OPIC
Office de la propriété intellectuelle du Canada

CIPO
Canadian Intellectual Property Office

(12) (19) (CA) **Brevet-Patent**

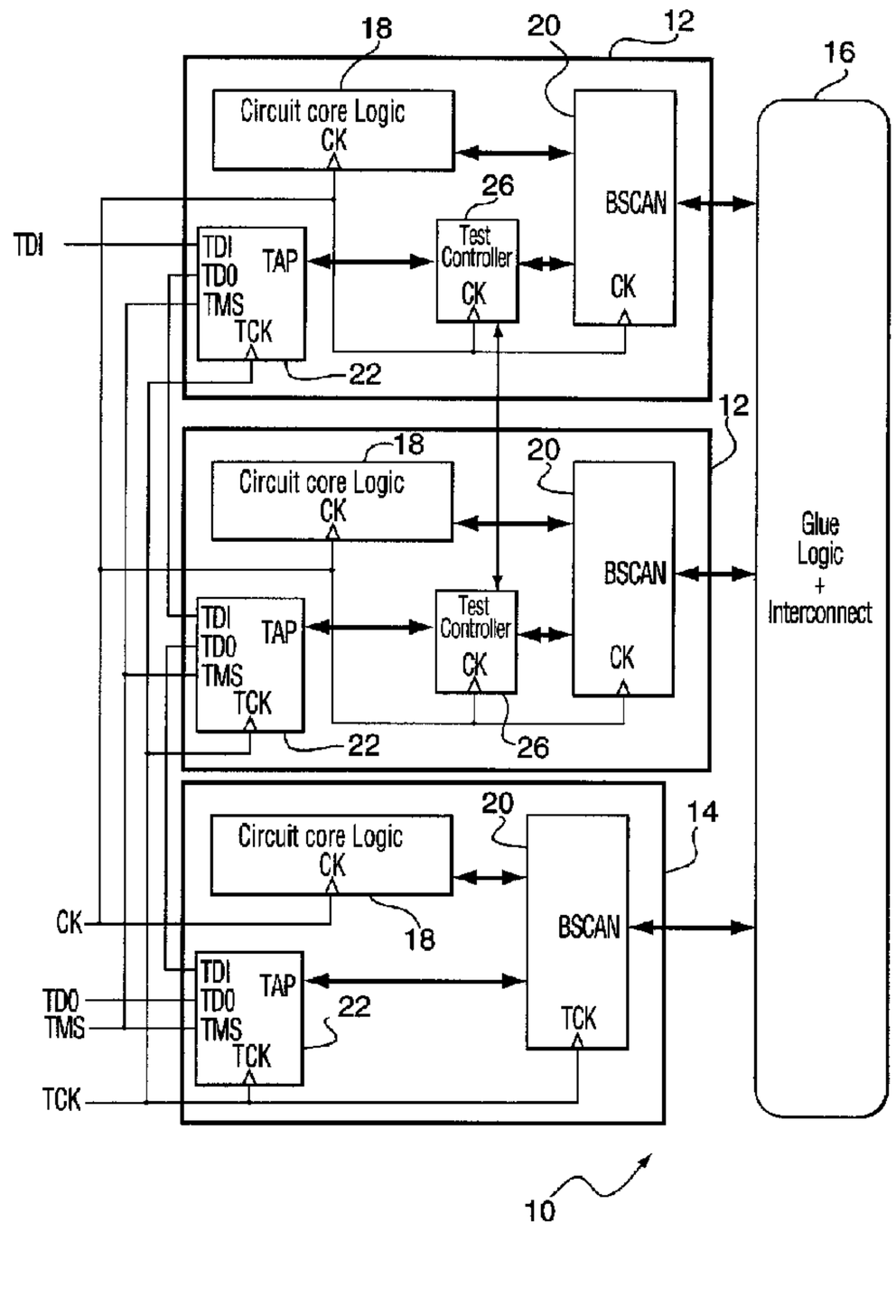
(11) (21) (C) **2,249,088**
(22) 1998/09/29
(43) 1999/04/10
(45) 2000/09/26

(72) NADEAU-DOSTIE, Benoit, CA
(72) COTE, JEAN-FRANCOIS, CA
(73) LOGICVISION, INC., CA

(51) Int.Cl.[6] G01R 31/28, G01R 31/3181, H01L 27/00
(30) 1997/10/10 (08/948,842) US

(54) **METHODE ET APPAREIL D'ESSAI D'INTERCONNEXION A GRANDE VITESSE**

(54) **METHOD AND APPARATUS FOR HIGH-SPEED INTERCONNECT TESTING**



(57) Une méthode d'essai de l'interconnectivité à grande vitesse des plaquettes comprenant des composants fonctionnant à une fréquence d'horloge système élevée à l'aide d'une méthode d'essai conforme à la norme IEEE 1149.1, grâce à laquelle les données d'essai sont transférées à vers des composants et en provenance de ceux-ci à la fréquence d'une horloge d'essai durant les

(57) A method of testing high speed interconnectivity of circuit boards having components operable at a high speed system clock, employing an IEEE 1149.1 standard test method in which test data is shifted into and from the components at the rate of a test clock during Shift_In and Shift_Out operations, and having an Update operation and a Capture operation between the Shift_In and

OPIC
Office de la propriété intellectuelle du Canada

CIPO
Canadian Intellectual Property Office

(11) (21) (C) **2,249,088**
(22) 1998/09/29
(43) 1999/04/10
(45) 2000/09/26

décalages d'entrée et de sortie, et durant une mise à jour et une saisie entre les décalages d'entrée et de sortie, les composants comprenant un premier groupe de composants pouvant effectuer la mise à jour et la saisie à la fréquence de l'horloge d'essai seulement, et un deuxième groupe de composants pouvant effectuer la mise à jour et la saisie à la fréquence de l'horloge système, la méthode comprenant les étapes d'exécution de décalage d'entrée dans tous les composants concurremment à la fréquence de l'horloge d'essai; exécution de la mise à jour et de la saisie dans le premier groupe de composants à la fréquence de l'horloge d'essai; et exécution de la mise à jour et de la saisie dans le deuxième groupe de composants à la fréquence de l'horloge système. Pour cette méthode, on emploie un circuit intégré de pointe, un contrôleur d'essai et des cellules d'essais périphériques.

Shift_Out operations, the components including a first group of components capable of performing the Update and Capture operations at the rate of the Test Clock only and a second group of components capable of performing the Update and Capture operations at the rate of the system clock, the method comprising the steps of performing the Shift_In operation in all of the components concurrently at the rate of the Test Clock; performing the Update and Capture Operations in the first group of components at the rate of the Test Clock; and performing the Update and Capture operations in the second group of components at the rate of the system Clock. The method employs a novel integrated circuit, test controller and boundary scan cells.

POINT-OF-SALE SYSTEM INCLUDING ISOLATION LAYER BETWEEN CLIENT AND SERVER SOFTWARE

Abstract of the Disclosure

A system and method of transferring information between a first software application and a second software application which employ an isolation layer. The system includes a client computer system provided by a first seller of computer systems, including a client software application, and a server computer system provided by a second seller of computer systems different from the first seller of computer systems, including a server software application which provides information from the server computer system to the client computer system. The system additionally includes isolation layer software, either at the client computer system or at the server computer system, which facilitates communication between the client software application and the server software application to transfer the information.

POINT-OF-SALE SYSTEM INCLUDING ISOLATION LAYER BETWEEN CLIENT AND SERVER SOFTWARE

Cross-reference to related applications

The present application is related to co-pending application serial number 08/539,449, filed 10/05/95, entitled "SYSTEM AND METHOD OF OBTAINING INFORMATION FROM A PRICE LOOK-UP FILE", and having as inventor, John Goodwin.

Background of the Invention

The present invention relates to point-of-sale (POS) systems, and more specifically to a POS system including an isolation layer between POS server software and a client POS application program.

POS systems typically include a central server and a plurality of checkout terminals connected through a client-server network.

The checkout terminals include bar code readers and keyboards for entry of the item numbers during a transaction.

The central server stores a price look-up (PLU) file which associates item numbers with item prices. The central server processes requests from the checkout terminals for price information for the items. The central server may perform additional tasks, such as storing transaction history (audit log

and transaction totals history), storing accountability totals, storing cashier and terminal accountability totals, holding future price change information (batches), performing credit authorization, performing check validation, and performing frequent shopper validation and specials.

POS checkout software has traditionally included client and server software that have been developed together and sold as a single proprietary product. However, this solution may not be an optimal one for a retailer. Retailers may find that combining server software from one developer with client software from another developer gives their POS systems the features that they want. This may be especially true for retailers that want the best solutions for both the server and client software in a market where no one developer of proprietary POS software exceeds at both.

Therefore, it would be desirable to provide a POS system including an isolation layer between the POS server software and the POS client application software to enable server and client software from different developers to be combined.

Summary of the Invention

In accordance with the teachings of the present invention, a POS system including an isolation layer between POS server software and a client POS application program is provided.

A system and method of transferring information between a first software application and a second software application which employ an isolation layer. The system includes a client computer system provided by a first seller of computer systems, including a client software application, and a server computer system provided by a second seller of computer systems different from the first seller of computer systems, including a server software application which provides information from the server computer system to the client computer system. The system additionally includes isolation layer software, either at the client computer system or at the server computer system, which facilitates communication between the client software application and the server software application to transfer the information.

The method includes the steps of providing the first software application by a first company, providing the second software application by a second company different from the first company, executing the first software application by a first computer, executing the second software application by a second computer different from the first computer, providing an isolation layer having hooks into the second software application, executing the isolation layer, and establishing communication between the first software application and the second software application by the isolation layer to transfer the information.

It is accordingly an object of the present invention to provide a POS system including an isolation layer between POS server software and a client POS application program.

It is another object of the present invention to allow server and client software from different developers to be combined using an isolation layer.

It is another object of the present invention to provide an isolation layer for server software that provides read and write services for client POS software in a computer network that includes client computers running different client POS software.

Brief Description of the Drawings

Additional benefits and advantages of the present invention will become apparent to those skilled in the art to which this invention relates from the subsequent description of the preferred embodiments and the appended claims, taken in conjunction with the accompanying drawings, in which:

Fig. 1 is a block diagram of a transaction management system;

Fig. 2 is a block diagram illustrating the client POS software, server POS software, and the isolation layer between them;

Fig. 3 is a block diagram of a system for producing the isolation layer;

Fig. 4 is a flow diagram illustrating how the isolation layer is created; and

Fig. 5 is a flow diagram illustrating operation of a system specific application and, in particular, a method of transferring information from the system specific application to the client POS software through the isolation layer.

Detailed Description of the Preferred Embodiment

Referring now to Fig. 1, transaction system 10 preferably includes host computer system 12 and point-of-service (POS) terminals 22A and 22B. Host computer system 12 and point-of-service (POS) terminals 22A and 22B are coupled together to form a network.

POS terminals 22A and 22B execute independent applications 23A and 23B, including client POS software 24A and 24B for completing transactions. POS software 24A and 24B may be identical (e.g. the same application from the same developer) or may be different transaction processing applications.

As illustrated, independent applications 23A were developed by a different developer than the developer of server POS software 16. Thus, independent applications 23A require isolation layer 14 to communicate with POS server software 16.

Independent applications 23B were developed by the same developer as the developer of POS server software 16. Thus, independent applications 23B do not require isolation layer 14 to communicate with server POS software 16. Advantageously, the present invention is capable of connecting a plurality of independent applications 23A running on a plurality of different client terminals from different developers to server POS software 16.

Client POS software 24A and 24B display transaction information on displays 27A, 27B. Client POS software 24A and 24B receive article identification information from bar code readers 26A, 26B and keyboards 28A, 28B. Client POS software 24A and 24B send the article identification information to host computer system 12. Host computer system 12 reads price look-up (PLU) file 20 to obtain price information and sends the price information to POS terminals 26A and 26B.

Other examples of independent applications 17 include price checking software for consumer price checking computers, weighing software for electronic scales, price checking software and item description label printing software for hand-held terminals, and EPL auditing software.

Host computer system 12 executes POS server software 16 and isolation layer 14. POS server software 16 processes requests from independent applications 23A and 23B. POS server software 16 includes system specific applications 17, such as

price look-up (PLU) file software 40, transaction totals software 42, cashier authentication software 44, and credit checking software 46. These are but examples of system specific software in use today. The present invention anticipates other types of system specific software 17 as well.

Isolation layer 14 provides translation services between independent applications 23A and server POS software 16. Thus, isolation layer 14 allows client POS software 24A from one developer to function with system specific applications 17 from a different developer.

Isolation layer 14 may reside in host computer system 12 or client terminal 22A. Typically, a provider of client terminals 22A also provides independent applications 23A, and a provider of host computer system 12 also provides server POS software 16.

If the provider of client terminals 22A is adding client terminals 22A to an existing network including a server from a different provider, the provider of client terminals 22A may wish to write and store isolation layer 14 in client terminals 22A.

On the other hand, if a provider of servers and server POS software 16 is adding host computer system 12 to an existing network including client terminals 22A from a different provider,

the provider of servers may wish to write and store isolation layer 14 in host computer system 12.

Isolation layer 14 preferably includes one library file for all system specific applications 17 or individual library files for system specific applications 17. Use of multiple libraries adds network design flexibility but penalizes operation by loading each library into memory, even if only part of isolation layer 14 is used. Isolation layer 14 may take other forms including a single application, a single driver, or multiple drivers that perform read operations into server POS software 16 in order to obtain and map information to be used by client POS software 24A. These drivers also perform writes into server POS software 16 map information created by client POS software 24A.

Storage medium 18 stores PLU file 20 and other information files and is preferably a fixed disk drive.

Turning now to Fig. 2, the software architecture within system 10 is described in more detail. Discussion references client POS software 24A, but other independent applications 23A operate in a similar fashion.

Client POS software 24A includes POS checkout application software 30 and interprocess communications (IPC) software 32. POS checkout software 30 records items by scanning them, prints them on a receipt, and adds their prices to produce

a total transaction amount. POS checkout software 30 also performs specific transaction-related functions, such as processing food stamps and other forms of payment under government entitlement programs, calculating service charges, performing price change functions, performing out of transaction functions (loans, pickups), etc. An example of POS checkout software 30 is the UNITY® checkout application developed and sold by the Assignee of the present invention.

IPC software 32 is software that is added to POS checkout software 30 to facilitate interprocess communications (IPC) between POS checkout software 30 and isolation layer 14. IPC software 32 will vary with operating system, from threads, queues, named pipes, shared files, sockets, etc.

Isolation layer 14 includes read/write engine and translation layer 34 and read/write layer 36. Read/write engine and translation layer 34 translates one request into one or more read/ write requests as needed for host computer system 12 based on standard server inputs and custom outputs. Read/write layer 36 performs the actual reads and writes to host computer system 12 based on the requests issued in read/write engine and translation layer 34.

An example of read/write layer 36 is the UNITY® file service layer.

Server POS software 16 may include both the UNITY® file service layer and the UNIX® file system.

In a traditional POS system, client POS software communicates directly with POS server software through a communication methodology, such as named pipes. Under Applicant's invention, client POS software 24A is modified to include IPC software 32, and an isolation layer 14 is added. Communication between client POS software 24A and POS server software 16 occurs between IPC software and isolation layer 14.

Turning now to Fig. 3, the components that are used to construct isolation layer 14 include templates 60, object code 62, example mappings and translations 64, target routines 66, and make files 68 for each of system-specific applications 17. Templates 60 are a super set of example mappings and translations 64 and target routines 66. Isolation layer 14 includes templates 60 for each of system specific applications 17. Example mappings and translations 64 and target routines 66 are edited and customized by developers to produce customized mappings and translations 65 and customized target routines 67 (customized templates 61). Customized mappings and translations 65 and customized target routines 67 along with object code 62 are fed into make files 68 in order to produce executable code for isolation layer 14.

Object code 62 consists of libraries 41 for each of system specific applications 17. As mentioned above, isolation layer 14 may include one library for establishing communication for all system specific applications 17 or a plurality of separate libraries.

Example mappings and translations 64 provide non-displayable information (in code and/or in documentation) and may be edited to customize isolation layer 14.

Target routines 66 are system specific application routines that may be edited and customized to suit the needs of the target transaction establishment.

Make files 68 are files that pass source code (templates 60) through compiler 72 and then pass object code 62 through linker 74 to produce the target executable.

Turning now to Fig. 4, the method of creating isolation layer 14 begins with START 80.

In step 82, independent applications 23A are isolated from system specific applications 17 through a non-displayable mapping determination and translation process. Example mappings and translations 64 result.

In steps 84, a template 60 for one of system specific applications 17 is provided. Template 60 is packaged in an installable format that can then be distributed and installed on the target system.

In step 86, customized mappings and translations 65 and customized target routines 67 are determined.

In steps 88-92, customized template 61 is produced by changing template 60 to reflect customized mappings and translations 65.

In step 88, functions to be implemented are edited. For example, for PLU file reader software 40, the "read first", "read next", and "read specific" sections are edited. The "read first" section is a routine that must be completed/customized to read the first PLU in PLU file 20. The "read next" section is a routine that must be completed/customized to read the next PLU in PLU file 20 sequentially. The "read specific" section is a routine that must be completed/customized to read a specific PLU file record.

In step 90, customized mapping and translations 65 and customized target routines 67 derived from step 86 are applied to template 60.

In step 92, operation returns to step 84 if library functionality for an additional specific application 17 must be added. Otherwise, operation continues to step 94.

In step 94, a new executable (isolation layer 14) is compiled and linked from object code 62 and templates 61 for each of system specific applications 17 using make files 68, compiler 72, and linker 74.

In step 96, the method ends.

Referring now to Fig. 5, operation of a system specific application 17 is illustrated in detail, beginning with START 100. In particular, operation illustrates a method of transferring information between a terminal 22A and host computer system 12 through isolation layer 14.

In step 102, a system specific application 17 starts up and configures itself.

In step 104, system specific application 17 waits for an interprocess communications message from isolation layer 14. Interprocess communication services (e.g., "queues" for UNIX, threads or pipes for OS/2) are provided by the operating system executed by host computer system 12.

Isolation layer 14 calls on the operating system to send a request from client POS software 24A to system specific application 17. Client POS software 24A issues a call to isolation layer 14, instructing isolation layer 14 to perform a routine provided by isolation layer 14 and related to the purpose of system specific application 17.

For example, when system specific application 17 is PLU file reader software 40, the routine would be a "read direct" routine within isolation layer 14. Isolation layer 14 calls on the operating system to send the interprocess communications message to system specific application 17.

If such a message is received, system specific application 17 determines whether the interprocess communications message contains an "exit" command in step 105. If it does, system specific application 17 terminates in step 116.

If the interprocess communications message does not contain an "exit" command, system specific application 17 obtains the information from its source in step 106. For example, if system specific application 17 is PLU file reader software 40, system specific application 17 uses operating system 52 to obtain the information from PLU file 20.

In step 114, system specific application 17 calls on the operating system to return the information via interprocess communications to the client POS software 24A via isolation layer 14. System specific application 17 returns to a waiting state in step 104.

After system specific application 17 sends the information to isolation layer 14, isolation layer 14 passes the information to client POS software 24A. Client POS software application 24A can then display the information, compare the information, or otherwise examine the information in accordance with the functions of client POS software 24A.

Although the present invention has been described with particular reference to certain preferred embodiments thereof,

variations and modifications of the present invention can be effected within the spirit and scope of the following claims.

What is claimed is:

1. A transaction system comprising:

a client computer system provided by a first seller of computer systems, including a client software application;

a server computer system provided by a second seller of computer systems different from the first seller of computer systems, including a server software application which provides information from the server computer system to the client computer system and isolation layer software which facilitates communication between the client software application and the server software application to transfer the information.

2. A transaction system comprising:

a server computer system provided by a first seller of computer systems, including a server software application;

a client computer system provided by a second seller of computer systems different from the first seller of computer systems, including a client software application which obtains information from the server computer system and isolation layer software which facilitates communication between the client software application and the server software application to transfer the information.

3. A method of transferring information between a first software application and a second software application comprising the steps of:

(a) providing the first software application by a first company;

(b) providing the second software application by a second company different from the first company;

(c) executing the first software application by a first computer;

(d) executing the second software application by a second computer different from the first computer;

(e) providing an isolation layer having hooks into the second software application;

(f) executing the isolation layer; and

(g) establishing communication between the first software application and the second software application by the isolation layer to transfer the information.

4. The method as recited in claim 3, wherein step f comprises the substep of:

(f) executing the isolation layer by the first computer.

5. The method as recited in claim 3, wherein step f comprises the substep of:

(f) executing the isolation layer by the second computer.

6. A method of transferring information between a client software application and a server software application comprising the steps of:

(a) installing a client computer by a first company;

(b) providing and installing the client software application by the first company;

(c) installing a server computer by a second company different from the first company after installation of the client computer and the client software by the first company;

(d) providing and installing the server software application by the second company;

(e) providing and installing an isolation layer having hooks into the server software application by the second company;

(f) executing the client software application by the client computer;

(g) executing the server software application and the isolation layer by the server computer; and

(h) establishing communication between the client software application and the server software application by the isolation layer to transfer the information.

7. A method of transferring information between a client software application and a server software application comprising the steps of:

(a) installing a server computer by a first company;

(b) providing and installing the server software application by the first company;

(c) installing a client computer by a second company different from the first company after installation of the server computer and the server software by the first company;

(d) providing and installing the client software application by the second company;

(e) providing and installing an isolation layer having hooks into the server software application by the second company;

(f) executing the client software application and the isolation layer by the client computer;

(g) executing the server software application by the server computer; and

(h) establishing communication between the client software application and the server software application by the isolation layer to transfer the information.

8. A method of obtaining information for a client software application from a server software application comprising the steps of:

(a) providing the client software application from a first company;

(b) providing a server software application for accessing the information by a second company different from the first company;

(c) providing an isolation layer between the client software application and the server software application;

(d) sending a call for the information to the isolation layer by the client software application;

(e) sending a request message for the information to the server software application by the isolation layer;

(f) instructing control software to obtain the information by the server software application;

(g) sending the information to the isolation layer by the server software application; and

(h) sending the information to the client software application by the isolation layer.

FIG. 1

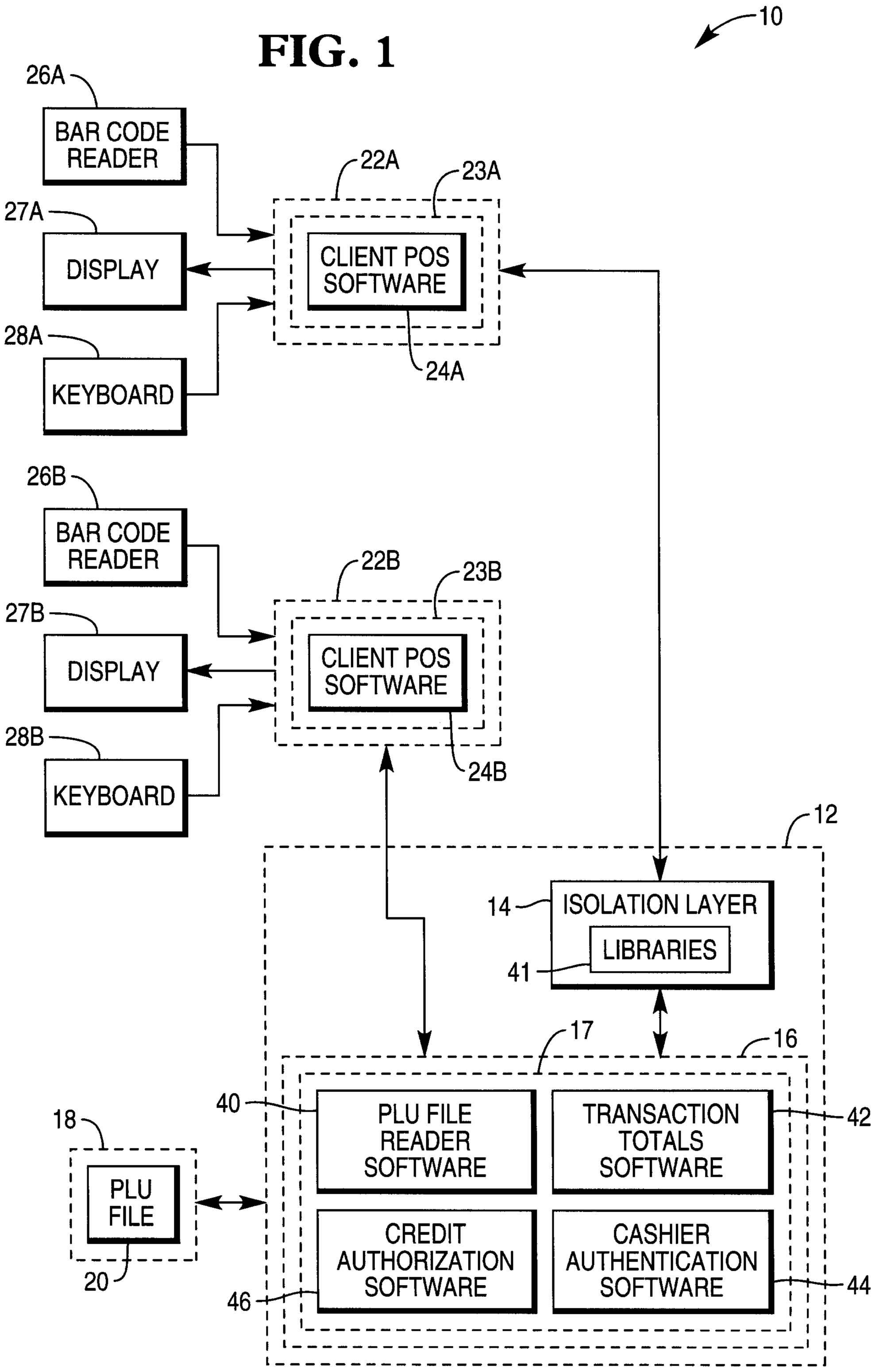

10
26A
BAR CODE READER
27A
DISPLAY
28A
KEYBOARD
22A
23A
CLIENT POS SOFTWARE
24A
26B
BAR CODE READER
27B
DISPLAY
28B
KEYBOARD
22B
23B
CLIENT POS SOFTWARE
24B
12
14
ISOLATION LAYER
LIBRARIES
41
17
16
40
PLU FILE READER SOFTWARE
TRANSACTION TOTALS SOFTWARE
42
18
PLU FILE
20
CREDIT AUTHORIZATION SOFTWARE
46
CASHIER AUTHENTICATION SOFTWARE
44

FIG. 2

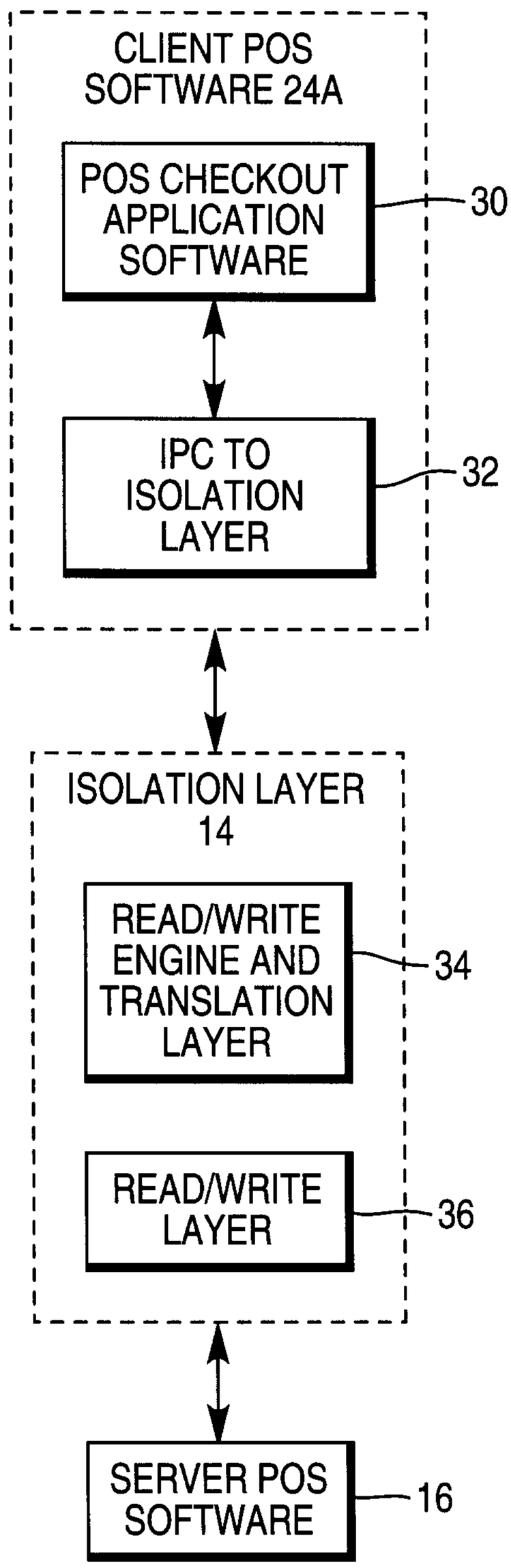

CLIENT POS
SOFTWARE 24A
POS CHECKOUT
APPLICATION
SOFTWARE
30
IPC TO
ISOLATION
LAYER
32
ISOLATION LAYER
14
READ/WRITE
ENGINE AND
TRANSLATION
LAYER
34
READ/WRITE
LAYER
36
SERVER POS
SOFTWARE
16

FIG. 3

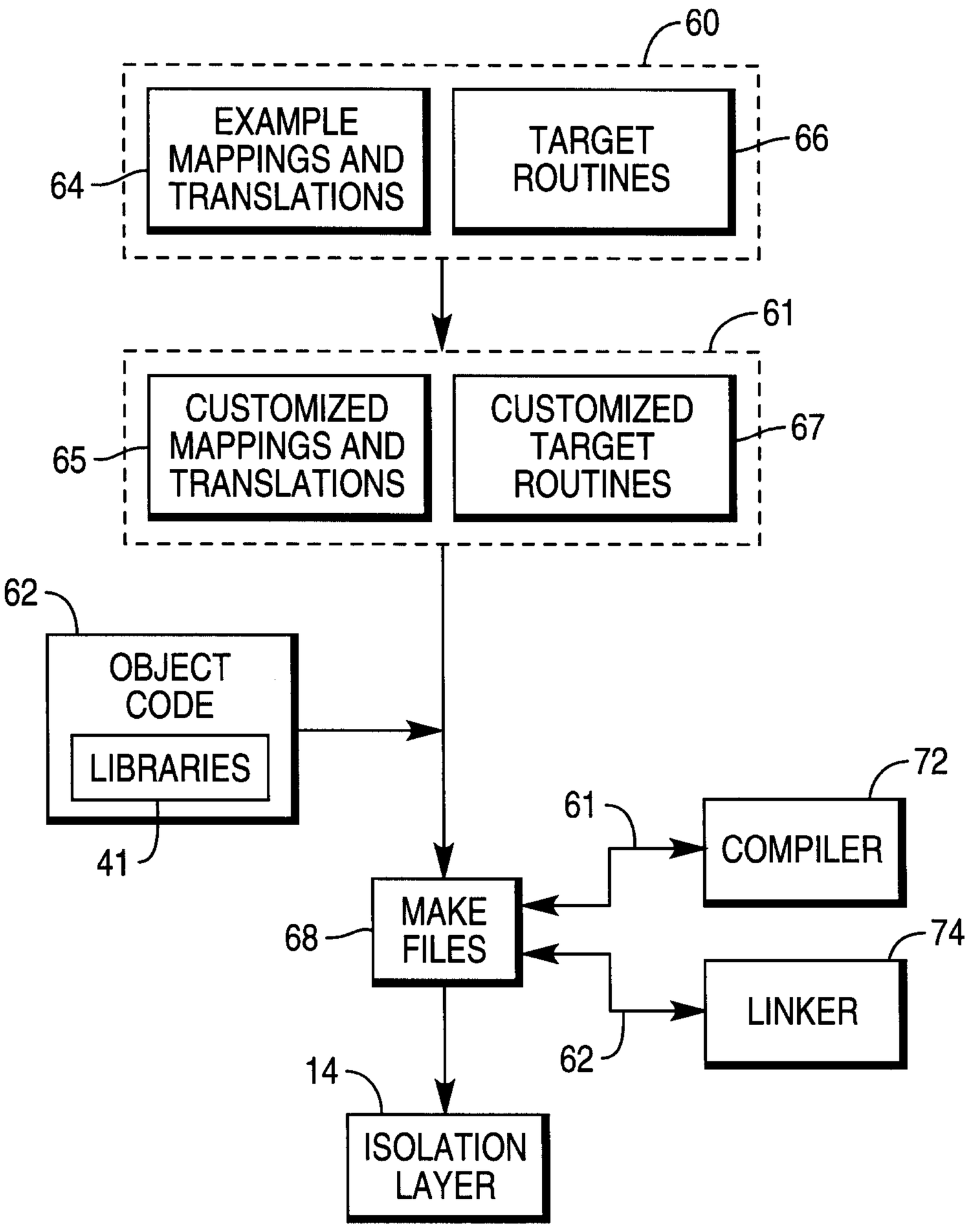

60
EXAMPLE MAPPINGS AND TRANSLATIONS
64
TARGET ROUTINES
66
61
CUSTOMIZED MAPPINGS AND TRANSLATIONS
65
CUSTOMIZED TARGET ROUTINES
67
62
OBJECT CODE
LIBRARIES
41
72
61
COMPILER
MAKE FILES
68
74
LINKER
62
14
ISOLATION LAYER

FIG. 4

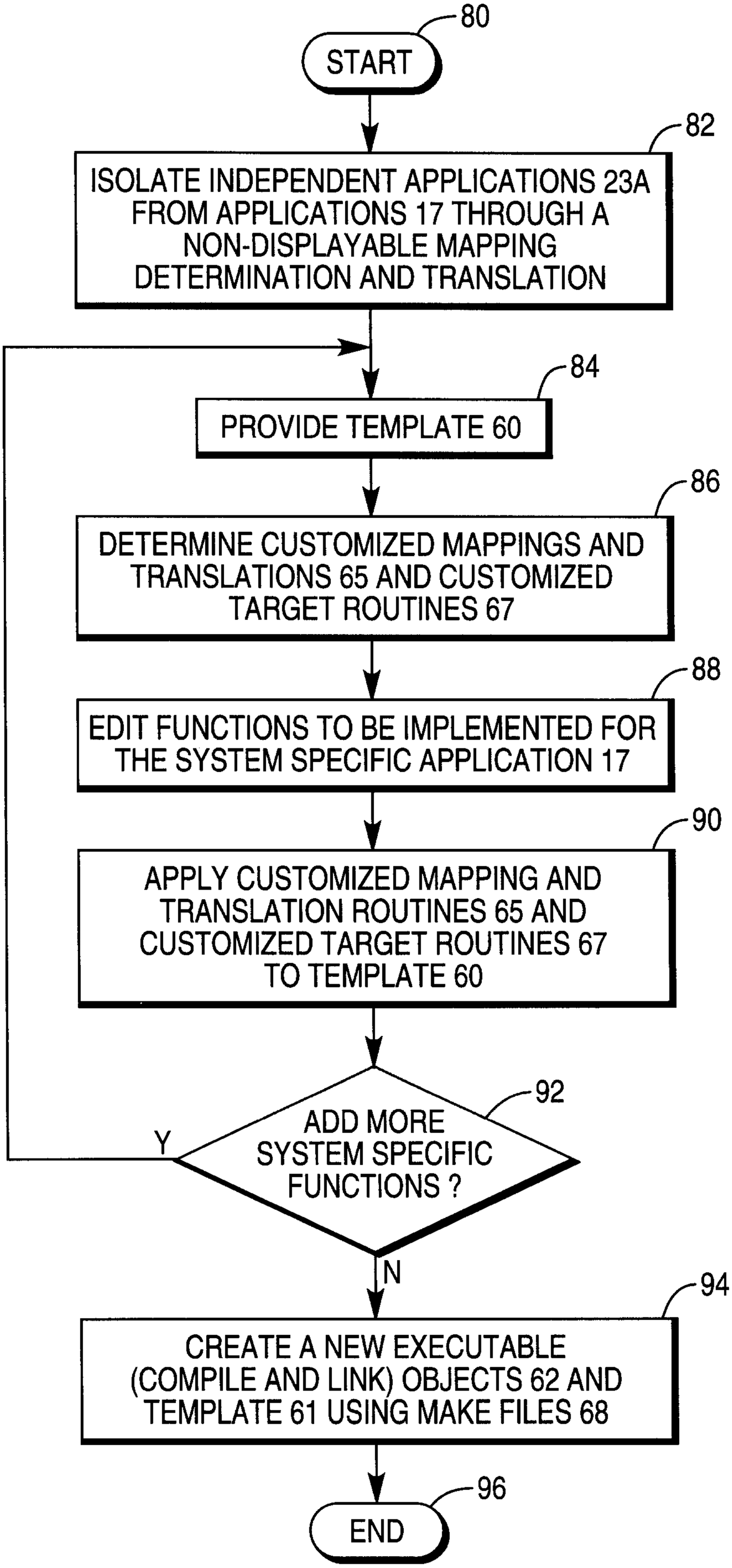
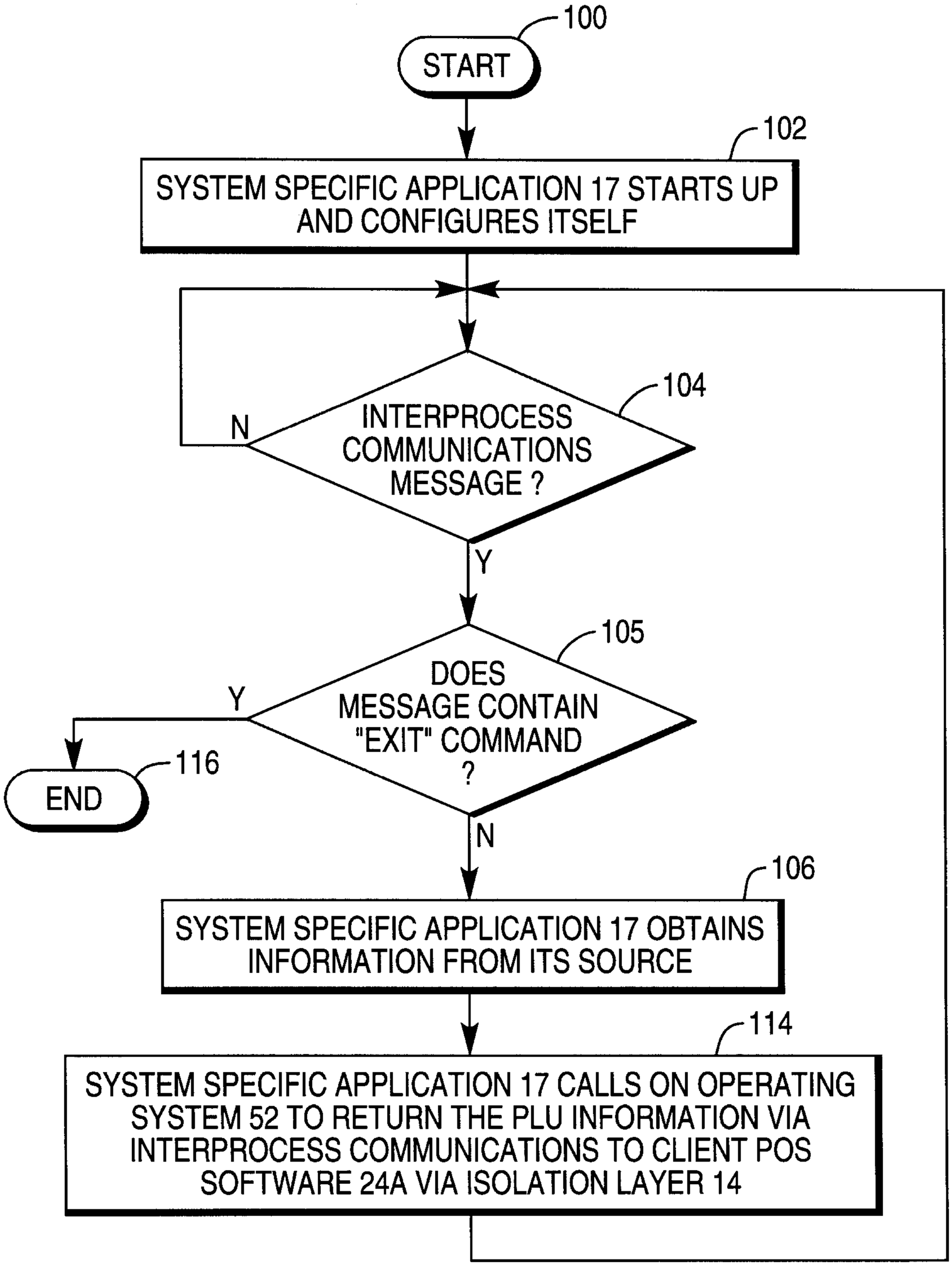
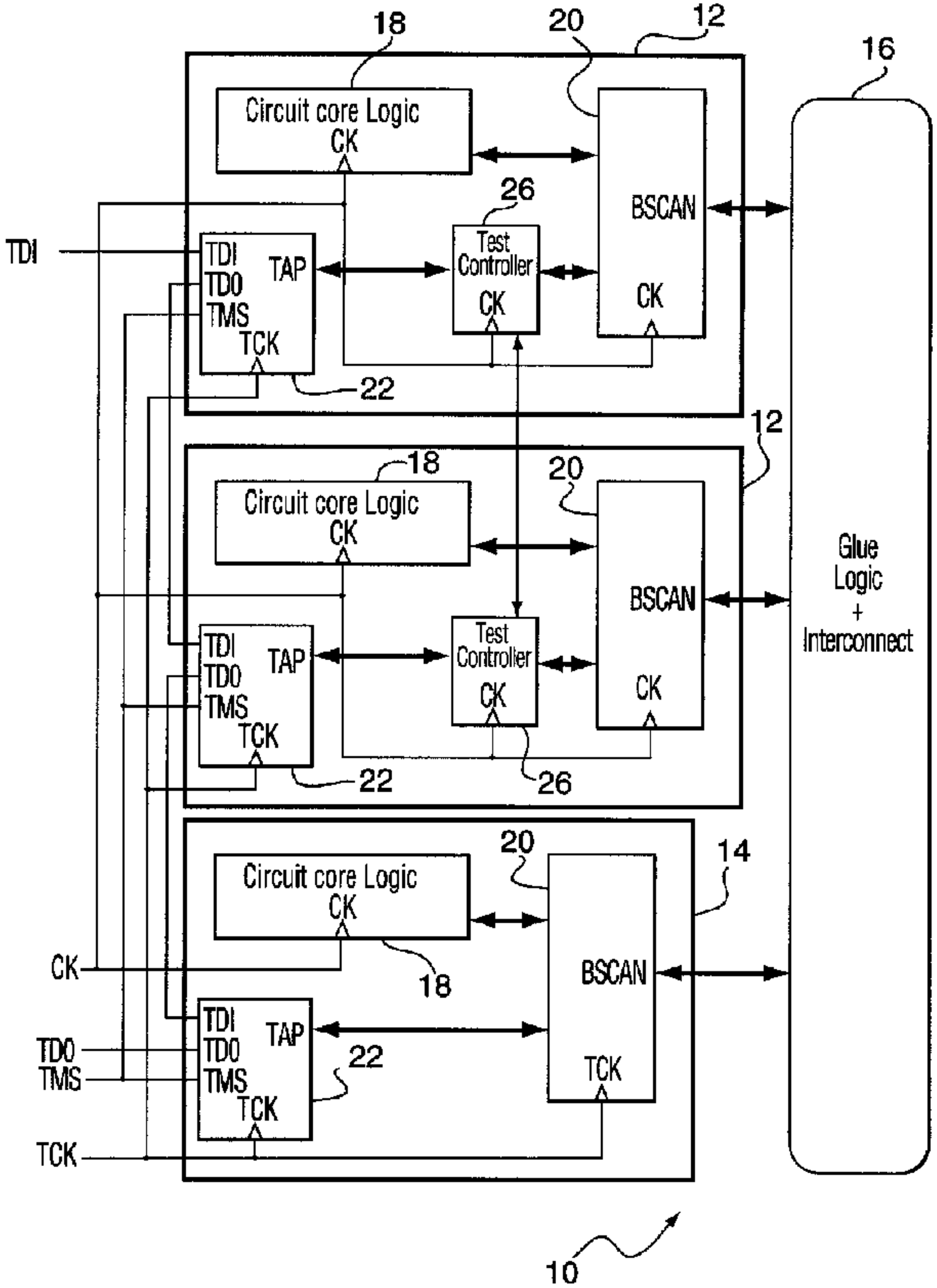

80
START
82
ISOLATE INDEPENDENT APPLICATIONS 23A FROM APPLICATIONS 17 THROUGH A NON-DISPLAYABLE MAPPING DETERMINATION AND TRANSLATION
84
PROVIDE TEMPLATE 60
86
DETERMINE CUSTOMIZED MAPPINGS AND TRANSLATIONS 65 AND CUSTOMIZED TARGET ROUTINES 67
88
EDIT FUNCTIONS TO BE IMPLEMENTED FOR THE SYSTEM SPECIFIC APPLICATION 17
90
APPLY CUSTOMIZED MAPPING AND TRANSLATION ROUTINES 65 AND CUSTOMIZED TARGET ROUTINES 67 TO TEMPLATE 60
92
ADD MORE SYSTEM SPECIFIC FUNCTIONS ?
Y
N
94
CREATE A NEW EXECUTABLE (COMPILE AND LINK) OBJECTS 62 AND TEMPLATE 61 USING MAKE FILES 68
96
END

FIG. 5


100
START
102
SYSTEM SPECIFIC APPLICATION 17 STARTS UP AND CONFIGURES ITSELF
104
INTERPROCESS COMMUNICATIONS MESSAGE ?
N
Y
105
DOES MESSAGE CONTAIN "EXIT" COMMAND ?
Y
116
END
N
106
SYSTEM SPECIFIC APPLICATION 17 OBTAINS INFORMATION FROM ITS SOURCE
114
SYSTEM SPECIFIC APPLICATION 17 CALLS ON OPERATING SYSTEM 52 TO RETURN THE PLU INFORMATION VIA INTERPROCESS COMMUNICATIONS TO CLIENT POS SOFTWARE 24A VIA ISOLATION LAYER 14


18
20
12
16
Circuit core Logic
CK
26
BSCAN
TDI
TDI
TDO
TMS
TCK
TAP
Test
Controller
CK
CK
22
12
18
20
Circuit core Logic
CK
Glue
Logic
+
Interconnect
BSCAN
TDI
TDO
TMS
TCK
TAP
Test
Controller
CK
CK
22
26
20
14
Circuit core Logic
CK
CK
18
BSCAN
TDI
TDO
TMS
TCK
TAP
22
TDO
TMS
TCK
TCK
10