丽茹，熊华钢. 航空电子双层任务分区调度设计[J]. 北京航空航天大学学报，2008，34(11)：1364-1368.
He Feng, Song Liru, Xiong Huagang. Two-level task partition scheduling design in integrated modular avionics [J]. Journal of Beijing University of Aeronautics and Astronautics, 2008, 34(11): 1364-1368.

[5] 李昕颖，熊华钢. 综合化航空电子分区隔离的建模与设计方法[J]. 北京航空航天大学学报，2011，37(1)：31-35.
Li Xinying, Xiong Huagang. Partition modeling and design in integrated avionics [J]. Journal of Beijing University of Aeronautics and Astronautics, 2011, 37(1): 31-35.

[6] 何锋，顾健，熊华钢. 基于动态优先级的核心处理安全分区优化设计[J]. 北京航空航天大学学报，2011，37(10)：1282-1287.
He Feng, Gu Jian, Xiong Huagang. Optimal safety partition design for integrated core processor under dynamical priority[J]. Journal of Beijing University of Aeronautics and Astronautics, 2011, 37(10): 1282-1287.

[7] 乔乃强，徐涛，谷青范. ARINC653 分区调度算法的研究与改进[J]. 计算机工程，2011，37(20)：249-251.
Qiao Naiqiang, Xu Tao, Gu Qingfan. Research and improvement of ARINC653 partition schedule algorithm [J]. Computer Engineering, 2011, 37(20): 249-251.

[8] 李昕颖，顾健，何锋，等. 硬实时系统在强分区约束下的双层分区调度[J]. 计算机学报，2010，33(6)：1032-1039.
Li Xinying, Gu Jian, He Feng, et al. Two-level partition scheduling in hard real time system under strong partition constraints[J]. Chinese Journal of Computers, 2010, 33(6): 1032-1039.

[9] 张永悦，云利军，孙瑜. 基于分区的航电系统调度分析工具实现[J]. 计算机工程，2014，40(4)：42-47.
Zhang Yongyue, Yun Lijun, Sun Yu. Implementation of scheduling analysis tool for avionics system based on partition[J]. Computer Engineering, 2014, 40(4): 42-47.

[10] 孟阳. 基于 ARINC653 标准的 AADL 模型配置工具的研究与实现[D]. 成都：电子科技大学，2012.
Meng Yang. Research and implementation of AADL model configuration tool based on ARINC653 standard [D]. Chengdu: University of Electronic and Technology of China, 2012.

[11] 胡军，马金晶，袁翔，等. 一种基于 AADL 的 IMA 系统配置信息的正确性检测方法[J]. 南京航空航天大学学报，2014，46(6)：920-930.
Hu Jun, Ma Jinjing, Yuan Xiang, et al. Correctness verification for integrated modular avionics system configuration based on AADL model [J]. Journal of Nanjing University of Aeronautics & Astronautics, 2014, 46(6): 920-930.

[12] 袁翔. 模型驱动的综合航电系统配置信息的分析与验证方法研究[D]. 南京：南京航空航天大学，2014.
Yuan Xiang. Research on model-driven analysis and verification of system configuration for integrated modular avionics [D]. Nanjing: Nanjing University of Aeronautics and Astronautics, 2014.

[13] SAE Aerospace. AS5506/2:SAE Architecture Analysis and Design Language(AADL) Annex Volume2 [EB/OL]. http://www.sae.org/technical/standards/AS5506/2, 2011-01-17.