409228

申請日期	87.10.21
案號	87118137
類別	G06F9/45

（以上各欄由本局填註）

A4
C4

發明 ~~新型~~ 專利說明書

409228

一、發明 ~~新型~~ 名稱	中文	編譯時間資料相依性驗證
	英文	"COMPILE-TIME DATA DEPENDENCY VERIFICATION"
二、發明 ~~創作~~ 人	姓名	葛瑞漢 威廉 艾華特
	國籍	加拿大
	住、居所	加拿大安大略省唐米爾市塔瓦利道1號
三、申請人	姓名（名稱）	美商萬國商業機器公司
	國籍	美國
	住、居所（事務所）	美國紐約州阿蒙市新果園路
	代表人姓名	費羅普

409228

（由本局填寫）

承辦人代碼：	A6 B6
大　　類：	
ＩＰＣ分類：	

本案已向：

國（地區）	申請專利，申請日期：	案號：	，□有　□無主張優先權
加拿大	1998年6月12日	2,240,584	□有 ☑無主張優先權

有關微生物已寄存於：　　，寄存日期：　　，寄存號碼：

（請先閱讀背面之注意事項再填寫本頁各欄）

裝　訂　線

五、發明說明（1 ）

發明領域

本發明係有關編譯器及相關軟體開發工具程式之改良，尤係有關在編譯時電腦程式中資料相依性之驗證。

發明背景

經常在電腦程式中用到資料相依性。因爲資料相依性可提供記憶體效率及執行時執行效率之優點，所以在電腦程式設計中使用資料相依性。

在產生資料相依性時，可撰寫將程式碼，該程式碼將利用與程式語言或資料類型定義中並未明確界定的資料結構相關之特徵或規則。此種資料相依性的例子包括下列要求：兩種不同的資料結構具有相同的長度，或兩個資料單元係位於記憶體中的相鄰處。程式設計師可利用各資料結構間之已知關係，而產生可以更快的速度來執行或所需的記憶體少於他種情形之程式。

良好的程式設計規範要求：在伴隨電腦程式碼的文件說明中示出與如何操作資料相關的假設或規則（資料相依性）。然而，若要無誤地使用受資料相依性支配的資料，將有賴於程式設計師的了解，而非有賴於編譯器所加諸的任何限制。當並未提及或無法取得文件說明時，或所準備的文件說明敘述不清或編排不當時，若要修改電腦程式，則可能會發生問題。

此外，於撰寫複雜的電腦程式碼時，最好是在電腦程式的一特定點上定義資料或資料類型，並在該電腦程式中遠離該資料定義點的各不同點上利用該資料或資料類型。在

五、發明說明(2　)

文件說明中，可能並未在資料定義點上提及在遠端點上所需的資料相依性。未來要修改該電腦程式時，執行修改的程式設計師可能無法輕易得知該等資料相依性，因而可能在程式中發生錯誤。

電腦程式設計師通常使用可用來驗證執行時資料相依性之各種程式語言功能。此種電腦程式語言功能的一個例子爲許多C程式語言的編譯器所提供的程式庫中包含之指稱巨集(asert macro)。此種巨集或程式語言功能將評估於執行發現有該敘述或巨集的程式時在該點上的資料結構之指定特徵。此種方式的困難處在於：在電腦程式中導入了編譯器最佳化作業在某些情形中原本可儘量減少或消除的額外程式碼。爲了消除該額外的程式碼，必須在程式開發的測試階段之後於長個程式中重新執行去除驗證巨集或敘述之步驟。

該習用技術方式的另一個困難處在於：因爲係在執行時進行驗證，所以如果在程式測試時並未執行的一特定程式碼段落中不正確地使用受資料相依性支配的資料，則可能在實際使用該程式一段時間之後，才會發現該錯誤。此外，單一次的程式測試將無法確認一些錯誤；必須持續地測試該程式，才能發現每一個此類錯誤。最後，在執行時之前，無法利用諸如指稱巨集而進行錯誤確認。

因此，目前需要一種改良式編譯器、組譯器、或相關工具程式，用以驗證是否正確地使用了受資料相依性支配的資料，而不會耗用額外的執行時間，也不會增加執行時程

五、發明說明 (3)

式碼，且可一次確認各資料相依性錯誤，而且該編譯器並不依賴程式碼的執行時測試即可發現此類錯誤。

發明概述

根據本發明的一個面向，提供了一種改良式編譯器。

根據本發明的另一面向，提供了一種可執行資料相依性的執行時驗證之改良式編譯器。

根據本發明的又一面向，提供了一種包含資料驗證模組之編譯器延伸程式，用以編譯一電腦程式中包含的一資料相依性表示式，其中該資料相依性表示式反映了該電腦程式中各資料間之一程式設計師界定的關係，且於該程式符合各資料間之該程式設計師界定的關係時，該資料相依性表示式具有一第一組預定值中之一預定值，而於該程式不符合各資料間之該程式設計師界定的關係時，該資料相依性表示式具有不同於該第一組預定值的一第二組預定值中之一預定值，其中於編譯該資料相依性表示式時，該資料驗證模組並不產生執行時程式碼或執行時資料儲存分配，且於編譯該資料相依性表示式時，若該資料相依性表示式之值係在該第二組預定值中，則該資料驗證模組產生一編譯時錯誤訊息，且若該資料相依性表示式之值係在該第一組預定值中，則該資料驗證模組不會產生編譯時錯誤訊息。

根據本發明的又一面向，該編譯器延伸程式包含一資料驗證模組，該資料驗證模組又包含一個將資料相依性表示式的參考位址加入該編譯時錯誤訊息之模組。

五、發明說明(4　)

根據本發明的又一面向，提供了一種用於一程式語言編譯器之編譯器延伸程式，該編譯器延伸程式包含一資料驗證結構，該資料驗證結構接受一電腦程式中之一資料相依性表示式作爲一引數，其中該資料相依性表示式反映了該電腦程式中各資料間之一程式設計師界定的關係，且於該程式符合各資料間之該程式設計師界定的關係時，該資料相依性表示式具有一第一組預定值中之一預定值，而於該程式不符合各資料間之該程式設計師界定的關係時，該資料相依性表示式具有不同於該第一組預定值的一第二組預定值中之一預定值，其中於編譯該電腦程式時，若該資料相依性表示式之值係在該第二組預定值中，則於該資料驗證結構的編譯中該程式語言編譯器將產生一編譯時錯誤訊息，且若該資料相依性表示式之值係在該第一組預定值中，則該資料驗證結構的編譯中該程式語言編譯器將不會產生編譯時錯誤訊息。

根據本發明的又一面向，上述的編譯器延伸程式將使該資料驗證結構於編譯時不會產生執行時程式碼。

根據本發明的又一面向，上述的編譯器延伸程式將使該資料驗證結構於編譯時不會分配執行時資料儲存。

根據本發明的一個面向，提供了一種於編譯時驗證程式設計師界定的資料相依性之改良式方法，該方法包含下列步驟：

a) 界定一電腦程式中之一檢查敘述，其中該檢查敘述具有一引數；

五、發明說明 (5)

b) 將該檢查敘述插入該電腦程式中該電腦程式所使用的具有一程式設計師界定的資料相依性之一資料結構之各點；以及

c) 將一引數傳送到該檢查敘述，其中當該資料結構符合該程式設計師界定的資料相依性時，一資料相依性表示式具有一第一組預定值中之一預定值，且當該資料結構不符合該程式設計師界定的資料相依性時，該資料相依性表示式具有不同於該第一組預定值的一第二組預定值中之一預定值。

該檢查敘述具有一定義，使該資料相依性表示式之值在該第一組預定值中時，該檢查敘述將成功編譯，且於該資料相依性表示式之值在該第二組預定值中時，該檢查敘述將無法成功編譯。

根據本發明的上述方法之一個面向，該檢查敘述的成功編譯將造成最少的或沒有額外的執行時間虛耗。

根據本發明的上述方法之一個面向，該檢查敘述的成功編譯將不會造成額外的執行時程式碼之產生。

根據本發明的上述方法之一個面向，該檢查敘述的成功編譯將不會造成額外的執行時資料儲存分配。

本發明的優點包含一個於編譯時間使用受資料相依性支配的資料時可確認錯誤之編譯器或組譯器。本發明提供了在不導入額外的執行時資料儲存成本或額外的執行時程式碼之情形下確認此種錯誤之優點。另一優點包含一編譯器，該編譯器不只可確認位於執行時間程式測試時所執行

五、發明說明（6　）

的程式碼中之錯誤，且提供了確認編譯時所有的資料相依性錯誤之機會。

附圖簡述

各附圖中示出本發明的各較佳實施例，這些附圖有：

圖1是根據本發明較佳實施例而撰寫的一電腦程式的結構之方塊圖；

圖2是本發明較佳實施例的編譯時程序流動之流程圖；以及

圖3示出根據本發明較佳實施例作業的一編譯器所產生的具有編譯器錯誤訊息之電腦程式列表。

在這些附圖中，係以舉例方式示出本發明之各較佳實施例。我們當可明確了解，這些說明及圖示之目的只在於舉例並有助於了解，並非在界定本發明之限制。

較佳實施例之詳細說明

請參閱圖1，圖中示出只有高階描述之方塊圖，亦即示出將由一編譯器編譯的一電腦程式及本發明較佳實施例的編譯器延伸程式之結構表示法。該例示電腦程式包含一程式部分10及界定受資料相依性支配的資料結構之資料定義部分12。部分14是一部分的程式碼或模組，包含用來驗證資料相依性之程式碼、及於執行時有賴於所界定的資料相依性之程式碼(例如部分16)。部分18及20代表電腦程式碼的其他部分，亦包含用來驗證資料相依性之程式碼、及在部分20中有賴於資料相依性之程式碼。

在圖2中，一編譯時流程圖以示意方式示出根據本發明

五、發明說明 (7　)

較佳實施例的編譯器程序之編譯時流程。步驟30表示開始的編譯時程序。步驟32表示一驗證巨集定義之編譯。步驟34表示資料定義之編譯，其中該等資料係受資料相依性的支配。步驟36表示程式碼中所設驗證巨集之巨集擴充。步驟38表示編譯器因在巨集擴充中發現一錯誤而產生的錯誤訊息。步驟40表示利用資料相依性的程式碼之編譯。

在圖3中，示出用來說明本發明的編譯器實例之程式碼。可以巨集程式庫所提供的一模組之方式實施本發明之編譯器改良部分，或者可以建入電腦程式本身的一編譯器延伸程式之方式實施本發明之編譯器改良部分。在本發明的說明中，將每一個此種實施方式稱爲一編譯器延伸程式。當以設於編譯器本身中之一延伸程式之方式實施時，本發明之實施例將造成編譯器延伸程式本身於編譯時產生錯誤訊息。當以與編譯器分離的一編譯器延伸程式之方式實施時，本發明將利用一資料驗證結構，編譯該資料驗證結構時將造成編譯器產生適當的錯誤訊息。亦可將本發明實施爲圖3所示之一獨立巨集定義。

熟悉本門技術者將可了解利用前文所述或其他標準程式設計方法而實施本發明所需之修改。熟悉本門技術者亦可了解，本發明可適用於編譯器及組譯器。熟熟悉本門技術者亦可了解，此處係參照編譯器而說明本發明，但提及編譯器時亦包括組譯器。在本詳細說明中，將說明一較佳實施例，其中係利用高階電腦語言C++中一巨集定義之方式

五、發明說明（8）

實施編譯器改良部分。在利用一巨集定義實施的本發明之較佳實施例中，係定一個接受一表示式作爲引數或參數之巨集。該巨集定義可擴充到一個將由編譯器編譯之敘述，但是所編譯的敘述不會產生執行時之記憶體分配、或於執行時需要執行的程式碼。在圖3所示的較佳實施例中，該巨集定義包含一類型定義，該類型定義是隨該巨集的參數或引數而變之一大小陣列。於編譯該驗證巨集定義時，編譯器產生一個接受一引數或參數之編譯時巨集。於編譯期間擴充該巨集時，參數表示式(資料驗證表示式)之值將根據該資料驗證表示式之值，而引發一個可成功編譯之敘述，或引發一個將在編譯器中產生一錯誤訊息之敘述。可將程式設計師所使用的巨集視爲一個接受一資料驗證表示式之檢查敘述，因而檢查敘述之成功編譯係取決於該資料驗證表示式之值。根據本發明之較佳實施例，如圖2中之步驟34所示，係以平常的方式編譯受資料相依性支配的資料定義。該程式碼將包含以有賴於程式設計師界定的關係之方式操作資料之各部分。這些部分的例子係示於圖1中之部分16及20。

本發明之較佳實施例以下文所述之方式提供編譯時的資料相依性驗證。圖2所示步驟32中界定的巨集係包含於該程式碼中以依照資料相依性規則而操作資料之程式碼部分。該部分係示於圖1中之部分14及18。當以圖2中步驟36所示之方式擴充該巨集時，將界定資料相依性之各表示式傳送到該巨集，作爲參數。其中情形係示於圖3中之

五、發明說明（9）

24、26、31、及33行。在圖3所示實例中，各資料相依性表示式利用巨集"Verify"作爲引數而，該程式碼的上述各行示出：結構x及結構y的大小是相同的，且資料項目(x，i)及(x，j)是相鄰的。在第一個實例中，係利用"Verify"巨集作爲檢查敘述，其中係以表示式"sizeof(x) == sizeof(y)"作爲表示各資料間之程式設計師界定的關係之資料相依性表示式。

在圖3所示實例中，傳送作爲參數的表示式之值爲"眞"的巨集擴充將繼續進行，而不會發生編譯時錯誤。因此，將界定一資料類型。一資料類型定義在記憶體中或執行時並無執行時間虛耗，因而最終執行程式碼中不會有導入之虛耗。然而，當傳送作爲引數之表示式之值爲"假"時，編譯器將確認一編譯時錯誤。

因此，當以一與所界定的資料相依性不一致之方式改變一資料結構時，將產生一編譯時錯誤，且程式設計師將收到程式中有破壞資料相依性的一改變之警示，而且將在程式碼中導入相依性的該點上顯示該警示(或錯誤訊息)。不論出現哪一種情形，如圖2之流程圖所示，將編譯有資料相依性之程式碼。

如圖3所示，當作爲"Verify"巨集的引數之各表示式之值不是眞值時，將產生編譯時錯誤訊息。請注意，圖3省略了使用受資料相依性支配的資料結構之程式碼。我們當了解，在一個使用Verify巨集之程式中，Verify巨集將如圖2所示繼續進行使用該資料結構之程式碼。

五、發明說明 (10)

我們當了解，亦可將其他的結構用於巨集定義。只要此種結構不會產生執行時記憶體分配或可執行程式碼，則將可獲致較佳實施例中所述的本發明之優點。

亦如前文所述，經由非利用巨集定義或程式庫而得到的一編譯器延伸程式，或經由將新的程式語言功能程式碼寫入編譯器本身，而加入圖3所示的各種功能，即可獲致在編譯時偵測資料相依性錯誤之方法。熟悉編譯器結構技術者將可了解此種替代方法之實施方式。此種實施例之優點在於：可將該編譯器設計成可產生由該編譯器所調整的一錯誤訊息或警示，以便將與電腦程式碼無法得知但會產生錯誤訊息的資料相依性相關之資訊提供給程式設計師。該錯誤訊息可包含與資料驗證表示式有關的或由資料驗證表示式衍生的資訊。雖然已詳述本發明的各種較佳實施例，但是熟悉本門技術者當可了解，在不脫離下列申請專利範圍所述的本發明之精神或範圍下，尚可對本發明作出各種變化。

四、中文發明摘要（發明之名稱：編譯時間資料相依性驗證）

本發明揭示了一種於編譯時驗證程式設計師界定的資料相依性之編譯器延伸程式。該編譯器延伸程式提供一檢查敘述，該檢查敘述係以一資料驗證表示式作爲一引數。當該程式設計師界定的資料相依性符合該程式時，該資料驗證表示式具有一預定值。如果於編譯該檢查敘述時，該資料驗證表示式並未具有該預定值，則將產生一編譯時錯誤。該檢查敘述的成功編譯不會產生執行時程式碼或執行時資料分配。

英文發明摘要（發明之名稱："COMPILE-TIME DATA DEPENDENCY VERIFICATION"）

A compiler extension for the compile-time verification of programmer-defined data dependencies. The compiler extension provides for a check statement which takes as an argument a data-verification expression. The data-verification expression has a predetermined value when the programmer-defined data dependency is conformed to in the program. A compile-time error is generated if the data-verification expression does not have the predetermined value on the compilation of the check statement. The successful compilation of the check statement does not result in run-time code or run-time data allocation occurring.

六、申請專利範圍

1. 一種包含資料驗證模組之編譯器延伸程式，用以編譯一電腦程式中包含的一資料相依性表示式，其中該資料相依性表示式反映了該電腦程式中各資料間之一程式設計師界定的關係，且於該程式符合各資料間之該程式設計師界定的關係時，該資料相依性表示式具有一第一組預定值中之一預定值，而於該電腦程式不符合各資料間之該程式設計師界定的關係時，該資料相依性表示式具有不同於該第一組預定值的一第二組預定值中之一預定值，其中於編譯該資料相依性表示式時，該資料驗證模組並不產生執行時程式碼或執行時資料儲存分配，且於編譯該資料相依性表示式時，若該資料相依性表示式之值係在該第二組預定值中，則該資料驗證模組產生一編譯時錯誤訊息，且若該資料相依性表示式之值係在該第一組預定值中，則該資料驗證模組不會產生編譯時錯誤訊息。
2. 如申請專利範圍第1項之編譯器延伸程式，其中該資料驗證模組又包含一個將資料相依性表示式的參考位址加入該編譯時錯誤訊息之模組。
3. 如申請專利範圍第1項之編譯器延伸程式，該編譯器延伸程式係用於支援程式設計師界定的資料類型的一電腦語言之編譯器，其中該資料驗證模組將該編譯器的資料類型用於編譯該資料相依性表示式。
4. 一種用於一程式語言編譯器之編譯器延伸程式，該編譯器延伸程式包含一資料驗證結構，該資料驗證結構接受

六、申請專利範圍

一電腦程式中之一資料相依性表示式作爲一引數，其中：

該資料相依性表示式反映了該電腦程式中各資料間之一程式設計師界定的關係，

於該電腦程式符合各資料間之該程式設計師界定的關係時，該資料相依性表示式具有一第一組預定值中之一預定值，且於該程式不符合各資料間之該程式設計師界定的關係時，該資料相依性表示式具有不同於該第一組預定值的一第二組預定值中之一預定值，

其中於編譯該電腦程式時，若該資料相依性表示式之值係在該第二組預定值中，則於該資料驗證結構的編譯中該程式語言編譯器將產生一編譯時錯誤訊息，且若該資料相依性表示式之值係在該第一組預定值中，則該資料驗證結構的編譯中該程式語言編譯器將不會產生編譯時錯誤訊息。

5. 如申請專利範圍第4項之編譯器延伸程式，其中該該資料驗證結構的編譯不會產生執行時程式碼。

6. 如申請專利範圍第4項之編譯器延伸程式，其中該資料驗證結構的編譯不會分配執行時資料儲存。

7. 如申請專利範圍第4項之編譯器延伸程式，其中該程式語言編譯器支援程式設計師界定的資料類型，且其中該資料驗證結構採用該程式語言編譯器之資料類型，使該程式語言編譯器產生錯誤訊息。

8. 如申請專利範圍第7項之編譯器延伸程式，其中該編譯

六、申請專利範圍

器延伸程式係包含在一C++程式語言編譯器的一巨集程式庫中。

9. 一種於編譯時驗證程式設計師界定的資料相依性之方法，該方法包含下列步驟：

a) 界定一電腦程式中之一檢查敘述，其中該檢查敘述具有一引數，

b) 將該檢查敘述插入該電腦程式中該電腦程式所使用的具有一程式設計師界定的資料相依性之一資料結構之各點，以及

c) 將一引數傳送到該檢查敘述，其中當該資料結構符合該程式設計師界定的資料相依性時，一資料相依性表示式具有一第一組預定值中之一預定值，且當該資料結構不符合該程式設計師界定的資料相依性時，該資料相依性表示式具有不同於該第一組預定值的一第二組預定值中之一預定值，

其中該檢查敘述具有一定義，使該資料相依性表示式之值在該第一組預定值中時，該檢查敘述將成功編譯，且於該資料相依性表示式之值在該第二組預定值中時，該檢查敘述將無法成功編譯。

10. 如申請專利範圍第9項之方法，其中該檢查敘述的成功編譯將造成最少的額外之執行時間虛耗。

11. 如申請專利範圍第9項之方法，其中該檢查敘述的成功編譯將不會造成額外的執行時程式碼之產生。

12. 如申請專利範圍第9項之方法，其中該檢查敘述的成功

六、申請專利範圍

編譯將不會造成額外的執行時資料儲存分配。

13. 如申請專利範圍第9、10、11、或12項之方法，其中該檢查敘述是一編譯時巨集。

14. 如申請專利範圍第13項之方法，其中該編譯時巨集是一程式設計師界定的資料類型定義。

15. 一種電腦可讀取之記憶體，用以儲存用於在一個根據申請專利範圍第9至14項所述方法中任一方法的電腦中執行的指令。

87118409228

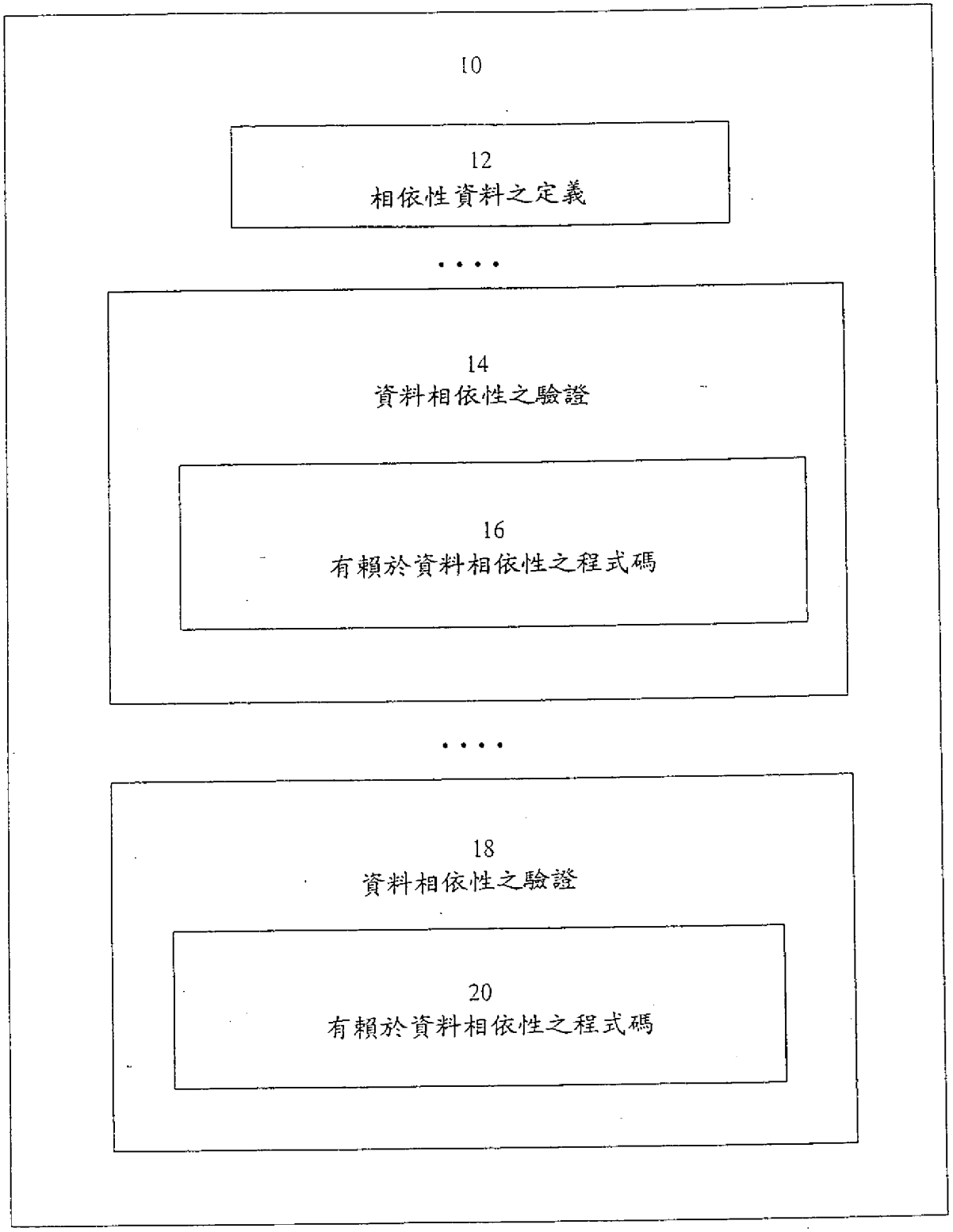


圖 1

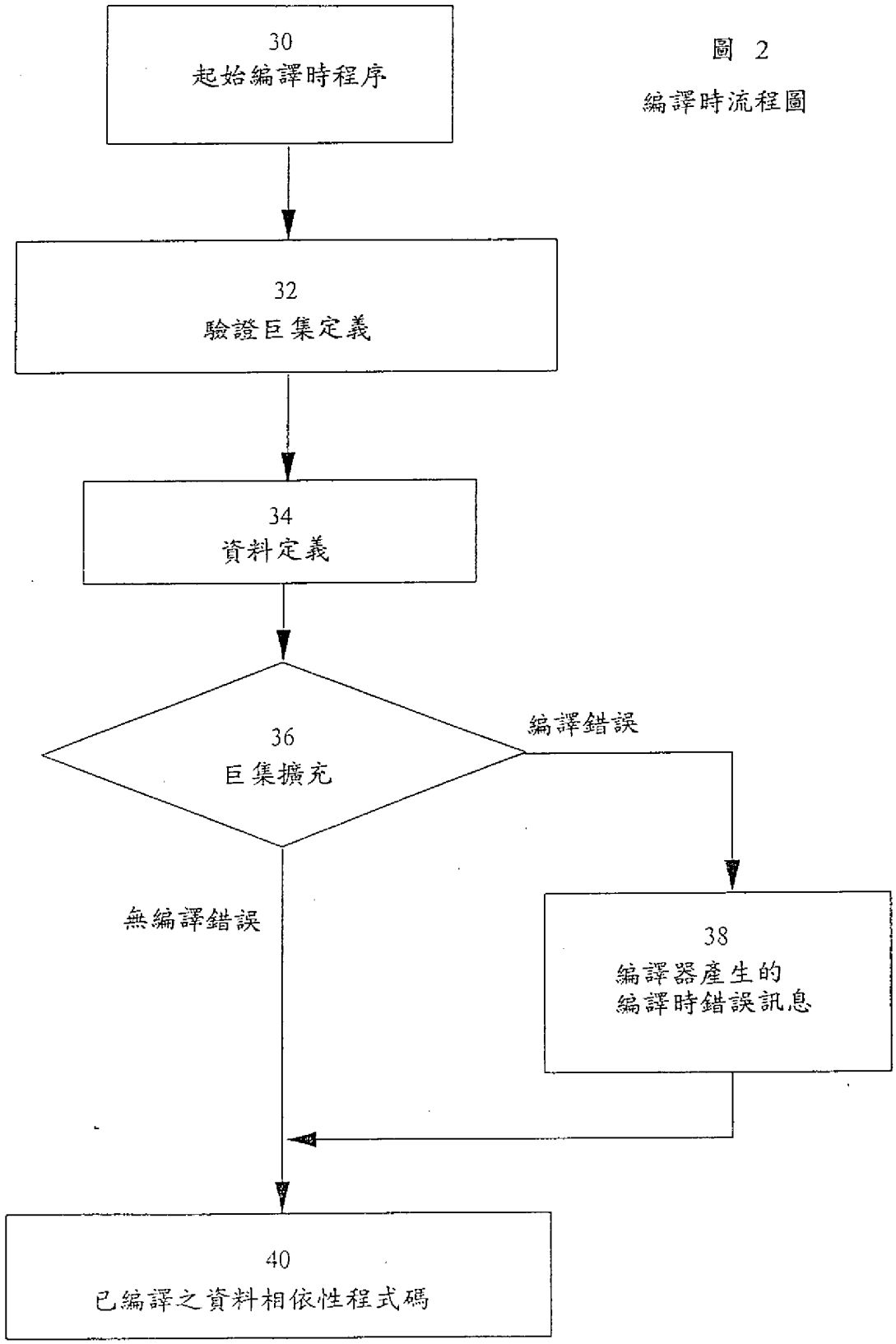

30
起始編譯時程序
32
驗證巨集定義
34
資料定義
36
巨集擴充
編譯錯誤
無編譯錯誤
38
編譯器產生的
編譯時錯誤訊息
40
已編譯之資料相依性程式碼

圖 2

編譯時流程圖

```
LINE  STMT
       *...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8....+....9.....
   1   |#include <stddef.h>
   2   |
   3   |#define Verify(i) typedef int __Compile_Time_Assertion_is_Not_True__&lbrk.!!(i)&rbrk.
   4   |
   5   |
   6   |struct x {
   7   |   int i;
   8   |   int j;
   9   |   int k;
  10   |   int l;
  11   |};
  12   |
  13   |struct y {
  14   |   long double d;
  15   |};
  16   |
  17   |struct z {
  18   |   float f;
  19   |};
  20   |
  21   |void good() {
  22   |
  23   |   // check that x & y are the same size (true)
  24   |   Verify(sizeof(x) == sizeof(y));
  25   |   // check that x.i and x.j are adjacent (true)
  26   |   Verify((offsetof(x,i)+sizeof(((x*)NULL)->i))==offsetof(x,j));
  27   |};
  28   |
  29   |void bad() {
  30   |   // check that x & z are the same size (false)
  31   |   Verify(sizeof(x) == sizeof(z));
vfytest.cpp(31:1) : error EDC3038: The array bound must be a positive integral constant expression.
  32   |   // check that x.i and x.k are adjacent (false)
  33   |   Verify((offsetof(x,i)+sizeof(((x*)NULL)->i))==offsetof(x,k));
vfytest.cpp(33:1) : error EDC3038: The array bound must be a positive integral constant expression.
  34   |};
```

図 3

409228

87118177

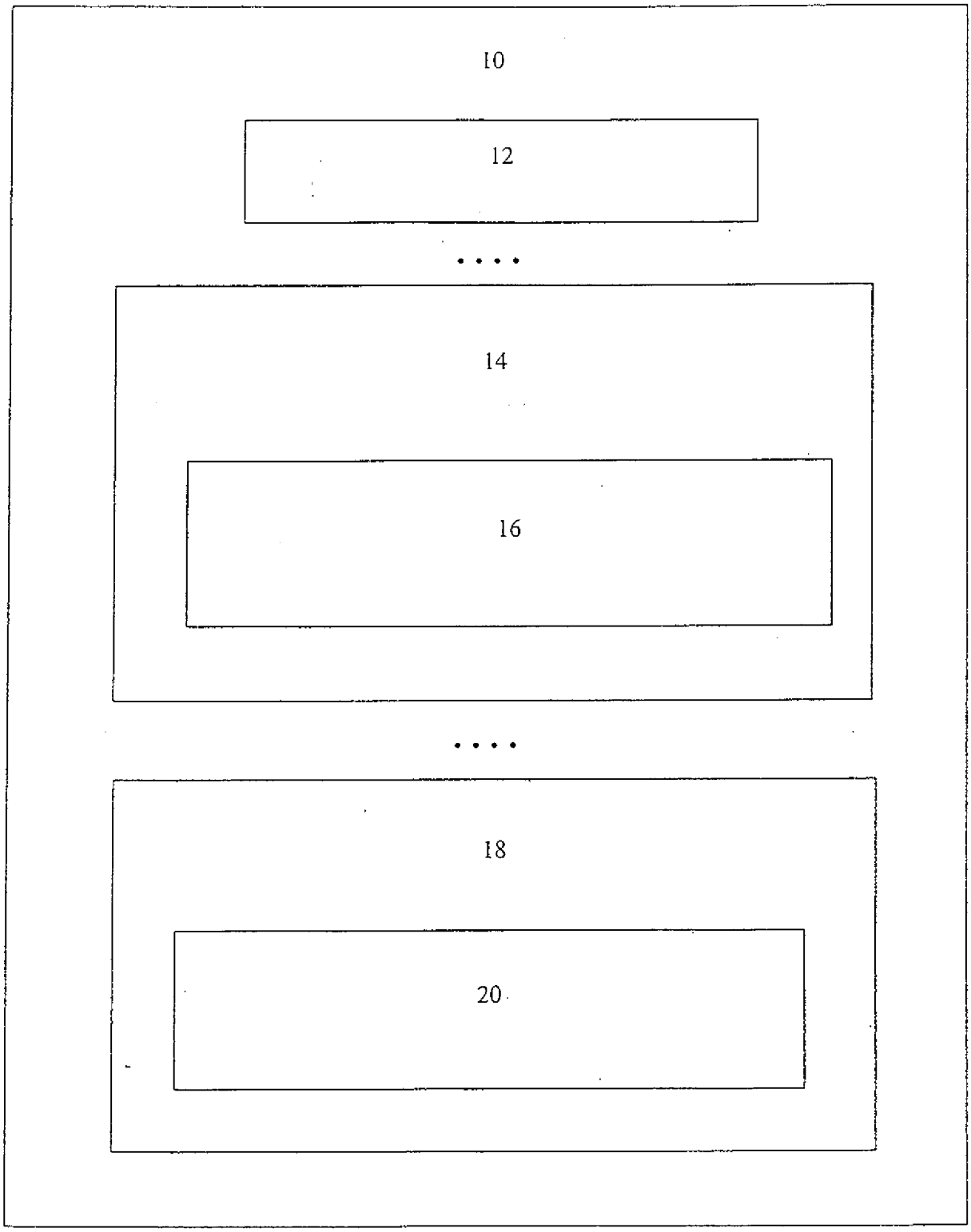
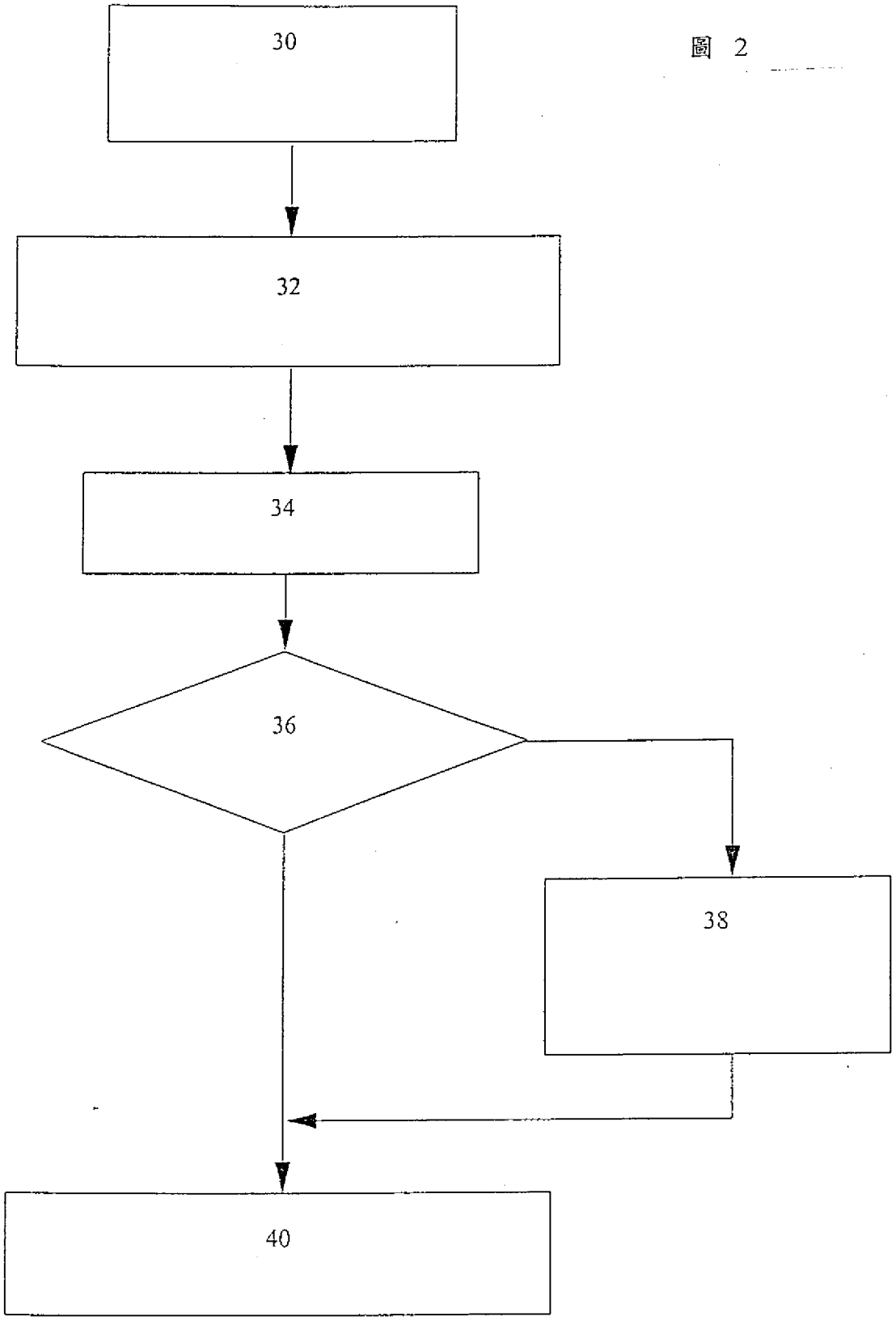


圖 1

圖 2



```
LINE  STMT
       *...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8....+....9.....
    1  |#include <stddef.h>
    2  |
    3  |#define Verify(i) typedef int __Compile_Time_Assertion_is_Not_True__&lbrk.!!(i)&rbrk.
    4  |
    5  |
    6  |struct x {
    7  |   int i;
    8  |   int j;
    9  |   int k;
   10  |   int l;
   11  |};
   12  |
   13  |struct y {
   14  |   long double d;
   15  |};
   16  |
   17  |struct z {
   18  |   float f;
   19  |};
   20  |
   21  |void good() {
   22  |
   23  |   // check that x & y are the same size (true)
   24  |   Verify(sizeof(x) == sizeof(y));
   25  |   // check that x.i and x.j are adjacent (true)
   26  |   Verify((offsetof(x,i)+sizeof(((x*)NULL)->i))==offsetof(x,j));
   27  |};
   28  |
   29  |void bad() {
   30  |   // check that x & z are the same size (false)
   31  |   Verify(sizeof(x) == sizeof(z));
vfytest.cpp(31:1) : error EDC3038: The array bound must be a positive integral constant expression.
   32  |   // check that x.i and x.k are adjacent (false)
   33  |   Verify((offsetof(x,i)+sizeof(((x*)NULL)->i))==offsetof(x,k));
vfytest.cpp(33:1) : error EDC3038: The array bound must be a positive integral constant expression.
   34  |};
```

圖 3