I849133

(19)中華民國智慧財產局

(12)發明說明書公告本　**(11)證書號數：TW I849133 B**

(45)公告日： 中華民國 113 (2024) 年 07 月 21 日

(21)申請案號：109117816　(22)申請日： 中華民國 109 (2020) 年 05 月 28 日

(51)Int. Cl.：　*G06F8/40　(2018.01)*　*G06F8/70　(2018.01)*

G06F9/44　(2018.01)

(30)優先權：2019/05/31　日本　2019-103138

(71)申請人：日商關連風科技股份有限公司 (日本) CONNECTFREE CORPORATION　(JP)

日本

(72)發明人：帝都　久利寿 TATE, KRISTOPHER ANDREW (US)

(74)代理人：洪澄文；洪茂

(56)參考文獻：

CN　105393216A　CN　107885503A

US　2013/0067155A1　US　2016/0147511A1

審查人員：姚乃綺

申請專利範圍項數：5 項　圖式數：10　共 33 頁

(54)名稱

軟體開發裝置和軟體開發程式

(57)摘要

對在邊緣元件等所被執行的程式，提供可自由設定各種制約的環境。由原始碼生成目的碼的軟體開發裝置係包含：評估手段，其係抽出原始碼所設定的制約，在該所抽出的制約的適用範圍中，評估該原始碼是否適合於該制約；及生成手段，其係以適合於制約的方式生成目的碼。

指定代表圖：

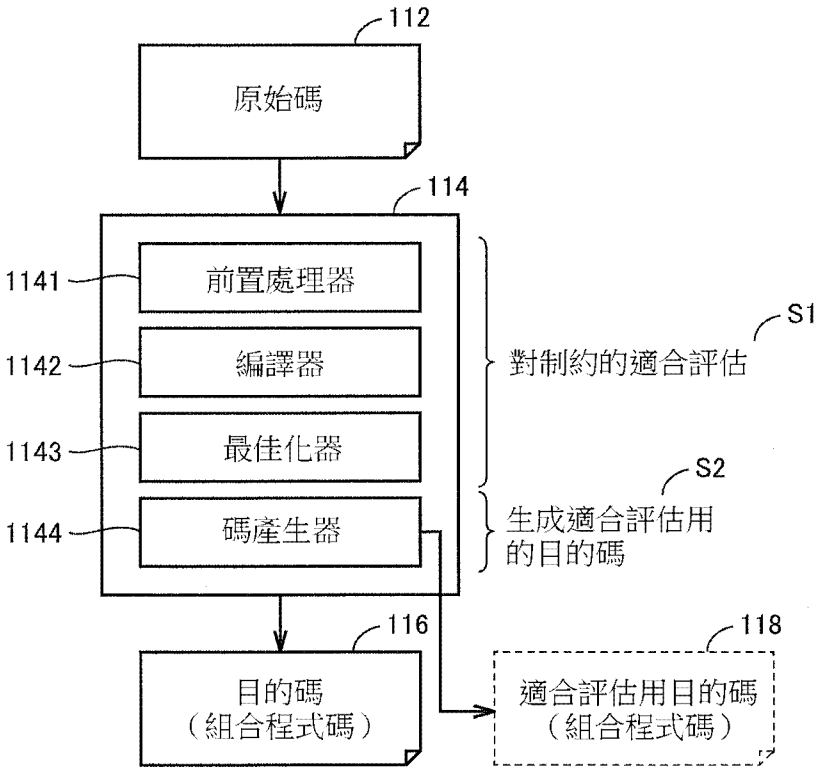


圖7

符號簡單說明：

112:原始碼

114:軟體開發程式

116:目的碼

118:適合評估用目的碼

1141:前置處理器

1142:編譯器

1143:最佳化器

1144:碼產生器

公告本

I849133

【發明摘要】

【中文發明名稱】　軟體開發裝置和軟體開發程式

【中文】

對在邊緣元件等所被執行的程式，提供可自由設定各種制約的環境。由原始碼生成目的碼的軟體開發裝置係包含：評估手段，其係抽出原始碼所設定的制約，在該所抽出的制約的適用範圍中，評估該原始碼是否適合於該制約；及生成手段，其係以適合於制約的方式生成目的碼。

【指定代表圖】　圖7

【代表圖之符號簡單說明】

112:原始碼

114:軟體開發程式

116:目的碼

118:適合評估用目的碼

1141:前置處理器

1142:編譯器

1143:最佳化器

1144:碼產生器

【發明說明書】

【中文發明名稱】　軟體開發裝置和軟體開發程式

【技術領域】

【0001】　本揭示係關於軟體開發裝置和軟體開發程式。

【先前技術】

【0002】　近年的資訊通訊技術（Information and Communication Technology：ICT）明顯進步，連接於網際網路等網路的元件並非侷限於習知的個人電腦或智慧型手機等資訊處理裝置，擴展於各種物（things）。如上所示之技術趨勢係被稱為「IoT（Internet of Things；物聯網）」，各種技術及服務不斷被提案及實用化。將來係假想地球上數十億人與數百億或數兆的元件同時相連的世界。為了實現如上所示之網路化的世界，必須提供可更簡單、更安全、更自由地相連的解決方案（solution）。

【0003】　伴隨在IoT所被利用的元件（亦稱為「邊緣元件」）的智慧化，必須作成各種種類的程式。另一方面，可在成為對象的元件中利用的資源與個人電腦等相比較，大多成為受到限制者。

【0004】　以考慮可利用資源來作成程式的方法之一例而言，日本特開2004-038956號公報係揭示將用以發現在各種計算元件中可利用的計算資源、且將如此之資源藉由軟體應用程式而外露為可指定位址的服務的系統。

〔先前技術文獻〕

〔專利文獻〕

【0005】　〔專利文獻1〕日本特開2004-038956號公報

【發明內容】

〔發明所欲解決之課題〕

【0006】 在作成在邊緣元件所執行的程式時，必須由可利用資源及安全性的觀點來考慮各方面。但是，上述專利文獻1係僅著重在考慮在對象的計算元件可利用的計算資源來作成程式，並非為在作成程式時，對必須考慮各方面等課題提供解決方案者。

〔用以解決課題的手段〕

【0007】 按照本揭示之一形態，提供一種由原始碼生成目的碼的軟體開發裝置。軟體開發裝置係包含：評估手段，其係抽出原始碼所設定的制約，在該所抽出的制約的適用範圍中，評估該原始碼是否適合於該制約；及生成手段，其係以適合於前述制約的方式生成目的碼。

【0008】 生成手段亦可在制約的適用範圍中，若評估原始碼不適合於制約，即中止目的碼的生成。

【0009】 生成手段亦可在制約的適用範圍中，若無法評估原始碼是否適合於制約，除了對應原始碼的目的碼之外，在該目的碼執行中，生成用以評估是否適合於該制約的其他目的碼。

【0010】 制約亦可包含：對目的碼執行時所利用的資源的限制或規則、對目的碼的執行狀態的限制或規則、對目的碼的執行順序的限制或規則、以及對原始碼所包含的命令的限制或規則之中任1個。

【0011】 評估手段亦可若在制約的適用範圍包含叫出命令，關於藉由該叫出命令所被叫出的命令，亦評估是否適合於制約。

【0012】 按照本揭示的其他形態，提供一種由原始碼生成目的碼的軟體開發程式。軟體開發程式係使電腦執行：抽出原始碼所設定的制約，在該所抽出的制約的適用範圍中，評估該原始碼是否適合於該制約的步驟；及以適合於制

約的方式生成目的碼的步驟。

〔發明之效果〕

【0013】 藉由本揭示，可提供可對在邊緣元件等所被執行的程式，自由設定各種制約的環境。

【圖式簡單說明】

【0014】

〔圖1〕係顯示按照本實施形態之IoT系統的全體構成之一例的模式圖。

〔圖2〕係顯示按照本實施形態之軟體開發裝置的硬體構成例的模式圖。

〔圖3〕係顯示按照本實施形態之控制器的硬體構成例的模式圖。

〔圖4〕係用以說明在按照本實施形態之軟體開發裝置中可設定的制約的圖。

〔圖5〕係用以說明在按照本實施形態之軟體開發裝置可設定的制約的作用域的圖。

〔圖6〕係用以說明在執行時必須判斷在按照本實施形態之軟體開發裝置可設定的制約的適合的情形的圖。

〔圖7〕係顯示按照本實施形態之軟體開發裝置所提供的功能構成的區塊圖。

〔圖8〕係用以說明實現在按照本實施形態之軟體開發裝置中對於對叫出的制約的適合的方法的圖。

〔圖9〕係顯示按照本實施形態之軟體開發裝置中由原始碼生成目的碼的處理順序的流程圖。

〔圖10〕係顯示按照本實施形態之軟體開發裝置所提供的使用者介面畫面之一例的圖。

【實施方式】

【0015】 一邊參照圖示，一邊詳細說明本揭示之實施形態。其中，關於圖中相同或相當部分，係標註相同符號而不反覆其說明。

【0016】 在以下說明中，以典型例而言，說明將按照本實施形態之軟體開發裝置100適用在IoT系統的情形，惟本揭示係可適用於任意系統及控制器，而非侷限於IoT系統。

【0017】 ＜A.IoT系統1＞

首先，說明包含按照本實施形態之軟體開發裝置100及邊緣元件2的IoT系統1的全體構成。

【0018】 圖1係顯示按照本實施形態之IoT系統1的全體構成之一例的模式圖。參照圖1，在IoT系統1中，典型而言，生成在軟體開發裝置100中在邊緣元件2所被執行的程式（目的碼）。所生成的程式係由軟體開發裝置100被轉送至邊緣元件2所包含的控制器200。

【0019】 在軟體開發裝置100係提供整合開發環境（IDE：Integrated Development Environment），使用者係可在整合開發環境上作成任意程式。亦即，軟體開發裝置100係由使用者任意作成的原始碼來生成目的碼。

【0020】 以邊緣元件2而言，亦可為任何元件，但是典型而言，假想工廠設備、家庭內的各種裝置、社會基礎設施設備、車輛等移動體、任意攜帶式元件等。如後所述，控制器200係具有處理器，可執行來自軟體開發裝置100的程式。

【0021】 說明IoT系統1中的處理順序之一例。首先，使用者使用軟體開發裝置100，作成原始碼（（1）原始碼作成）。接著，所作成的原始碼係在軟體開發裝置100中被編譯，且生成目的碼（（2）目的碼生成）。所生成的目的碼係被轉送至邊緣元件2的控制器200（（3）目的碼轉送）。所被轉送的目的碼係在控制器200被執行（（4）目的碼執行）。

【0022】 藉由如上所示之順序，可使在軟體開發裝置100中所開發的任意程式在控制器200中執行。

【0023】 如後所述，按照本實施形態之軟體開發裝置100係對在控制器200所執行的程式，提供可自由設定各種制約的環境。典型而言，可按照執行程式的控制器200所具有的資源、邊緣元件2的種類及用途、所執行的程式的重要性及應確保的安全性等，對程式任意設定各種制約。藉由如上所示之制約的設定，可實現利用有限資源的應用程式的實現、或應用程式之未意圖的舉動的防止等。

【0024】 ＜B.硬體構成例＞

接著，說明按照本實施形態之IoT系統1所包含的元件的硬體構成例。

【0025】 （b1：軟體開發裝置100）

典型而言，軟體開發裝置100係以通用電腦來實現。

【0026】 圖2係顯示按照本實施形態之軟體開發裝置100的硬體構成例的模式圖。參照圖2，軟體開發裝置100係包含處理器102、主記憶體104、輸入部106、顯示器108、硬碟110、及通訊介面122，作為主要的元件（component）。該等元件係透過內部匯流排120而相連接。

【0027】 處理器102係由例如CPU（Central Processing Unit，中央處理單元）或GPU（Graphics Processing Unit，圖形處理單元）等所構成。亦可配置複數處理器102，亦可採用具有複數核心（core）的處理器102。

【0028】 主記憶體104係由DRAM（Dynamic Random Access Memory，動態隨機存取記憶體）或SRAM（Static Random Access Memory，靜態隨機存取記憶體）等揮發性記憶裝置所構成。硬碟110係保持在處理器102所執行的各種程式或各種資料。其中，亦可採用SSD（Solid State Drive，固體狀態驅動機）或快閃記憶體等非揮發性記憶裝置，來取代硬碟110。儲存在硬碟110的程式之中所被指定的程式在主記憶體104上展開，且處理器102係依序執行在主記憶體104上所

展開的程式所包含的電腦可讀命令（computer-readable instructions），藉此實現如後所述的各種功能。

【0029】 典型而言，在硬碟110係儲存：使用者任意作成的原始碼112、用以實現整合開發環境的軟體開發程式114；及由原始碼112所生成的目的碼116。軟體開發程式114係由使用者任意作成的原始碼112生成目的碼116者，包含提供程式的開發環境的模組。

【0030】 輸入部106係受理操作軟體開發裝置100的使用者的輸入操作。輸入部106亦可為例如鍵盤、滑鼠、顯示元件上所配置的觸控面板、配置在軟體開發裝置100的框體的操作按鍵等。

【0031】 顯示器108係顯示在處理器102的處理結果等。顯示器108亦可為例如LCD（Liquid Crystal Display，液晶顯示器）或有機EL（Electro-Luminescence，電激發光）顯示器等。

【0032】 通訊介面122係擔任與控制器200的資料交換。通訊介面122係包含例如USB（Universal Serial Bus，通用串列匯流排）埠、IEEE1394等串列埠、傳統的並列埠等有線連接端子。或者，通訊介面122亦可包含乙太網路（註冊商標）埠。

【0033】 其中，軟體開發裝置100的全部或一部分亦可使用被組入相當於電腦可讀命令的電路的ASIC（Application Specific Integrated Circuit，特定功能積體電路）等硬佈線電路來實現。再或者，亦可在FPGA（field-programmable gate array，現場可程式化閘陣列）上使用相當於電腦可讀命令的電路來實現。此外，亦可將處理器102及主記憶體、ASIC、FPGA等適當組合來實現。

【0034】 軟體開發裝置100亦可另外具有：用以由儲存包含電腦可讀命令的軟體開發程式114的非暫態性（non-transitory）的媒體，讀出該所儲存的程式等的元件。媒體亦可為例如DVD（Digital Versatile Disc，數位影音光碟）等光學

媒體、USB記憶體等半導體媒體等。

【0035】 其中，軟體開發程式114係不僅透過媒體而被安裝在軟體開發裝置100，亦可由網路上的配訊伺服器所提供。

【0036】 （b2：控制器200）

控制器200亦可使用通用電腦來實現，亦可使用包含為實現處理為所需的元件的半導體基板來實現。

【0037】 圖3係顯示按照本實施形態之控制器200的硬體構成例的模式圖。參照圖3，控制器200係包含運算處理部210、無線通訊模組212、USB控制器214、通訊控制器216、及與1或複數墊片220作電性連接的IO驅動器218，作為主要的元件。

【0038】 運算處理部210係執行程式的運算部，包含：處理器202、主記憶體204、及快閃記憶體206，作為主要的元件。處理器202係由例如CPU或GPU等所構成。亦可配置複數處理器202，亦可採用具有複數核心的處理器202。主記憶體204係由DRAM或SRAM等揮發性記憶裝置所構成。快閃記憶體206係保持在處理器202所執行的程式或所需資料的非揮發性記憶裝置。在被儲存在快閃記憶體206的程式之中被指定的程式在主記憶體204上展開，藉由處理器202予以執行，藉此實現各種功能。

【0039】 無線通訊模組212係擔任與其他任意元件之間藉由無線所為之資料交換。無線通訊模組212亦可包含用以與元件、路由器、移動體基地台等進行無線通訊的處理電路及天線等。無線通訊模組212所對應的無線通訊亦可為例如Wi-Fi（註冊商標）、Bluetooth（藍牙，註冊商標）、ZigBee（註冊商標）、LPWA（Low Power Wide Area，低功率廣域）、GSM（註冊商標）、W-CDMA、CDMA200、LTE（Long Term Evolution，長期演進）、第5世代移動通訊系統（5G）的任一者。

【0040】 USB控制器214係擔任與軟體開發裝置100的資料交換。通訊控制器216係擔任與其他任意元件之間之藉由有線所為之資料交換。通訊控制器216亦可對應例如串列通訊、並列通訊、GPIO（General-purpose input/output）等周知的資料交換方式。

【0041】 IO驅動器218係擔任與透過墊片220而作電性連接的任意元件之間的電訊號的交換。IO驅動器218係按照來自運算處理部210的指令來輸出電訊號。此外，IO驅動器218係感測透過墊片220所被供予的電訊號，將該感測結果輸出至運算處理部210。更具體而言，IO驅動器218係由訊號生成電路、訊號感測電路、緩衝電路等所構成。

【0042】 控制器200亦可藉由來自未圖示的電池的電力予以驅動。

＜C.制約＞

接著，說明在按照本實施形態之軟體開發裝置100中可設定的制約。

【0043】 在本說明書中，「制約」係包含在執行由原始碼112所生成的目的碼116（組合程式碼）中應遵守的規則。以「制約」而言，係可包含：對目的碼116執行時所利用的資源的限制或規則、對目的碼116的執行狀態的限制或規則、對目的碼116的執行順序的限制或規則、以及對原始碼112所包含的命令的限制或規則等。

【0044】 圖4係用以說明在按照本實施形態之軟體開發裝置100中可設定的制約的圖。圖4（A）所示之原始碼112係用以將藉由預先定義的函數fn1（）所決定的輸出值寫入至所指定的位址的碼例。

【0045】 更具體而言，原始碼112係包含：輸出值變數的定義1121、及輸出值變數的定義1121。函數fn1（）的返回值被設定為輸出值OutValue（命令1123）。接著，輸出值OutValue的值被寫入至位址「0x1000」（命令1124）。

【0046】 可藉由對圖4（A）所示之原始碼112追加制約碼1125，來設定制

約。在圖4（B）所示之原始碼112中，追加用以指定可存取的記憶體範圍的制約碼1125。例如，「$allowedAddressRange=0x0000…0x0FFF」意指僅可對「0x0000」～「0x0FFF」的位址進行存取。

【0047】 若設定有如上所示之制約，用以將輸出值OutValue的值寫入至位址「0x1000」的命令1124並不適合於制約。亦即，命令1124係不可執行。

【0048】 圖4（B）所示之制約碼1125係例如在控制器200中，如「0x0000」～「0x0FFF」的記憶體範圍被設定為非安全區域，「0x1000」～「0x1FFF」的記憶體範圍被設定為安全區域的情形下為有效。

【0049】 以禁止不適合於制約的命令的執行的典型手法而言，假想（1）在由原始碼112生成目的碼116的過程中進行判斷的方法、及（2）生成目的碼116時進行判斷的方法。關於（1）的方法，評估用以生成前置處理器、編譯器、最佳化器等的目的碼116的功能適合於制約。另一方面，關於（2）的方法，除了由原始碼112所生成的目的碼116之外，亦可生成用以評估適合於制約的目的碼（以下亦稱為「適合評估用目的碼」）。適合評估用目的碼118亦可組入於目的碼116的一部分，亦可與目的碼116獨立存在。如上所示之構裝例的詳細內容後述。

【0050】 接著，說明按照本實施形態之制約的適用範圍（以下亦稱為「作用域（scope）」）。

【0051】 如圖4（B）所示，基本上，對於接續制約碼1125的記述的部分，適用制約碼1125所規定的制約。亦即，接續制約碼1125的記述的部分成為制約的作用域。該制約的作用域的結束位置係可任意設定，但是基本上，可將包含制約碼1125的括弧的範圍內形成為制約的作用域。

【0052】 此外，若程序或函數被叫出，關於該等被叫出的程序或函數，亦可形成為制約的作用域。

【0053】 圖5係用以說明在按照本實施形態之軟體開發裝置100中可設定的制約的作用域的圖。如圖5所示，針對叫出函數fn1（）而將函數fn1（）的返回值設定為輸出值OutValue的命令1123，係關於所被叫出的函數fn1（）的部分原始碼1126，亦包含在制約的作用域。

【0054】 如上所示，軟體開發裝置100係若在制約的作用域包含叫出命令，關於藉由該叫出命令所被叫出的命令（程序或函數），亦評估是否適合於制約。藉由依序繼承如上所示之制約的作用域，關於規定所被叫出的程序或函數的原始碼，亦藉由評估適合於制約，可確實遵守制約。

【0055】 圖6係用以說明在執行時必須判斷在按照本實施形態之軟體開發裝置100可設定的制約的適合的情形的圖。在圖6（A）中係顯示與圖4（B）所示之原始碼112相同的原始碼112。在圖6（A）所示之原始碼112中，寫入輸出值OutValue的值的位址被固定為「0x1000」（命令1124），將原始碼112進行詞句解析及句法解析，可評估為不適合於制約。

【0056】 相對於此，圖6（B）所示之原始碼112由於寫入輸出值OutValue的值的位址係使用輸出位址初期值IniPtr來決定，因此僅以原始碼112的解析，並無法單義決定寫入的位址。

【0057】 更具體而言，在圖6（B）所示之原始碼112中，除了輸出位址初期值變數的定義1127之外，規定決定輸出位址初期值IniPtr的處理1128。接著，輸出位址OutAddrs係在原始碼112規定決定規定在輸出位址初期值IniPtr加算「0F00」來決定的輸出位址的命令1129。接著，輸出值OutValue的值被寫入至藉由輸出位址OutAddrs所示的位址（命令1130）。

【0058】 在圖6（B）所示之原始碼112中，輸出位址OutAddrs的值係取決於輸出位址初期值IniPtr的值，在執行相對應的目的碼116的時點動態決定。

【0059】 按照本實施形態之軟體開發裝置100係提供在執行相對應的目的

碼116中，亦可評估適合於制約的構造（詳細內容後述）。

【0060】 ＜D.制約的種類＞

以按照本實施形態之制約而言，亦可採用如以下所示之種類者。

【0061】 〔表1〕

No	制約碼	設定值	說明
1	$allowedAddressRange	0x0000 ... 0xFFFF	指定可存取的記憶體範圍
2	$allowedScopeDepth	usize (default: none)	指定作用域為有效的深度 $allowSelfRecursion係被設定為False
3	$allowedStackDepth	usize (default: none)	指定作用域為有效的堆疊的深度 $allowSelfRecursion係被設定為False
4	$enableRuntimeSafety	True | False (default: True)	執行時間（runtime）的安全執行的有效/無效
5	$enforceStrictSwitches	True | False (default: True)	強制/不強制在（）陳述內的所有選項指定
6	$allowPanics	True | False (default: False)	容許/不容許發生恐慌
7	$allowUndefinedVariables	True | False (default: False)	可否使用未定義變數
8	$allowUnreachable	True | False (default: False)	容許/不容許無法到達經指定的收件端的狀態
9	$allowNoReturns	True | False (default: False)	容許/不容許無法取得返回值的狀態
10	$clearMemoryUponEnd	true | false (default: false)	若作用域結束，是否完全清除記憶體
11	$allowInputToOutput	True | False (default: True)	許容/不容許由經指定的記憶體的作用域返回
12	$allowConditionalBranches	True | False (default: True)	可否使用條件分歧
13	$allowSelfRecursion	True | False (default: True)	可否使用遞迴叫出
14	$allowPtrToZero	True | False (default: False)	可否使用表示0x0的指標
15	$allowRawPointers	True | False (default: False)	可否使用表示物理值的指標
16	$inlineFunctionCalls	True | False (default: False)	強制/不強制將叫出的function的全部內聯展開
17	$floatMode = @import("builtin").FloatMode	Strict/ Optimized (default:.Strict)	Strict：按照IEEE合規性的浮動小數點運算的執行 Optimized：對高速數學處理的最適化
18	$evalBranchQuota	Usize (default: 1000)	可向後（ backward ）的最大數
19	$allowSubWithOverflow	True | False (default: True)	容許/不容許伴隨溢位的減算處理
20	$allowMulWithOverflow	True | False (default: True)	容許/不容許伴隨溢位的乘算處理
21	$allowRem	True | False (default: True)	容許/不容許剩餘處理
22	$allowPtrCast	True | False (default: True)	容許/不容許型轉換處理
23	$allowMemset	True | False (default: True)	容許/不容許記憶體區塊設定

【0062】 〔表2〕

No	制約碼	設定值	說明
24	$allowIntToPtr	True | False (default: True)	容許/不容許對指標設定整數
25	$allowBreakpoint	True | False (default: True)	容許/不容許使用中斷點
26	$allowBitCas	True | False (default: True)	容許/不容許bitcast運算子
27	$allowExternalCalls	True | False (default: True)	容許/不容許外部函數的呼叫
28	$allowExternalVariables	True | False (default: True)	容許/不容許外部變數
29	$allowVarArgs	True | False (default: True)	容許/不容許將變數作引數的使用
30	$allowErrorTermination	True | False (default: True)	容許/不容許由作用域所產生的錯誤資料流
31	$allowGlobalVariables	True | False (default: True)	容許/不容許使用全域變數
32	$allowGlobalConstants	True | False (default: True)	容許/不容許使用全域常數
33	$allowIntermediateVariables	True | False (default: True)	容許/不容許使用中間變數/中間常數 *中間變數/中間常數意指在瞬前的作用域及後續的全域作用域所使用的構造體及列舉型的集合
34	$allowIntermediateConstants	True | False (default: True)	
35	$allowSuperVariables	True | False (default: True)	容許/不容許使用超變數/超常數 *超變數/超常數意指僅在現在的作用域所使用的構造體及列舉型的集合
36	$allowSuperConstants	True | False (default: True)	
37	$allowLocalVariables	True | False (default: True)	容許/不容許使用局部變數
38	$allowLocalConstants	True | False (default: True)	容許/不容許使用局部常數
39	$allowThreadLocalVariables	True | False (default: True)	容許/不容許使用局部變數的執行緒
40	$allowInlineAssembly	True | False (default: True)	經內聯展開的組合語言可否適用於作用域
41	$coldScope	True | False (default: False)	有無對最佳化器通知作用域的執行頻度低
42	$secureScope	True | False	設定為True係意指以下 將$allowInputToOutput設定為False 將$clearMemoryOnExit設定為True 將$allowConditionalBranches設定為False 將$enableRuntimeSafety設定為True 將$allowUnreachablePaths設定為False 將$allowUndefinedVariables設定為False 將$allowPanics設定為False 將$allowFunctionRecursion設定為False 將$floatMode設定為Strict 將$allowErrorTermination設定為False

【0063】 典型而言，如上所述之制約係可包含：對目的碼116執行時所利

用的資源的限制或規則、對目的碼116的執行狀態的限制或規則、對目的碼116的執行順序的限制或規則、以及對原始碼112所包含的命令的限制或規則等。

【0064】 其中，亦可不需要構裝上表所示之所有制約碼，而按照所被要求的規格，僅構裝其一部分。此外，亦可採用上表所示之制約碼以外的制約碼。

【0065】 ＜E. 評估適合於制約的構造＞

接著，說明評估按照本實施形態之軟體開發裝置100適合於制約的構造之一例。

【0066】 圖7係顯示按照本實施形態之軟體開發裝置100所提供的功能構成的區塊圖。典型而言，圖7所示之各功能係藉由軟體開發裝置100的處理器102執行軟體開發程式114來實現。

【0067】 參照圖7，軟體開發程式114係接受原始碼112的輸入，而生成目的碼116（組合程式碼）。更具體而言，軟體開發程式114係包含：前置處理器1141、編譯器1142、最佳化器1143、及碼產生器1144。

【0068】 前置處理器1141係執行對原始碼112的詞句解析及句法解析，並且控制編譯器1142、最佳化器1143、及碼產生器1144的行動。

【0069】 編譯器1142係根據對原始碼112的詞句解析及句法解析的結果，生成目的碼。最佳化器1143係將所生成的目的碼最適化。碼產生器1144係根據藉由最佳化器1143所得之最適化的結果，輸出最終的目的碼116。

【0070】 評估上述適合於制約時，前置處理器1141、編譯器1142、及最佳化器1143係抽出原始碼112所規定的制約，並且評估所抽出的適合於制約（步驟S1）。其中，亦可按照所設定的制約的內容，最佳化器1143修正目的碼。如上所示，軟體開發裝置100係抽出原始碼112所設定的制約，在該所抽出的制約的作用域中，評估該原始碼112是否適合於該制約。

【0071】 碼產生器1144係若非為目的碼116執行時即無法評估所抽出的適

合於制約時，生成用以評估適合於制約的組合程式碼亦即適合評估用目的碼118（步驟S2）。

【0072】 如上所示，軟體開發裝置100係以適合於制約的方式生成目的碼116。

【0073】 圖8係用以說明實現在按照本實施形態之軟體開發裝置100中對於對叫出的制約的適合的方法的圖。圖8所示之處理亦可藉由軟體開發程式114所提供，亦可在執行目的碼116時藉由適合評估用目的碼118來提供。

【0074】 參照圖8，生成表示叫出源的程序所設定的制約作用域（以下亦稱為「父作用域」）的資料集150。父作用域的資料集150係包含變數、制約、程序等資訊。與父作用域的資料集150建立對應，而亦生成管理物件152。

【0075】 表示叫出目的端的程序應繼承的制約作用域（以下亦稱為「子作用域」）的資料集154係與父作用域的資料集150建立關連。與子作用域的資料集154建立對應而亦生成管理物件156。

【0076】 例如，若程序或函數被叫出，參照與父作用域的資料集150建立關連的管理物件152（步驟S11）。接著，根據父作用域的資料集150，生成子作用域的資料集154（步驟S12）。響應由子作用域的資料集154對管理物件152的參照（步驟S13），由管理物件152生成管理物件156（步驟S14）。所生成的管理物件156係與子作用域的資料集154建立關連（步驟S15）。

【0077】 藉由反覆如上所示之一連串處理，程序或函數按照叫出，而繼承所設定的制約的作用域。

【0078】 ＜F.處理順序＞

接著，說明按照本實施形態之軟體開發裝置100中由原始碼生成目的碼的處理順序。

【0079】 圖9係顯示按照本實施形態之軟體開發裝置100中由原始碼生成

目的碼的處理順序的流程圖。典型而言，圖9所示之各步驟係藉由處理器102執行軟體開發程式114來實現。

【0080】 參照圖9，軟體開發裝置100係對所被輸入的原始碼112執行詞句解析及句法解析（步驟S100）。軟體開發裝置100係根據解析結果，判斷是否設定制約（步驟S102）。亦即，軟體開發裝置100係抽出原始碼112所設定的制約，在該所抽出的制約的作用域中，評估原始碼112是否適合於該制約。在此，若未設定制約（步驟S102中為NO），途中的步驟被跳過，而執行步驟S118以下的處理。

【0081】 若設定制約（步驟S102中為YES），軟體開發裝置100係在所設定的制約之中選擇1個（步驟S104），判斷所選擇出的制約的作用域所包含的原始碼是否適合於該制約（步驟S106）。

【0082】 所被選擇出的制約的作用域所包含的原始碼之中若有不適合於該制約的部分（步驟S106中為NO），軟體開發裝置100係輸出不適合於制約之要旨的訊息（步驟S108），且中斷生成目的碼116的處理（步驟S110）。接著，處理即結束。如上所示，軟體開發裝置100係若在制約的作用域中，評估出原始碼112不適合於該制約時，即中止目的碼116的生成。

【0083】 圖10係顯示按照本實施形態之軟體開發裝置100所提供的使用者介面畫面之一例的圖。參照圖10，使用者介面畫面300係包含：用以作成原始碼的編輯區域302；用以開始在編輯區域302中所作成的原始碼的編譯的編譯按鍵304；及用以顯示錯誤訊息的訊息顯示區域306。不適合於制約之要旨的訊息亦可顯示在訊息顯示區域306。

【0084】 再度參照圖9，若無法判斷所被選擇出的制約的作用域所包含的原始碼是否適合於該制約（步驟S106中為「不明」），軟體開發裝置100係標記無法判斷原始碼是否適合於該制約的部分（步驟S112）。接著，處理係進至步

驟S114。

【0085】 若所被選擇出的制約的作用域所包含的所有原始碼適合於該制約（步驟S106中為YES），軟體開發裝置100係判斷關於所設定的所有制約的評估是否已完成（步驟S114）。所設定的制約之中，若有剩下評估未完成者（步驟S114中為NO），軟體開發裝置100係在未評估的制約之中選擇1個（步驟S116），且執行步驟S106以下的處理。

【0086】 若關於所設定的所有制約的評估完成（步驟S114中為YES），軟體開發裝置100係生成目的碼116（步驟S118）。

【0087】 接著，軟體開發裝置100係判斷無法判斷原始碼是否適合於該制約的部分是否被標記（步驟S120）。亦即，在上述步驟S112中，判斷是否任何部分被標記。

【0088】 若無法判斷原始碼是否適合於該制約的部分未被標記（步驟S120中為NO），步驟S122的處理係被跳過。相對於此，若無法判斷原始碼是否適合於該制約的部分被標記（步驟S120中為YES），軟體開發裝置100係生成用以判斷被標記的部分是否適合於執行時所被設定的制約的適合評估用目的碼118（步驟S122）。如上所示，軟體開發裝置100係在制約的作用域中，若無法評估原始碼112是否適合於制約時，除了對應原始碼112的目的碼116之外，在目的碼116執行中，生成用以評估是否適合於該制約的其他目的碼（適合評估用目的碼118）。

【0089】 最終，軟體開發裝置100係輸出所生成的目的碼（步驟S124）。亦即，軟體開發裝置100係以適合於原始碼112所包含的制約的方式生成目的碼。接著，處理即結束。

【0090】 ＜G.變形例＞

在上述說明中，為方便說明，係例示設定1個制約的情形，惟亦可重複設定

複數制約而非侷限於此。此外，亦可使各個作用域彼此部分重複來設定複數制約。

【0091】 此外，在上述說明中，係例示在原始碼112埋入制約碼的構成例，惟並非侷限於此，亦可有別於原始碼112而準備規定制約的定義檔案。此時，在定義檔案，亦可建立關連地包含特定成為制約的作用域的程序名或函數名的資訊、與所適用的制約的內容。

【0092】 如上所示，關於制約的設定方法，可採用任意方法。

＜H.優點＞

藉由按照本實施形態之軟體開發裝置100，對在邊緣元件等所被執行的程式，可提供可自由設定各種制約的環境。藉此，由可利用的資源及安全性的觀點來看，可考慮到各種情形之後，作成在邊緣元件所執行的程式。

【0093】 應理解本次所揭示的實施形態全部均為例示，並非為具限制性者。本發明之範圍並非藉由上述說明而是藉由申請專利範圍所示，意圖包含在與申請專利範圍為均等的涵義及範圍內的所有變更。

【符號說明】

【0094】

1:IoT系統

2:邊緣元件

100:軟體開發裝置

102,202:處理器

104,204:主記憶體

106:輸入部

108:顯示器

110:硬碟

112:原始碼

114:軟體開發程式

116:目的碼

118:適合評估用目的碼

120:內部匯流排

122:通訊介面

150,154:資料集

152,156:管理物件

200:控制器

206:快閃記憶體

210:運算處理部

212:無線通訊模組

214:USB控制器

216:通訊控制器

218:驅動器

220:墊片

300:使用者介面畫面

302:編輯區域

304:編譯按鍵

306:訊息顯示區域

1121,1127:定義

1123,1124,1129,1130:命令

1125:制約碼

1126:部分原始碼

1128:處理

1141:前置處理器

1142:編譯器

1143:最佳化器

1144:碼產生器

【發明申請專利範圍】

【請求項1】 一種軟體開發裝置，其係執行以下處理：

抽出制約碼的處理，前述制約碼指定包含於原始碼的制約(restriction)的內容；

決定第一適用範圍的處理，前述第一適用範圍係前述原始碼中的適用由前述制約碼所指定的前述制約的範圍；

決定第二適用範圍的處理，前述第二適用範圍係前述原始碼中的與前述第一適用範圍建立關連的範圍；

管理處理，係管理適用於前述第一適用範圍的制約以及適用於前述第二適用範圍的制約；以及

評估處理，係評估前述第一適用範圍和前述第二適用範圍的前述原始碼是否符合由前述制約碼指定的制約；

其中，前述制約係包含：對目的碼執行時所利用的資源的限制或規則、對前述目的碼的執行狀態的限制或規則、對前述目的碼的執行順序的限制或規則、以及對前述原始碼所包含的命令的限制或規則之中任1個。

【請求項2】 如請求項1之軟體開發裝置，其中係執行由前述原始碼生成前述目的碼的處理。

【請求項3】 如請求項2之軟體開發裝置，其中若評估前述原始碼不適合於前述制約，即中止前述目的碼的生成。

【請求項4】 如請求項1至3中任一項之軟體開發裝置，其中若前述第一適用範圍的該原始碼包含叫出命令，則與藉由該叫出命令所叫出的命令對應的範圍被決定為前述第二適用範圍。

【請求項5】 一種軟體開發程式，其係使電腦執行：

抽出制約碼的步驟，前述制約碼指定包含於原始碼的制約的內容；

決定第一適用範圍的步驟，前述第一適用範圍係前述原始碼中的適用由前述制約碼所指定的前述制約的範圍；

決定第二適用範圍的步驟，前述第二適用範圍係前述原始碼中的與前述第一適用範圍建立關連的範圍；以及

管理步驟，係管理適用於前述第一適用範圍的前述制約以及適用於前述第二適用範圍的前述制約；

其中，前述軟體開發程式執行評估處理，其係評估前述第一適用範圍和前述第二適用範圍的前述原始碼是否符合由前述制約碼指定的前述制約；

其中，前述制約係包含：對目的碼執行時所利用的資源的限制或規則、對前述目的碼的執行狀態的限制或規則、對前述目的碼的執行順序的限制或規則、以及對前述原始碼所包含的命令的限制或規則之中任1個。

【發明圖式】

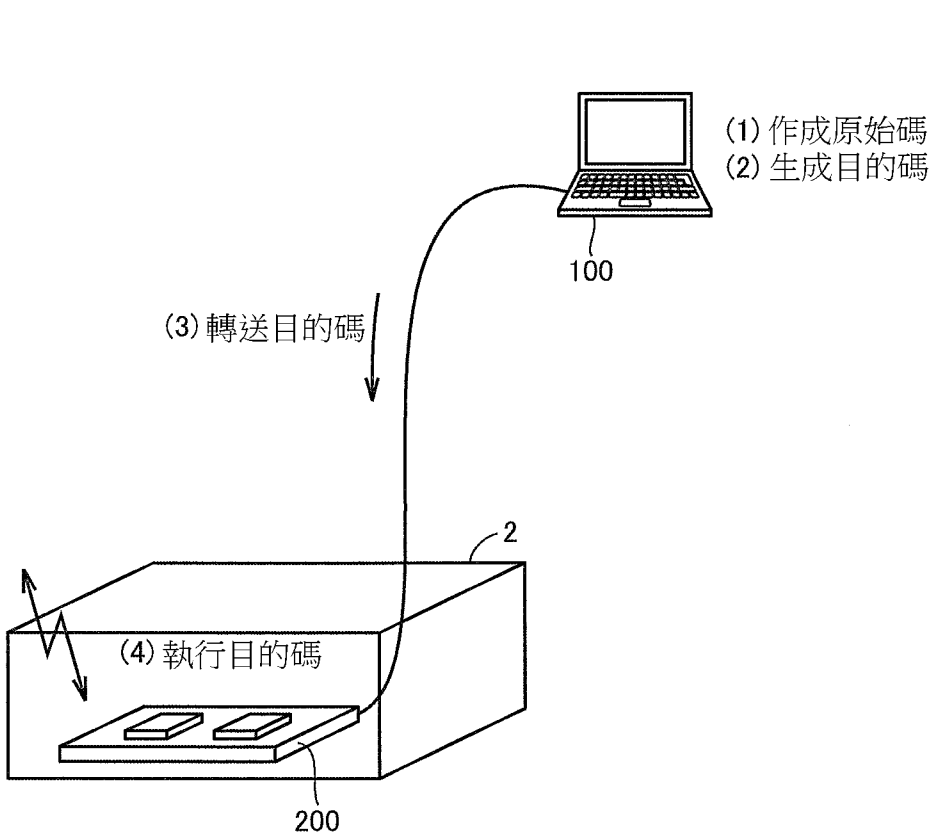


圖1

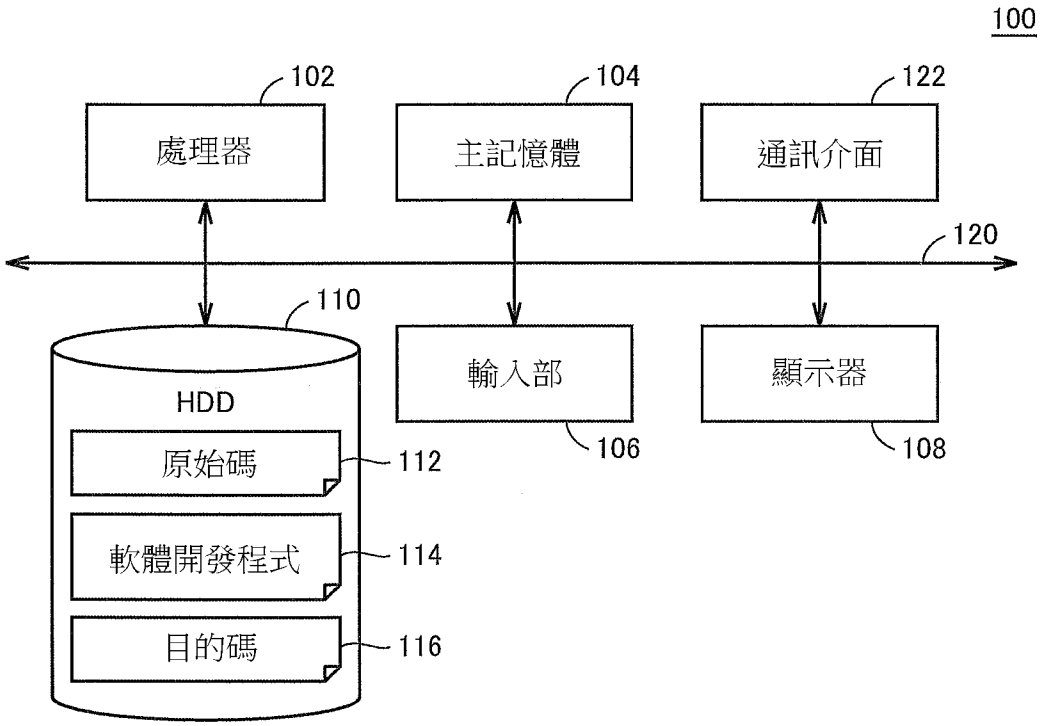

100
102
處理器
104
主記憶體
122
通訊介面
120
110
HDD
原始碼
112
軟體開發程式
114
目的碼
116
輸入部
106
顯示器
108

圖2

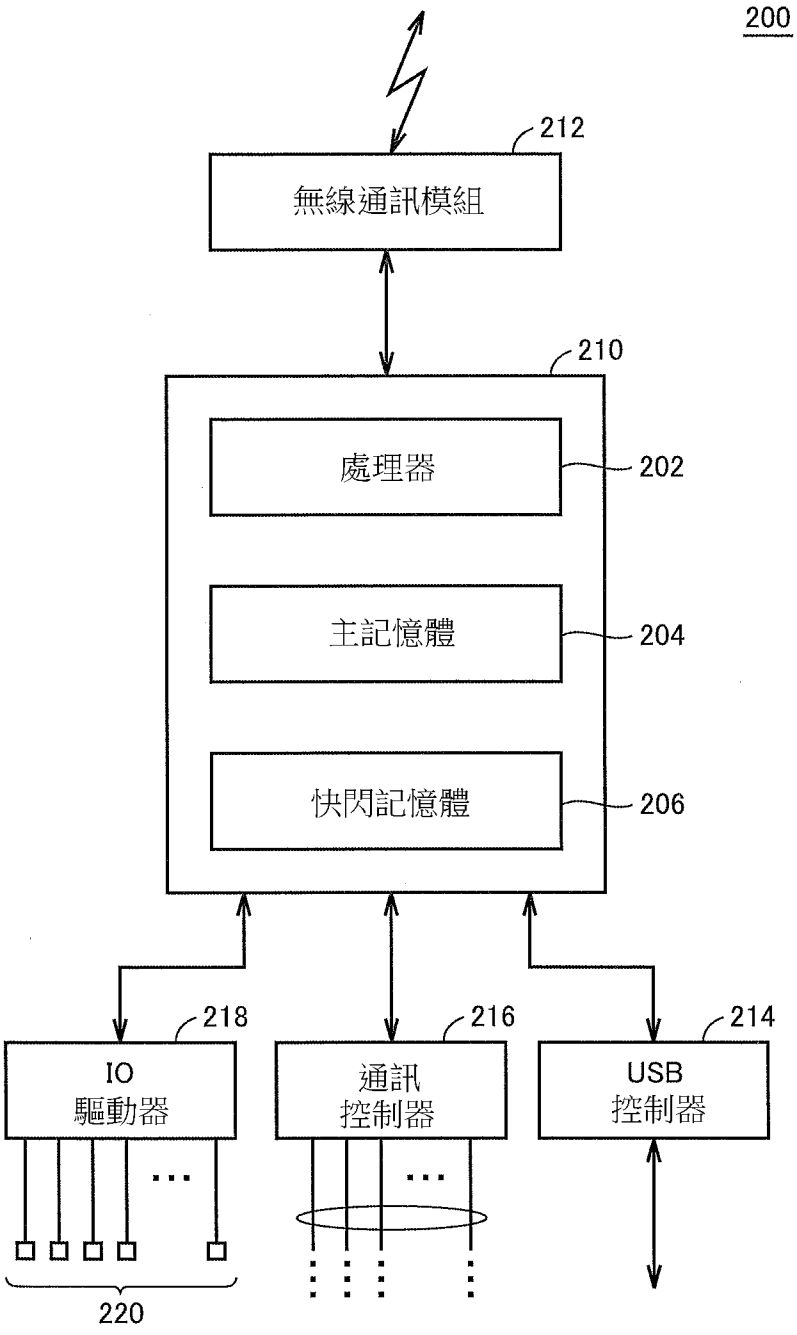

200
212
無線通訊模組
210
處理器
202
主記憶體
204
快閃記憶體
206
218
216
214
IO
驅動器
通訊
控制器
USB
控制器
220

圖3

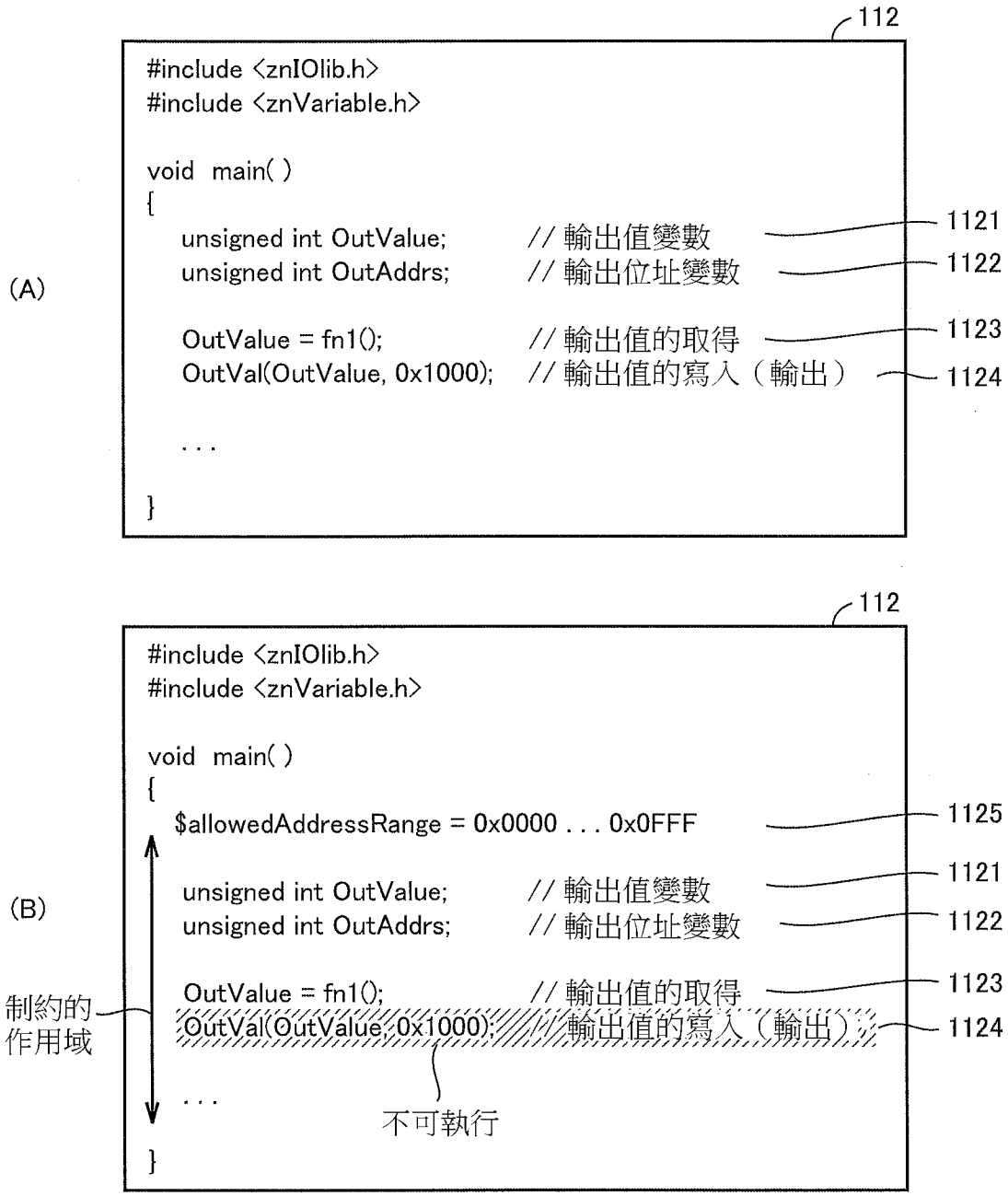

112
#include <znIOlib.h>
#include <znVariable.h>
void main( )
{
unsigned int OutValue; // 輸出值變數
unsigned int OutAddrs; // 輸出位址變數
OutValue = fn1(); // 輸出值的取得
OutVal(OutValue, 0x1000); // 輸出值的寫入（輸出）
. . .
}
1121
1122
1123
1124
(A)
112
#include <znIOlib.h>
#include <znVariable.h>
void main( )
{
$allowedAddressRange = 0x0000 . . . 0x0FFF
unsigned int OutValue; // 輸出值變數
unsigned int OutAddrs; // 輸出位址變數
OutValue = fn1(); // 輸出值的取得
OutVal(OutValue, 0x1000); // 輸出值的寫入（輸出）
. . .
}
1125
1121
1122
1123
1124
(B)
制約的
作用域
不可執行

圖4

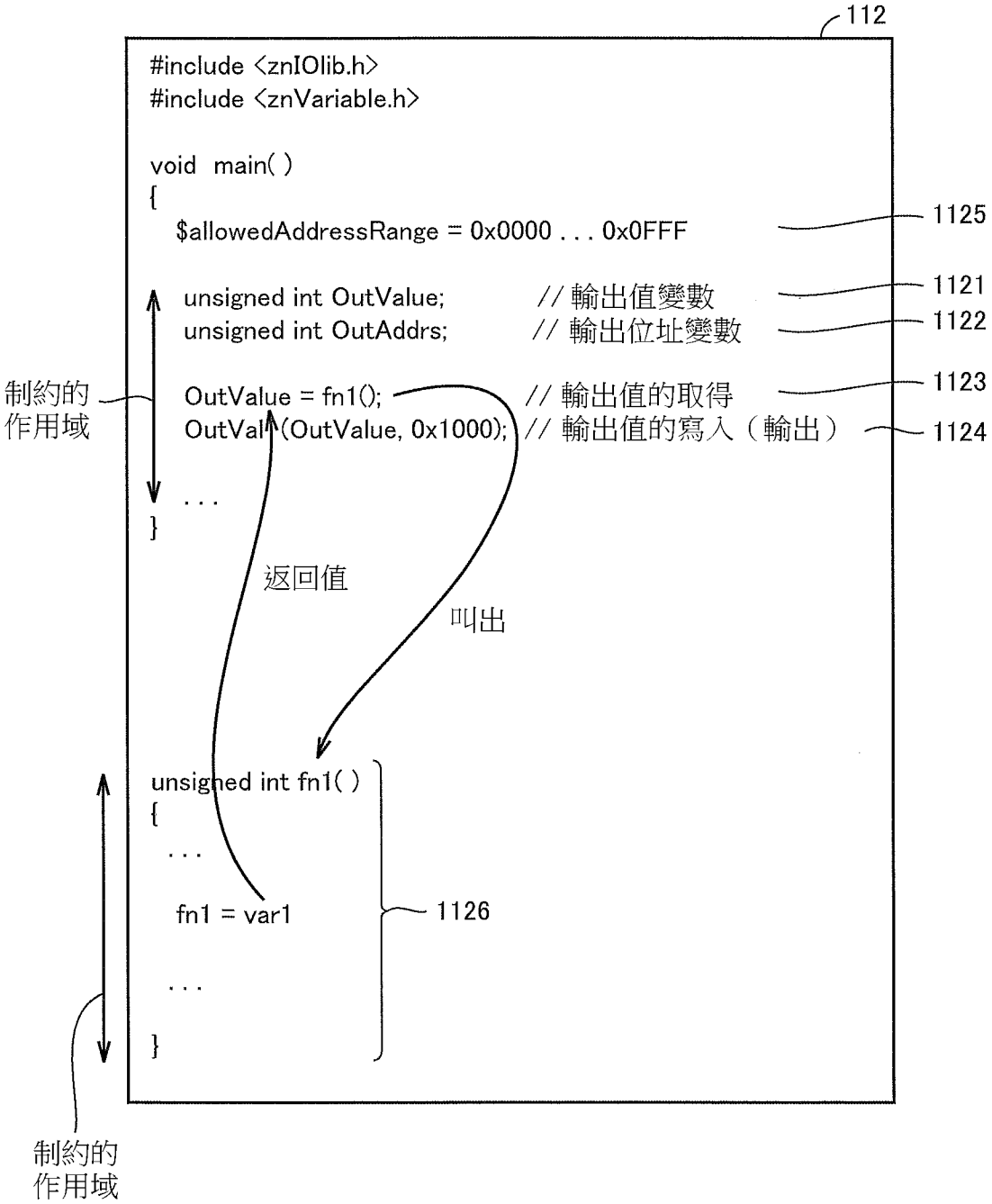

112
#include <znIOlib.h>
#include <znVariable.h>
void main( )
{
$allowedAddressRange = 0x0000 . . . 0x0FFF
1125
unsigned int OutValue; // 輸出值變數
1121
unsigned int OutAddrs; // 輸出位址變數
1122
制約的作用域
OutValue = fn1(); // 輸出值的取得
1123
OutVal(OutValue, 0x1000); // 輸出值的寫入（輸出）
1124
. . .
}
返回值
叫出
unsigned int fn1( )
{
. . .
fn1 = var1
1126
. . .
}
制約的作用域

圖5

(A)

112

```
#include <znIOlib.h>
#include <znVariable.h>

void  main( )
{
  $allowedAddressRange = 0x0000 . . . 0x0FFF                    1125

   unsigned int OutValue;        // 輸出值變數                  1121
   unsigned int OutAddrs;        // 輸出位址變數                1122

   OutValue = fn1();             // 輸出值的取得                1123
   OutVal(OutValue, 0x0100);     // 輸出值的寫入（輸出）        1124

   . . .

}
```

(B)

112

```
#include <znIOlib.h>
#include <znVariable.h>

void  main( )
{
  $allowedAddressRange = 0x0000 . . . 0x0FFF                    1125

   unsigned int OutValue;        // 輸出值變數                  1121
   unsigned int OutAddrs;        // 輸出位址變數                1122
   unsigned int IniPtr;          // 輸出位址初期值變數          1127

   OutValue = fn1();             // 輸出值的取得                1123

   . . . //決定輸出位址初期值 IniPtr 的處理                    1128

   OutAddrs = IniPtr + 0F00;     // 輸出位址的決定              1129

   OutVal(OutValue, OutAddrs); // 輸出值的寫入（輸出）          1130

   . . .

}
```

圖6

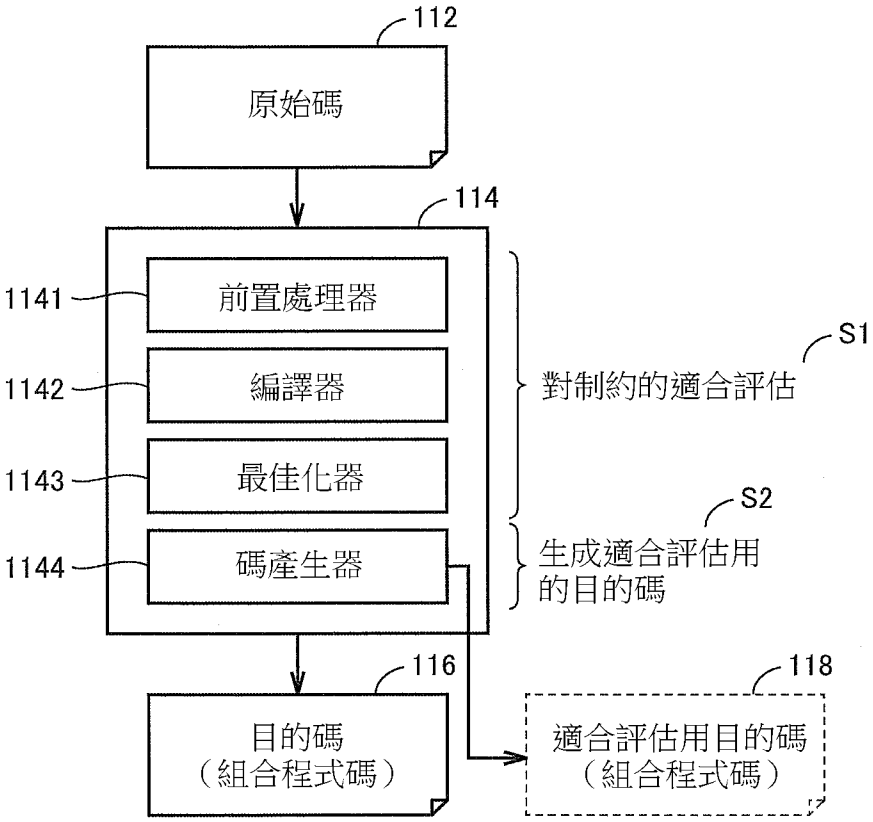

112
原始碼
114
1141
前置處理器
1142
編譯器
1143
最佳化器
1144
碼產生器
S1
對制約的適合評估
S2
生成適合評估用
的目的碼
116
目的碼
（組合程式碼）
118
適合評估用目的碼
（組合程式碼）

圖7

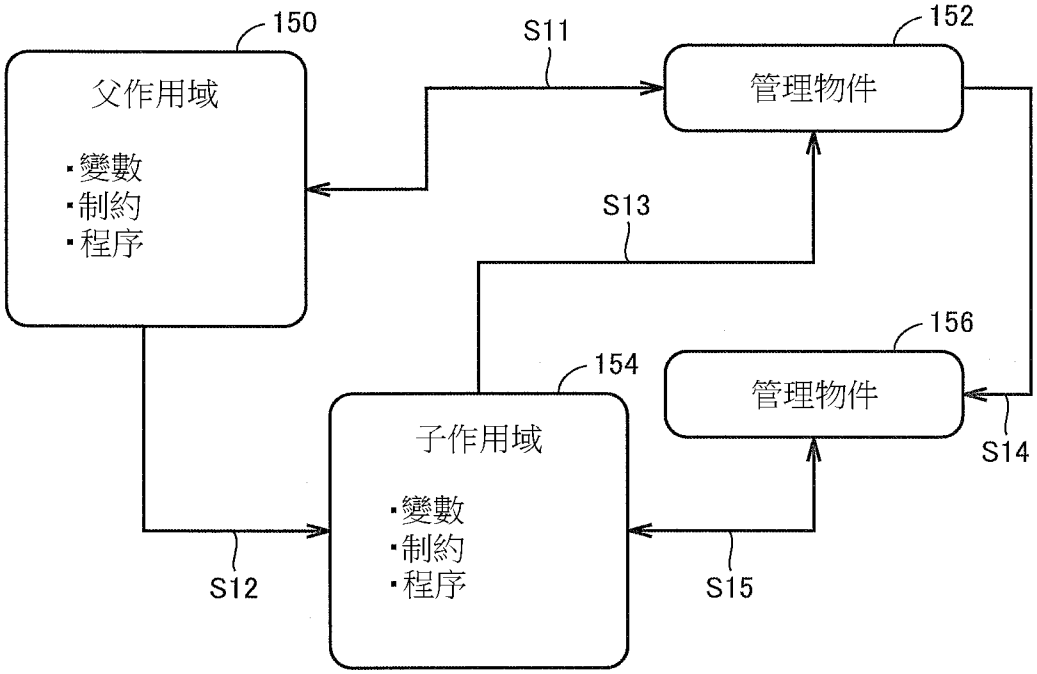

150
父作用域
▪變數
▪制約
▪程序
S11
152
管理物件
S13
156
管理物件
S14
154
子作用域
▪變數
▪制約
▪程序
S12
S15

圖8

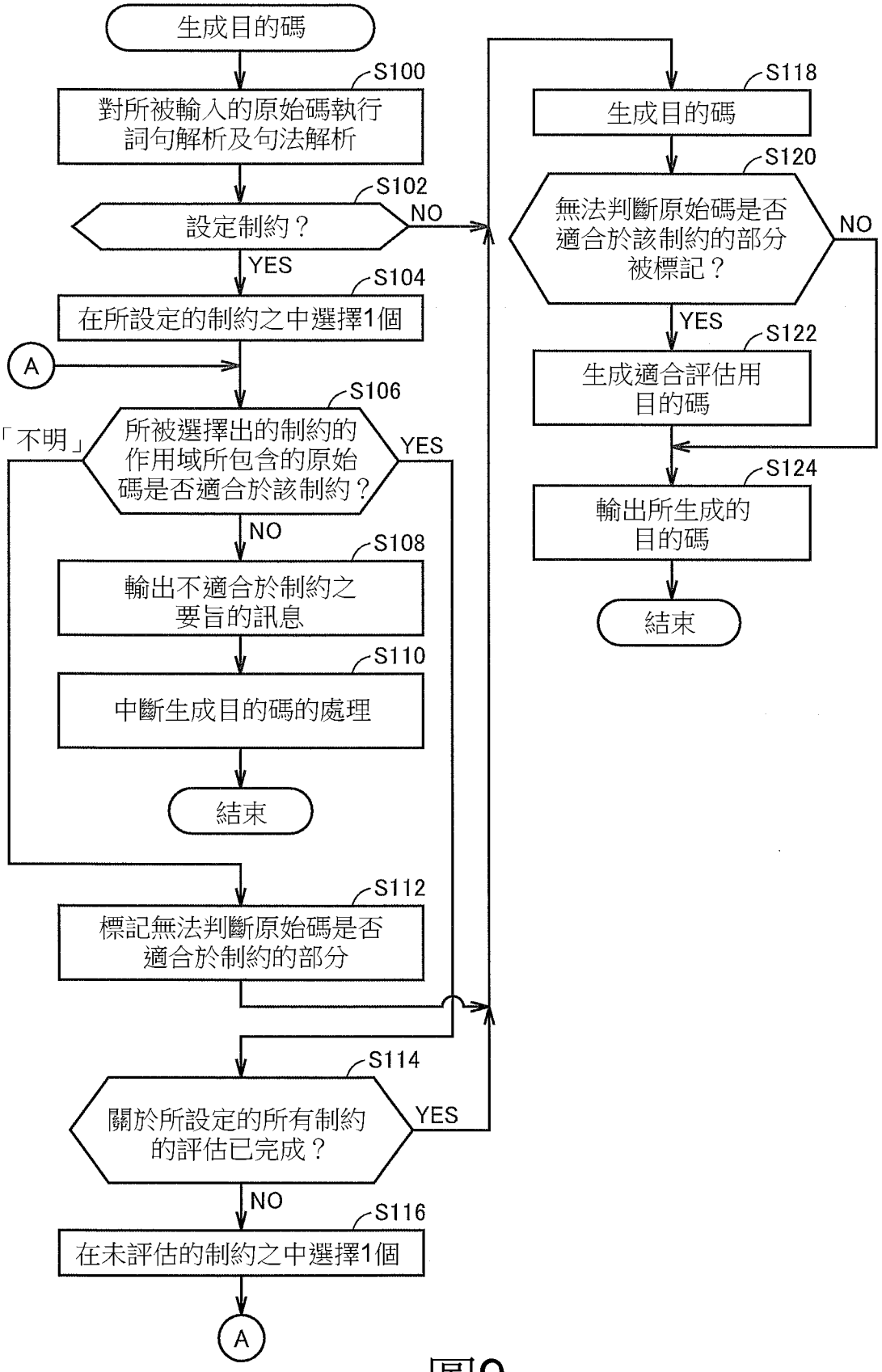

生成目的碼
S100
對所被輸入的原始碼執行詞句解析及句法解析
S102
設定制約？
NO
YES
S104
在所設定的制約之中選擇1個
A
S106
「不明」
所被選擇出的制約的作用域所包含的原始碼是否適合於該制約？
YES
NO
S108
輸出不適合於制約之要旨的訊息
S110
中斷生成目的碼的處理
結束
S112
標記無法判斷原始碼是否適合於制約的部分
S114
關於所設定的所有制約的評估已完成？
YES
NO
S116
在未評估的制約之中選擇1個
A
S118
生成目的碼
S120
無法判斷原始碼是否適合於該制約的部分被標記？
NO
YES
S122
生成適合評估用目的碼
S124
輸出所生成的目的碼
結束

圖9

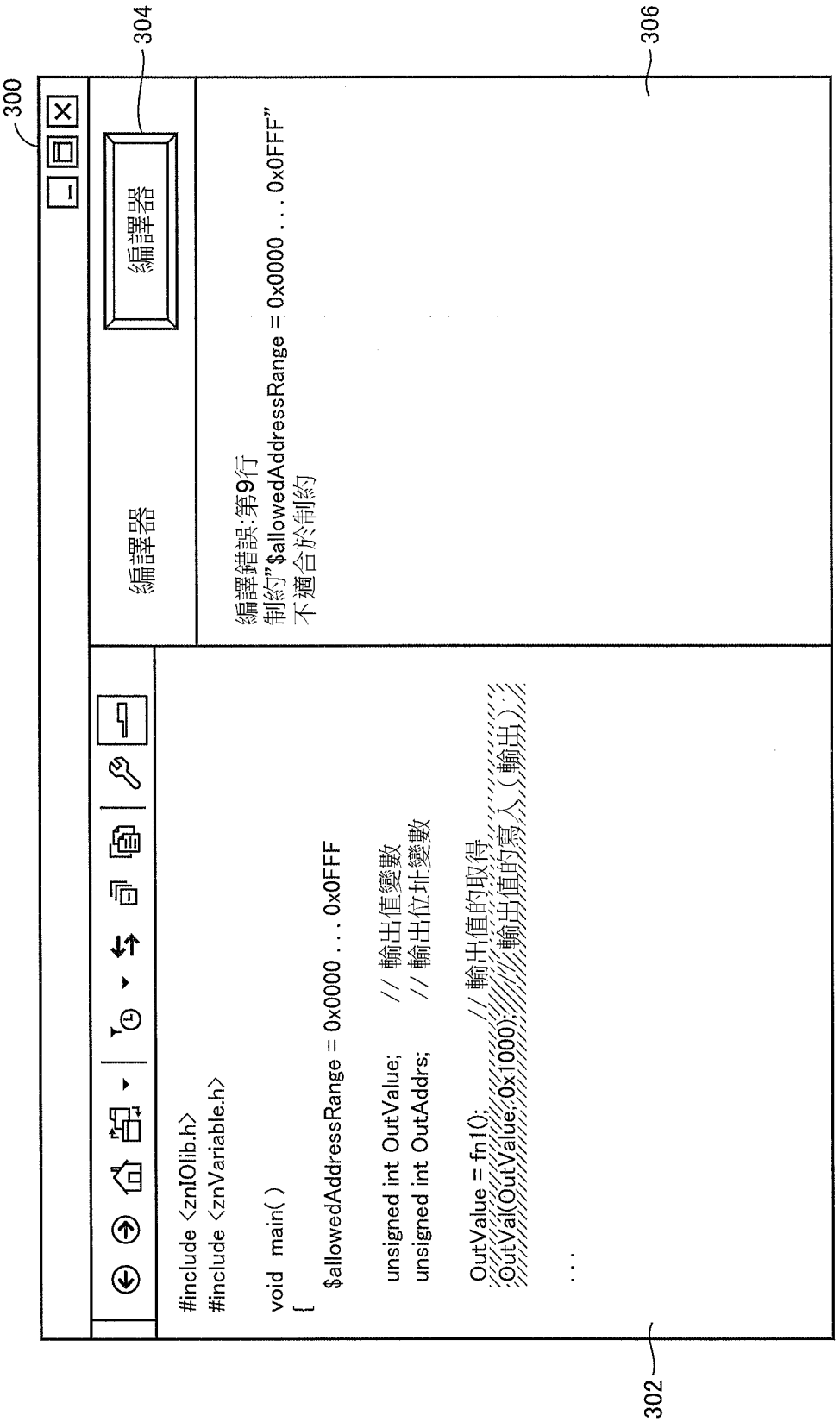

300
304
306
302
編譯器
編譯器
編譯錯誤:第9行
制約"$allowedAddressRange = 0x0000 . . . 0x0FFF"
不適合於制約
#include <znIOlib.h>
#include <znVariable.h>
void main( )
{
$allowedAddressRange = 0x0000 . . . 0x0FFF
unsigned int OutValue; // 輸出值變數
unsigned int OutAddrs; // 輸出位址變數
OutValue = fn1();
. . .

圖10