PCS 3216
Sistemas de Programação

Aula 11

Programação Simbólica Relocável

INTRODUÇÃO

Programas relocáveis

- Programas absolutos carecem de **modularidade** pois estão intrinsecamente vinculados a endereços fixos.
- A programação absoluta dificulta também a criação de **bibliotecas**, mas isto se pode atenuar promovendo-se a criação de **módulos** que possam ser montados em separado, de forma independente.
- Essa característica de modularidade também pode ser estendida para a programação em linguagem simbólica, por meio da incorporação, à linguagem simbólica, do conceito de **relocabilidade**, dando ao programador a possibilidade da criação de módulos relativamente **autônomos**, isentos de vínculos a endereços fixos.

Referências simbólicas entre módulos

- Os módulos assim concebidos podem **referenciar-se** uns aos outros **simbolicamente**.
- Operações de **ligação entre módulos** que se referenciam permitem criar **módulos compostos** a partir dos existentes, visando à obtenção, em um só módulo, de um programa relocável completo.
- Uma **operação adicional de relocação** (i.é., ajuste adequado dos endereços relativos internos aos módulos) permite **transformar esses programas, relocáveis,** em **programas absolutos, executáveis**.

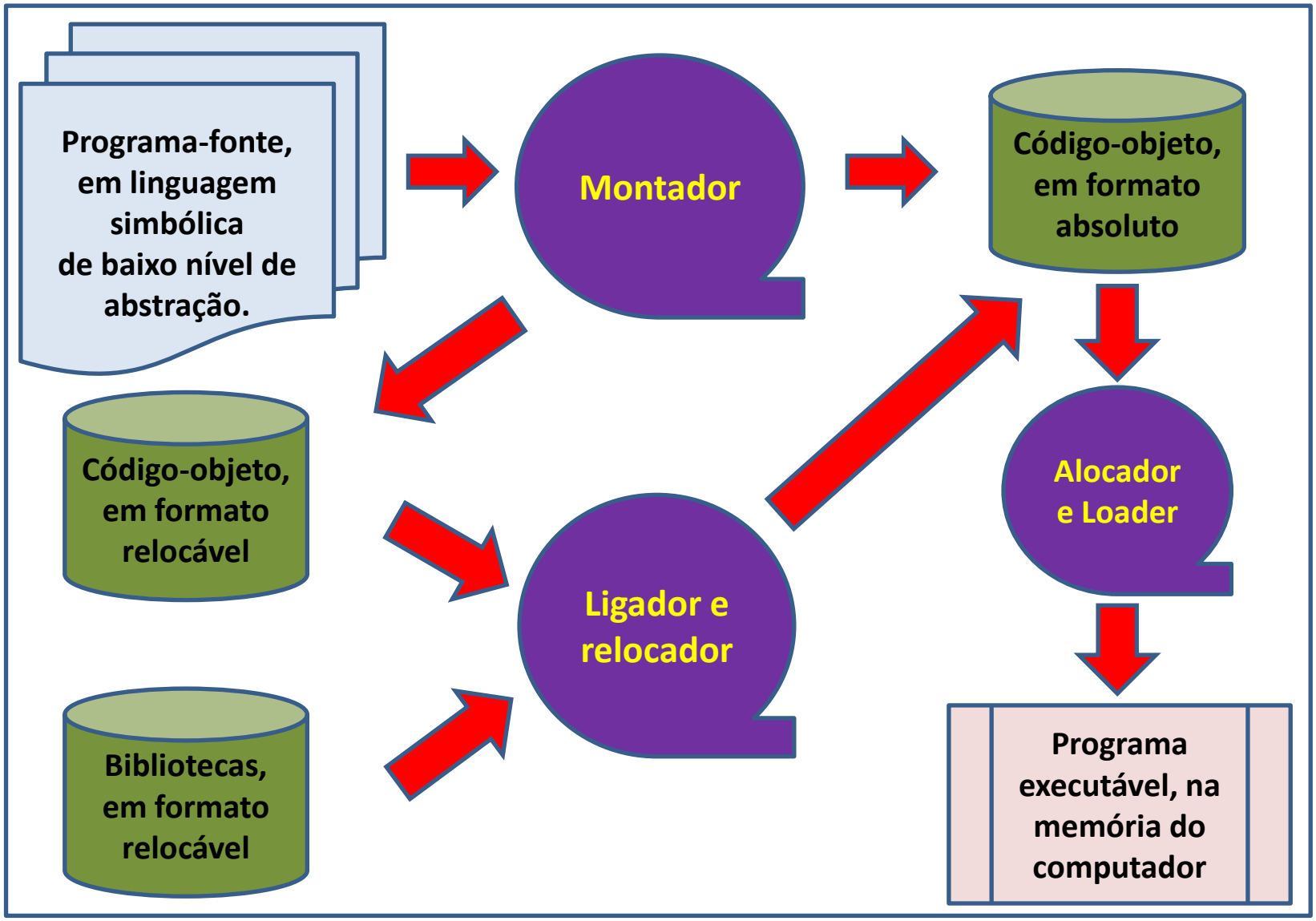
Linguagens simbólicas e módulos relocáveis

- As chamadas ***linguagens simbólicas relocáveis*** viabilizam o desenvolvimento de programas modulares em nível baixo de abstração.
- **Módulos relocáveis** costumam formar programas incompletos, e por isso, antes da execução, necessitam ligar-se aos demais módulos de que dependem, para que **as suas referências simbólicas** mútuas possam coexistir no mesmo módulo, para então ser **resolvidas**.
- Consideram-se totalmente **resolvidas essas referências** quando estiverem determinados os endereços (absolutos) de memória, associados a todas elas.

Ligadores, Relocadores, Alocadores

- Uma breve menção em conjunto aos softwares mais importantes de um Sistema de Programação:
  - **Montadores** geram código relocável, em módulos (partes) que se referenciam (e complementam) mutuamente.
  - **Ligadores** são os softwares de sistema que garantem que estejam presentes todos os módulos referenciados, integrando-os, e compondo assim o programa completo.
  - **Relocadores** são softwares de sistema que adéquam os endereços relativos internos de cada módulo à região específica de memória em que o módulo deverá ser executado.
  - **Alocadores** são softwares de sistema que em geral fazem parte do sistema operacional, e que determinam a região de memória disponível mais adequada para a execução dos programas do usuário.
  - **Loaders** são softwares de sistema encarregados de promover a execução imediata de um programa do usuário ou do sistema.

Preparação de programas em linguagem de baixo nível



LIGAÇÃO E RELOCAÇÃO

Ligação e relocação

- Assim, o programa só ficará pronto para a execução quando todas as suas referências relocáveis à memória tiverem sido convertidas em endereços absolutos.
- Isto é feito na época da **relocação dos endereços relativos** presentes no código do programa.

- A época em que são determinados os endereços associados às referências simbólicas entre módulos é chamada **época de ligação**, ou de ***resolução de referências simbólicas*** (= *linking, binding*).

Tratamento de endereços relativos

- Em um **programa relocável:**
  - Os **endereçamentos** efetuados **internamente** a um módulo costumam ser majoritariamente **relativos.**
  - Um módulo relocável é usualmente montado **a partir de um endereço relativo zero**, que fica subentendido.
  - Uma vez decidido alocar um módulo relocável a partir de um endereço absoluto arbitrário X, todos os endereços internos (relativos) devem ser então **relocados**, ou seja, **deslocados na memória para X posições adiante de seu antigo endereço relocável.**

Pseudo-Instruções e código-objeto

- Na linguagem simbólica, programas absolutos diferem dos relocáveis quanto às **pseudo-instruções** disponibilizadas ao programador.
- Nos programas relocáveis, as pseudo-instruções se destinam a manipular elementos da linguagem que envolvam conceitos ausentes em programas absolutos.
- O código objeto **relocável** também difere em estrutura do código absoluto, pois contém **meta-dados** distintos.
- Particularmente significativos são os meta-dados que caracterizam **referências simbólicas** responsáveis pelo estabelecimento de **conexões entre módulos**.

Montador e bibliotecas

- Para a elaboração de programas usando a opção relocável, a principal ferramenta é o **montador**, com o qual há a possibilidade de montagem de programas-objeto relocáveis, os quais podem ser colecionados para formar ***bibliotecas relocáveis***.
- Muitos montadores relocáveis dão ao usuário a possibilidade de, como opção, montar também programas absolutos, já que programas absolutos podem ser vistos como sendo um caso muito particular de formato, dentre os muitos que no caso geral podem ser tratados pelo montador.

PROGRAMAÇÃO SIMBÓLICA RELOCÁVEL

Características dos programas relocáveis

- ***Programas relocáveis*** constituem módulos que não ficam atrelados a posições fixas de memória.
- A conhecida conveniência da reutilização de códigos já validados justifica e estimula o uso de ***bibliotecas*** relocáveis.
- ***Programas-objeto relocáveis*** apresentam os mesmos elementos dos correspondentes programas absolutos, exceto quanto ao emprego de ***meta-dados*** específicos e ao uso adicional de ***endereços relativos e simbólicos***.

Endereçamentos

- Códigos-objeto relocáveis típicos admitem referências a endereços de naturezas variadas:
  - **Absolutos** – para referenciar posições fixas de memória: endereços de informações do núcleo do Sistema Operacional, nas memórias compartilhadas etc.
  - **Relativos** – usados em referências internas a um programa: registram distância a algum ponto de referência.
  - **Relocáveis** – usados em programação relocável. Trata-se de um caso particular de endereçamento relativo, no qual o ponto de referência fica implícito, e corresponde ao endereço relativo zero do programa relocável.
  - **Simbólicos** – usados para permitir referências mútuas através de rótulos simbólicos, importados ou exportados entre módulos distintos.

Relocação

- Para se poder executar um programa, todos os seus endereços relativos devem estar resolvidos, ou seja, convertidos em endereços absolutos.
- Isto pode ser feito por uma ***relocação*** prévia, ou seja, pela correção de todos os endereços relativos referenciados mediante a adição da base apropriada de relocação. Por ser efetuada antes da execução do programa, esta operação é dita ***relocação estática***.
- Algumas arquiteturas realizam a ***relocação dinâmica***, ou seja, uma correção automática das referências a endereços relativos, efetuada por hardware, na ocasião da execução, dispensando assim a manipulação prévia do programa.

Endereçamento simbólico

- Outro tipo usual de referência tratada pelo montador relocável é a referente aos ***endereços simbólicos***.
- Neste caso, os módulos podem referenciar-se mutuamente ***importando*** e ***exportando*** os nomes de alguns dos seus rótulos (***externals/entry-points***).
- Para tratar a ocorrência de endereços deste tipo, é necessário que eles sejam **convertidos em endereços absolutos ou relativos** antes que possam ser processados da forma anteriormente apresentada para os demais rótulos.
- O software do sistema de programação que efetua este tratamento é o ***ligador*** (*linker, binder*).

ALGUMAS OBSERVAÇÕES

Resolução de referências

- Ao contrário dos programas absolutos, por sua natureza, os programas desenvolvidos em **linguagens simbólicas relocáveis** apresentam-se geralmente incompletos, logo não podem, como aqueles, ser executados diretamente.
- Por ser possível efetuar referências entre módulos independentemente desenvolvidos, endereços referenciáveis entre um módulo e outro apresentam-se inicialmente "***não ligados***", ou seja, não são associados a qualquer posição específica de memória, absoluta ou relativa.

Referências externas

- A vinculação, a posições físicas de memória, dos endereços não-ligados (contidos em programas relocáveis na forma de "***referências externas***" simbólicas) não é efetuada na ocasião da tradução do programa simbólico para o formato objeto.
- Ao contrário, é postergada até que se conheça a posição de memória, relativa ou absoluta, a ser ocupada por todos esses endereços, externamente referenciáveis.

Linking, *binding*

- A época em que os símbolos correspondentes às referências externas são associados aos respectivos endereços físicos (absolutos ou relativos) é denominada "***fase de resolução de referências externas***", ou "***fase de ligação entre módulos***" ("***linking***", "***binding***").
- Para que um programa-objeto relocável possa ser executado em qualquer região de memória, ele não pode depender de ser carregado em posições físicas fixas de memória, devendo por isso estar isento de referências internas absolutas, que o possam atrelar a posições específicas de memória.
- Naturalmente, há algumas exceções quando se trata de programas especiais de sistema, especialmente aqueles que são partes do sistema operacional, pois estes, por várias razões, comunicam-se através de áreas fixas de memória.

Origem virtual

- Assim sendo, convém que o endereçamento interno a um módulo relocável seja sempre relativo ao início do código desse módulo.
- Usualmente, costuma-se adotar que todo módulo relocável tenha início em uma "***posição zero***" , que representa um endereço de origem **(origem virtual)**, específico do módulo, ao qual são relativos, por convenção, todos os demais endereços internos do módulo (**endereços relocáveis**).

Base de relocação

- Dessa forma, uma vez decidido, por qualquer critério ou método, que o módulo deverá ser carregado em uma posição física X de memória, o módulo poderá ser convertido em um programa executável através de um procedimento denominado "***relocação***", que consiste em "***resolver as referências relativas***", adicionando-se a todas elas a "***base de relocação***" X.

Ferramentas para programação relocável

- Com esse conjunto de programas de sistema: **montador** (absoluto e relocável), **ligador, relocador e *loader***, fica em poder do usuário um poderoso ferramental de auxílio ao desenvolvimento de programas relocáveis, que o habilita a criar e manter **bibliotecas** de módulos independentes, que poderão servir como pontos de partida para a implementação dos seus programas.

PRINCIPAIS CONCEITOS ENVOLVIDOS

Limitações do código absoluto

- Os **programas absolutos** caracterizam-se, como foi visto, pela sua forte vinculação a **posições físicas de memória**, determinadas às vezes já na época em que o programa está sendo codificado.
- Isto cria um esquema rígido, em que a cada programa ficam associadas regiões de memória preestabelecidas.

Reuso de sub-rotinas e utilitários

- Esta característica seria perfeitamente tolerável se os programas fossem sempre tão diferentes uns dos outros que cada novo programa exigisse todo um desenvolvimento completo.
- Felizmente, apesar de as aplicações dos computadores variarem significativamente, muitas das partes dos mais diversos programas repetem-se sistematicamente
- Isso se dá com certas **sub-rotinas e utilitários** como, por exemplo, as responsáveis pela formatação de dados de entrada e saída, de cálculo de funções transcendentais, trigonométricas e muitas outras, utilizadas em programas científicos e comerciais.

Programação simbólica, caso geral

- Por outro lado, isto pode tornar relativamente **complicado** o esquema de **programação absoluta** para ser utilizado no caso geral da programação simbólica, o que motiva a criação de mecanismos de representação de programas-objeto em uma forma tal que, mediante algum processamento prévio, seja possível criar facilmente versões absolutas do mesmo, vinculadas à posição de memória que se desejar.

Linguagem relocável

- Nessas representações, os programas se apresentam na forma denominada ***linguagem de máquina relocável*** ou ***linguagem-objeto relocável*** ou simplesmente ***linguagem relocável***.
- Esta forma de representação do programa contém todas as informações exibidas nos programas em forma absoluta, exceto no que tange à origem (posição física) associada ao programa.

Para finalizar

- Com os conceitos apresentados, é possível construir a parte do Sistema de Programação necessária à preparação de programas para a execução, partindo de códigos fonte simbólicos relocáveis, bibliotecas relocáveis e códigos objeto em formato relocável.
- Isso flexibiliza muito o uso de linguagens de baixo nível pois permite a prática de programação modular e outras técnicas desejáveis de engenharia de software.
- Na sequência, são apresentados exemplos práticos desenvolvidos passo a passo, mostrando detalhadamente a preparação, bem como o processamento por parte do Sistema de Programação, de programas relocáveis produzidos em linguagem simbólica.

DETALHAMENTO DOS VÁRIOS TIPOS DE ENDEREÇAMENTO

Resolução de endereços em programas simbólicos relocáveis

- Na sequência, detalha-se um pouco mais o tratamento que deve ser dado pelo montador para a **resolução** de cada uma das formas de **endereçamento** tipicamente encontradas nos programas relocáveis: **absoluto, relocável, relativo e simbólico**.

Endereçamento absoluto

- O primeiro tipo de endereçamento que pode encontrado em programas-objeto relocáveis é o já conhecido endereçamento ***absoluto***.
- Mesmo em programas não absolutos, este tipo de endereçamento é, muitas vezes, necessário, já que alguns programas necessitam referenciar posições físicas específicas e invariáveis de memória, independente da região de memória em que o programa deverá ser executado.

Usos do endereçamento absoluto

- Isto ocorre nos casos em que o programa deve **acessar regiões de comunicação** com o sistema operacional, ou, em casos mais simples, posições de memória através das quais o **hardware interage com o software**, como é o caso das posições associadas a eventos de **interrupção**.
- **Endereços absolutos** são endereços já resolvidos, portanto **dispensam tratamento de resolução**.

Endereçamento relativo

- O segundo tipo de endereçamento é o ***relativo***.
- Através de ***deslocamentos***, as instruções de referência à memória fazem referências a endereços pertencentes ao programa que se está desenvolvendo.
- Esses **deslocamentos** correspondem a **distâncias** entre algum **ponto de referência** (interno ao programa) e a posição relativa de memória assim referenciada.

Referenciais para endereçamento relativo

- Os deslocamentos são portanto **endereços relativos**, e cada Sistema de Programação geralmente convenciona alguns **pontos de referência** (por exemplo: a primeira instrução executável do programa; o início da área de *common*; o início da área de dados, o início da área de pilha, etc.), em relação aos quais são expressos, internamente ao módulo relocável, os demais endereços, internos ao programa.

Endereçamento relocável

- Os endereços relocáveis referem-se a locais cuja **posição física** permanece indefinida, e portanto, **flutuante,** até que seja escolhido o endereço de memória a partir do qual deverá residir o programa que a contém.
- Através de operações de **relocação**, tais referências relocáveis devem ser, então, convertidas em referências absolutas, para que só então o programa possa ser executado.

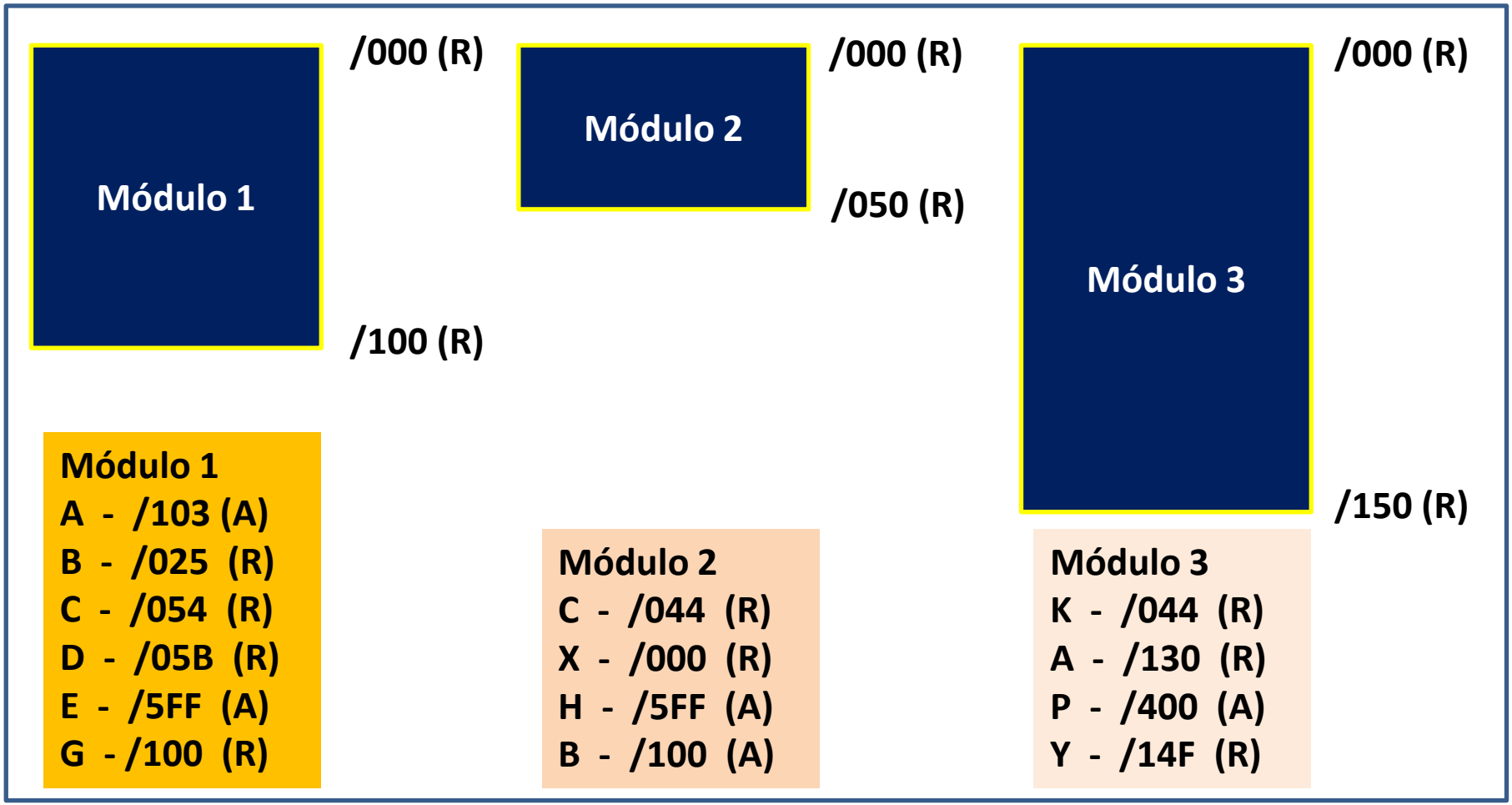
Resolução de endereços relativos

- Para ser possível executar o programa, todos os endereços relativos devem ser convertidos para endereços absolutos (operação de **resolução de endereços relativos**).
- Isto é feito através da **adição**, aos endereços relativos, de um valor, correspondente ao **endereço absoluto** associado ao **ponto de referência** a que se referem os endereços relativos em questão (**base de relocação**).

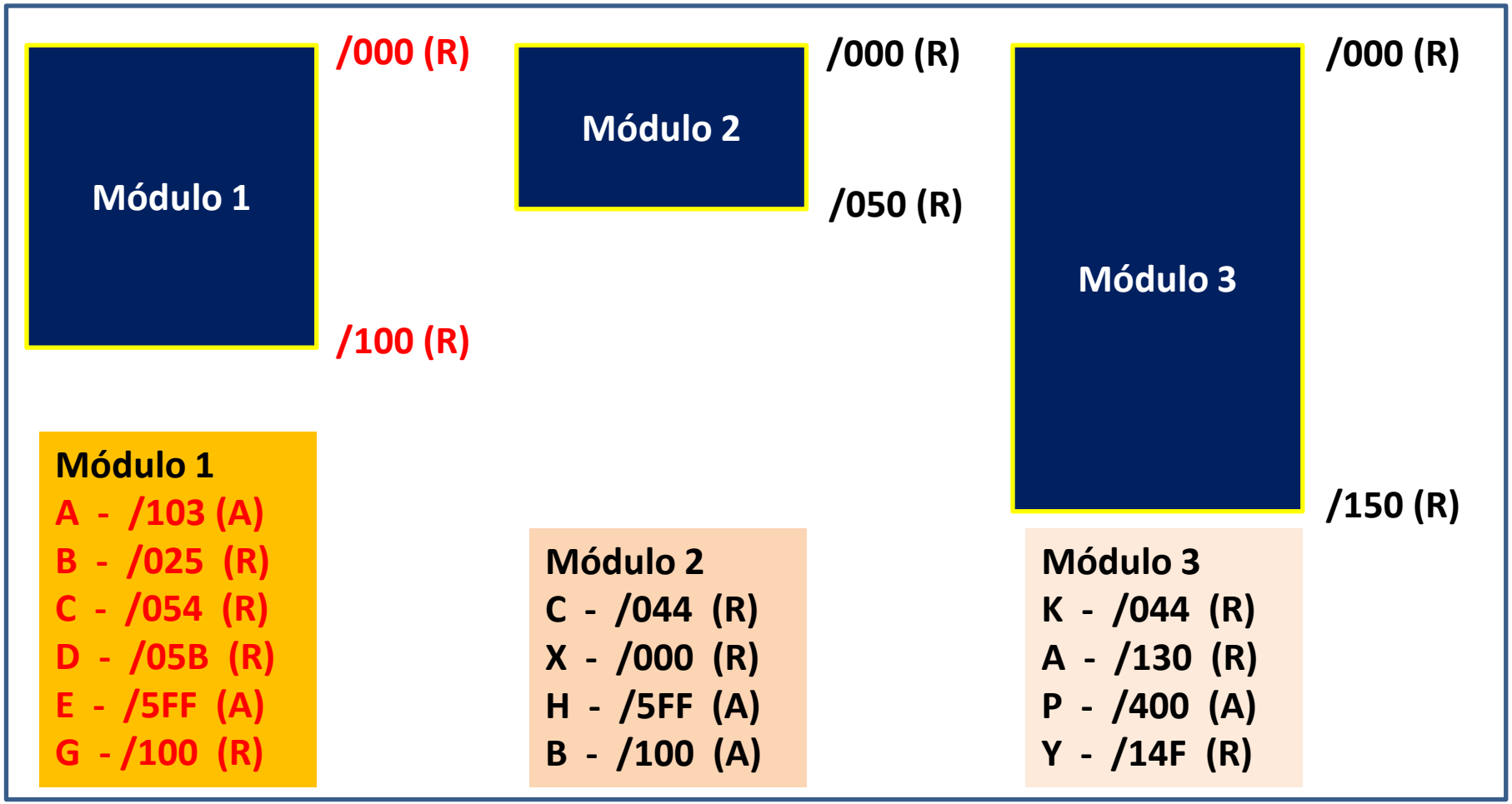
Relocações estáticas intermediárias

- Isto acarretaria um excessivo aumento no tempo de execução do programa, o qual pode ser facilmente reduzido se for efetuada a priori uma **relocação (estática)** das rotinas que compõem o programa, mediante a **alteração física dos endereços relativos** internos a cada módulo.
- Tal operação simplifica o código objeto, produzindo **um único módulo** composto, em que figuram apenas referências relocáveis na forma de endereços **relativos a uma única base**.
- Convém, em cada caso particular, levar em conta se esta operação traz reais vantagens, dado que se trata de um procedimento não obrigatório, o qual pode tornar-se oneroso, se sistematicamente utilizado.

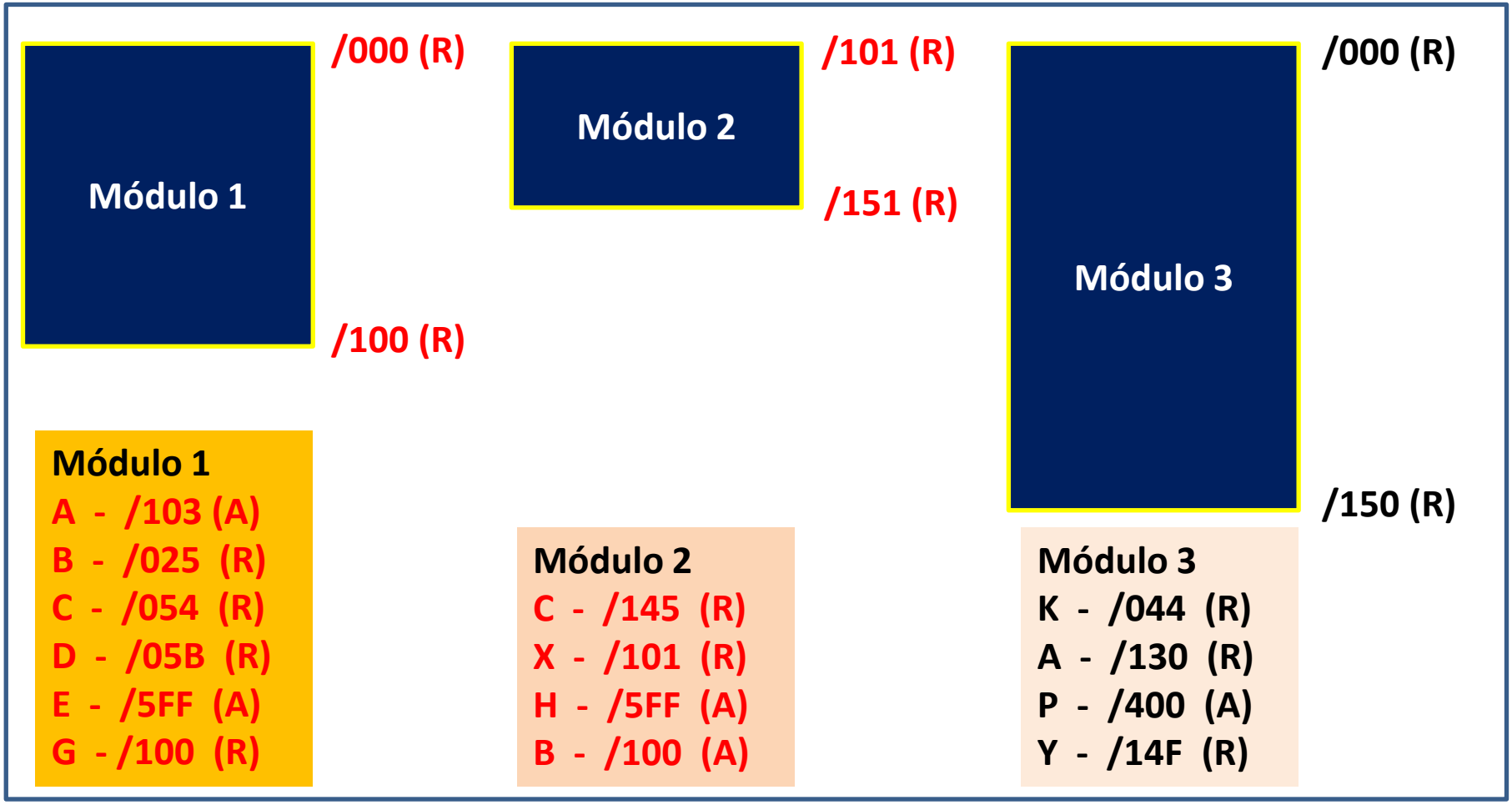
Relocação estática intermediária (situação inicial)



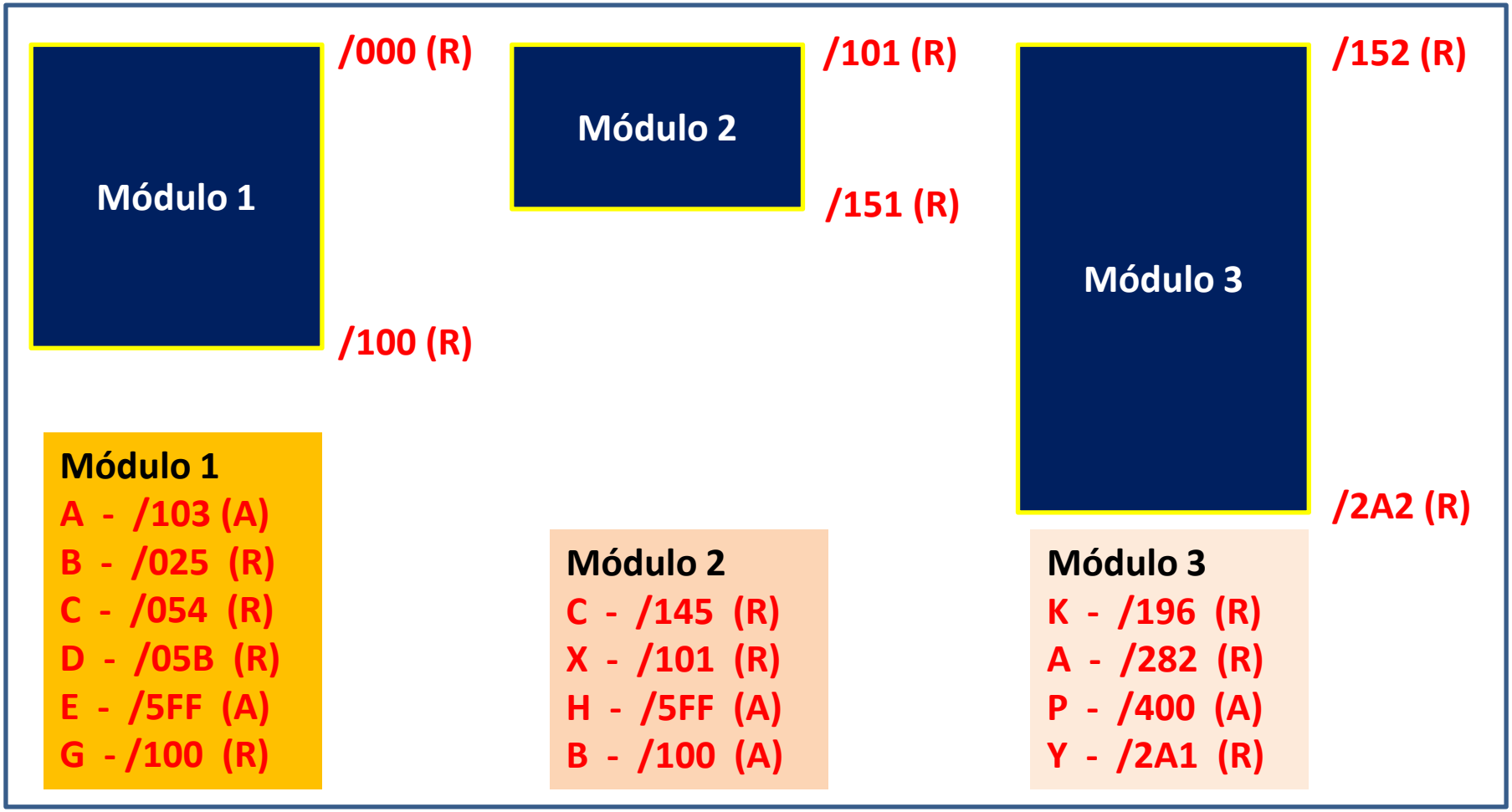
Alocação com relocação,módulo a módulo (Módulo 1: alocação em /000 (R))



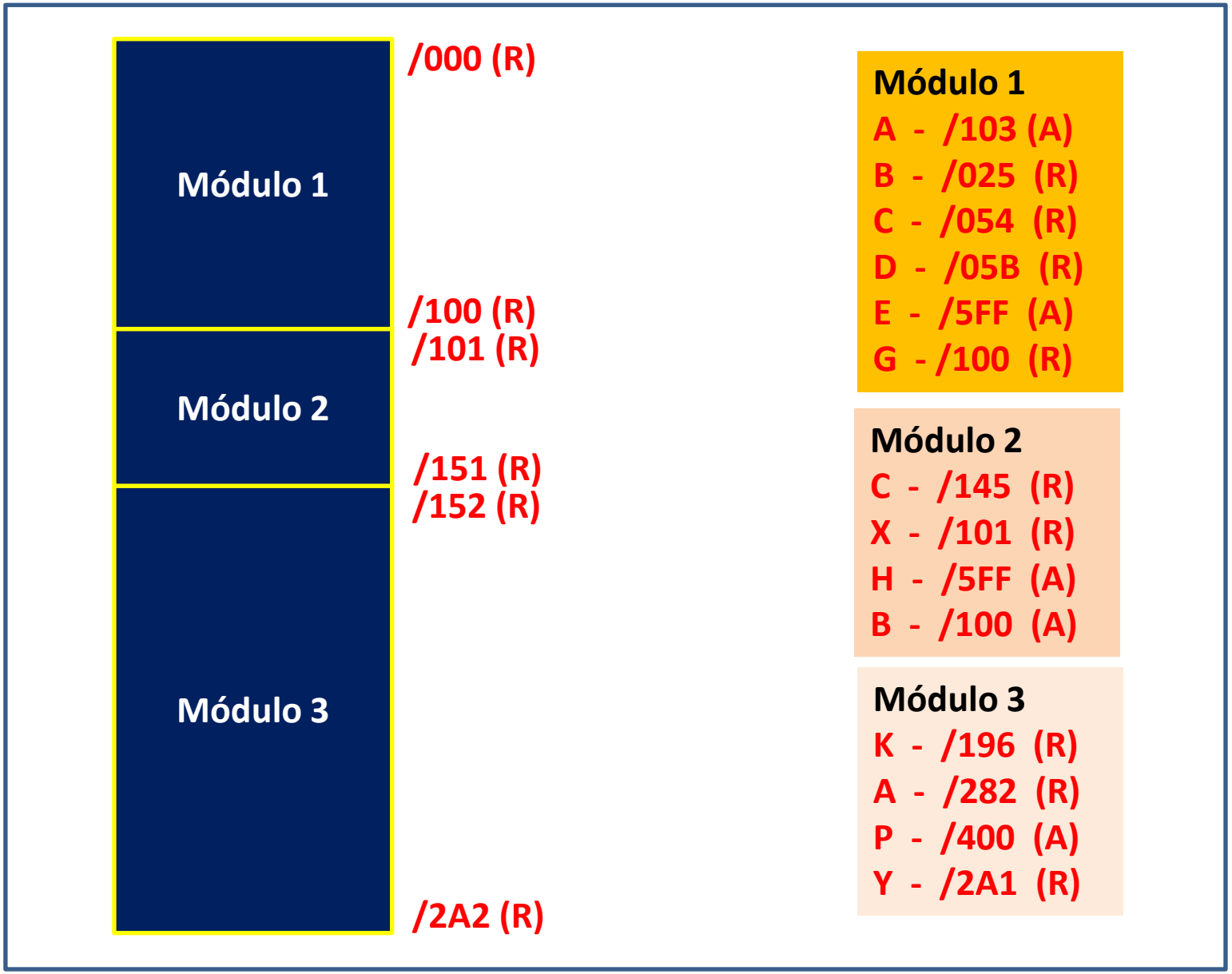
Alocação com relocação,módulo a módulo (Módulo 2: alocação em /101 (R))



Alocação com relocação,módulo a módulo (Módulo 3: alocação em /152 (R))



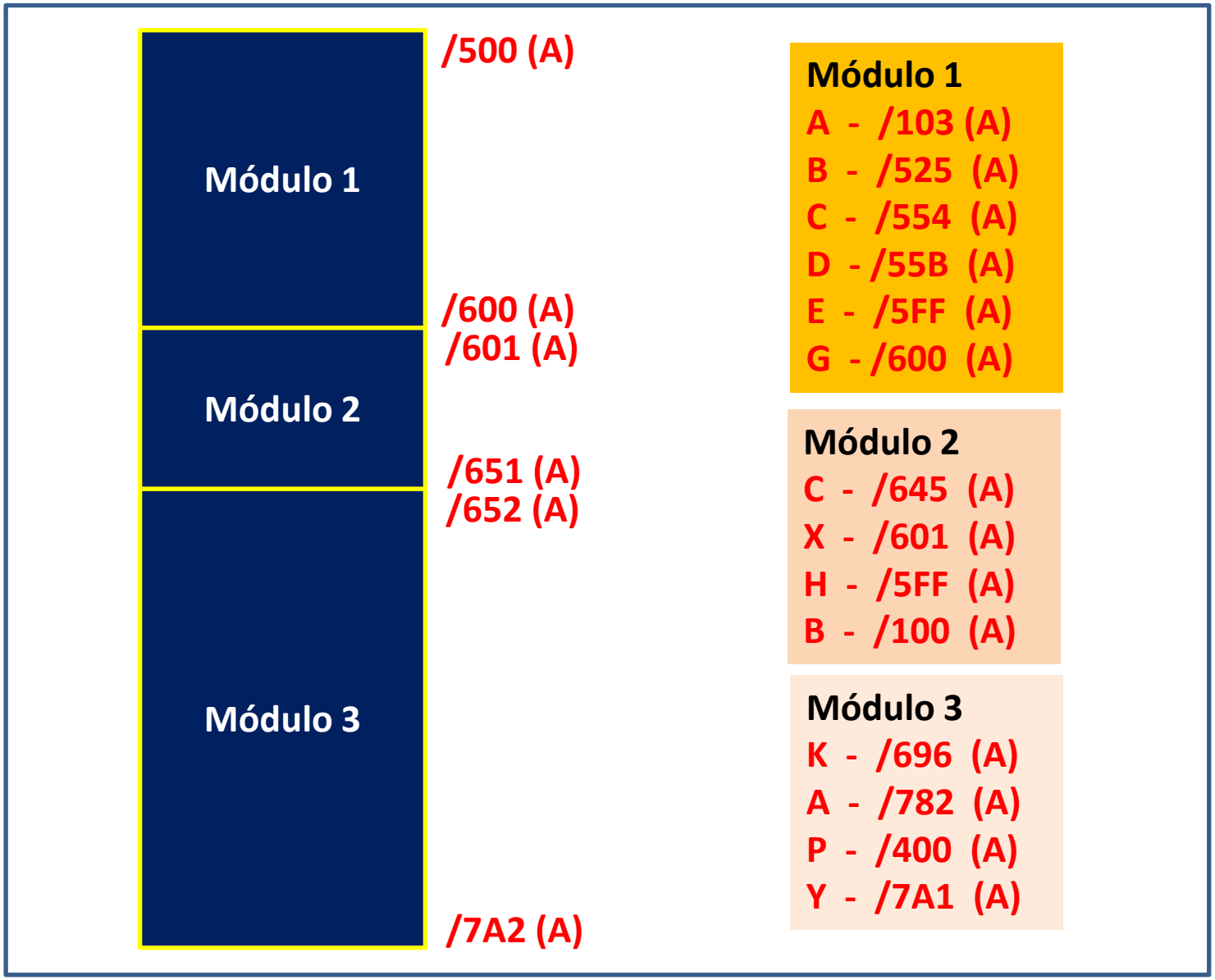
Result.final da reloc.estática intermediária



Relocação final

- A relocação estática intermediária ainda não produz, como pode ser observado, um código executável, pois muitas referências permanecem relocáveis após sua aplicação.
- Por outro lado, o seu resultado final é um **código objeto relocável sem referências simbólicas**, pronto para ser relocado a partir de uma base única, a ser arbitrada.
- Para obter portanto um **código absoluto**, pronto para ser carregado na memória para execução, basta efetuar a **relocação final** a partir de uma origem (base de relocação), geralmente estabelecida pelo usuário ou então pelo sistema operacional.
- Se, ao resultado da relocação estática intermediária do exemplo anterior, aplicarmos o endereço absoluto **/500 como base de relocação** para construir um código pronto para a execução, com todos os endereços resolvidos, obtemos o resultado que pode ser visualizado a seguir.

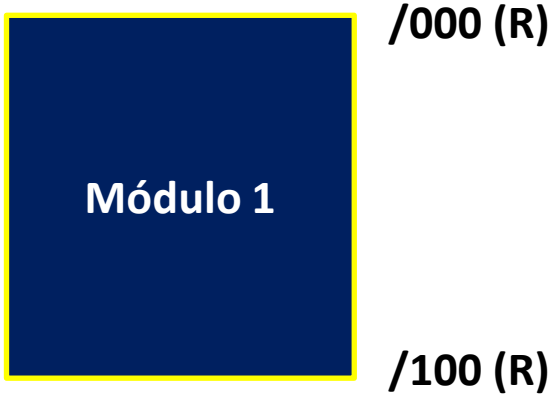
Código final, relocado p/exec. em /500 (A)



Fase de relocação

- Qualquer que seja o caso, portanto, é de grande importância, sempre que necessário, o ajuste *a priori* de todos os endereços relativos contidos no código do programa, antes da sua execução.
- A operação de ***relocação estática*** constitui uma tarefa fundamental na preparação de programas para a execução em um Sistema de Programação.
- A atividade de um Sistema de Programação para resolver completamente os endereços do programa, (convertê-los em programas absolutos) constitui a ***fase de relocação*** dos módulos que compõem esses programas.

Situação inicial: relocação módulo a módulo



/000 (R)

/100 (R)

/000 (R)

/050 (R)

/000 (R)

/150 (R)

Módulo 1
A - /103 (A)
B - /025 (R)
C - /054 (R)
D - /05B (R)
E - /5FF (A)
G - /100 (R)

Módulo 2
C - /044 (R)
X - /000 (R)
H - /5FF (A)
B - /100 (A)

Módulo 3
K - /044 (R)
A - /130 (R)
P - /400 (A)
Y - /14F (R)

Alocação com relocação, módulo a módulo (Módulo 1: alocação em /500 (A))

/500 (A)

/600 (A)

/000 (R)

/050 (R)

/000 (R)

/150 (R)

Módulo 1
A - /103 (A)
B - /525 (R)
C - /554 (R)
D - /55B (R)
E - /5FF (A)
G - /600 (R)

Módulo 2
C - /044 (R)
X - /000 (R)
H - /5FF (A)
B - /100 (A)

Módulo 3
K - /044 (R)
A - /130 (R)
P - /400 (A)
Y - /14F (R)

Alocação com relocação, módulo a módulo (Módulo 2: alocação em /601 (A))

/500 (A)

/600 (A)

/601 (A)

/651 (A)

/000 (R)

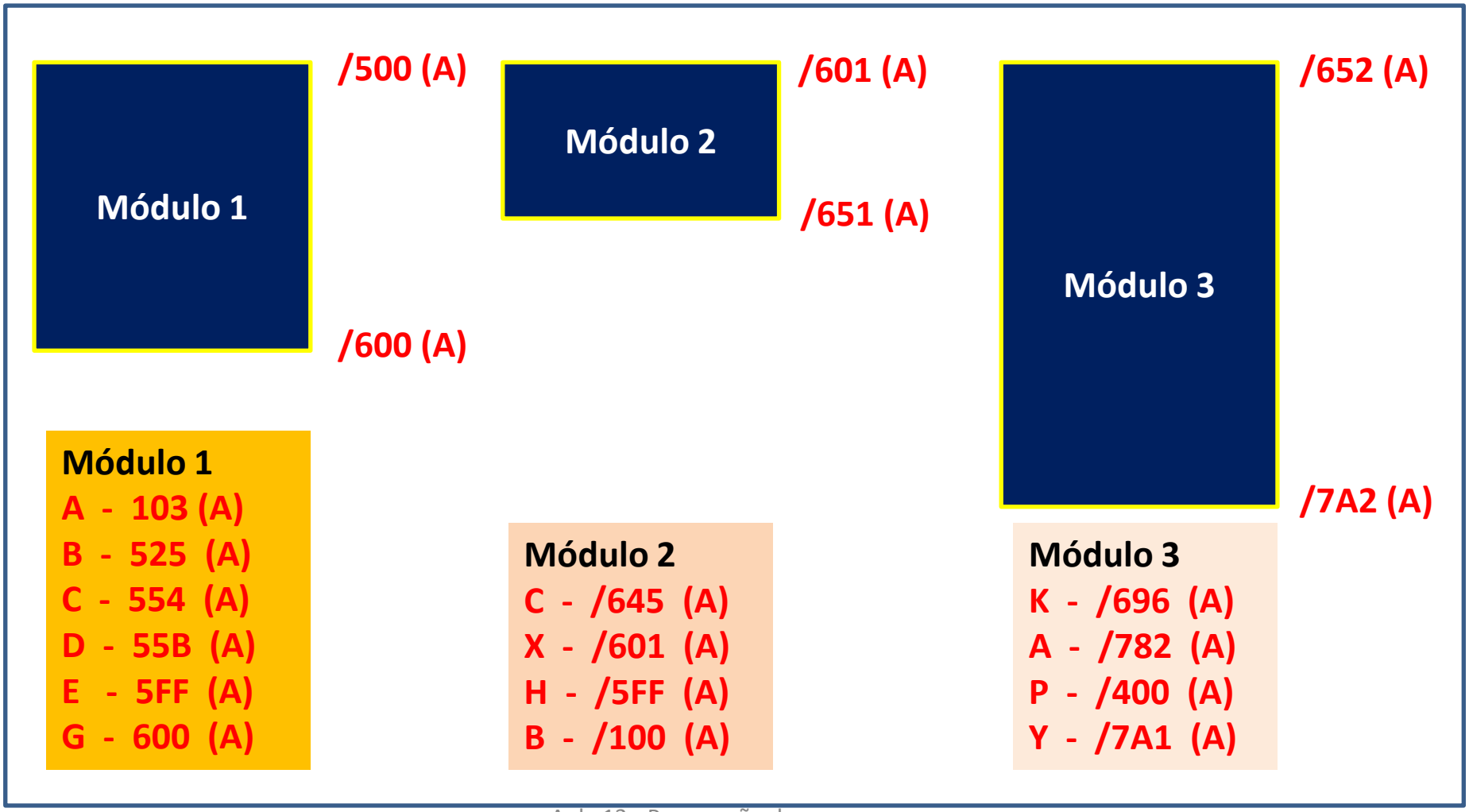
/150 (R)

Módulo 1
A - 103 (A)
B - 525 (A)
C - 554 (A)
D - 55B (A)
E - 5FF (A)
G - 600 (A)

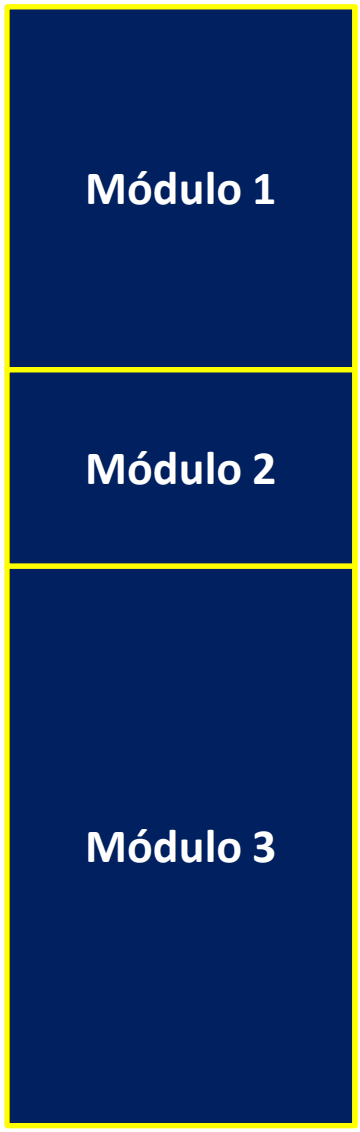
Módulo 2
C - /645 (A)
X - /601 (A)
H - /5FF (A)
B - /100 (A)

Módulo 3
K - /044 (R)
A - /130 (R)
P - /400 (A)
Y - /14F (R)

Alocação com relocação, módulo a módulo (Módulo 3: alocação em /652 (A))



Código final resultante da relocação



/500 (A)

/600 (A)
/601 (A)

/651 (A)
/652 (A)

/7A2 (A)

Módulo 1
A - /103 (A)
B - /525 (A)
C - /554 (A)
D - /55B (A)
E - /5FF (A)
G - /600 (A)

Módulo 2
C - /645 (A)
X - /601 (A)
H - /5FF (A)
B - /100 (A)

Módulo 3
K - /696 (A)
A - /782 (A)
P - /400 (A)
Y - /7A1 (A)

Notas sobre a relocação apresentada

- Inicialmente havia **3 módulos relocáveis**, contendo apenas referências internas, a endereços absolutos e relocáveis.
- Após a relocação separada de cada módulo, as referências internas **absolutas se mantêm** e as **relocáveis se transformam em absolutas** mediante a **soma da base** de relocação específica de cada módulo.
- A base de relocação vai sendo atualizada de acordo com a ocupação da memória pelo código absoluto em construção.
- A **política de alocação** adotada posiciona **contiguamente** na memória os módulos à medida que vão sendo relocados.
- Ao final, todos os endereços se tornam **absolutos** e os três módulos se transformam em um código único, ocupando **posições adjacentes de memória**.

Endereçamento simbólico

- Outro tipo ainda de endereçamento, encontrado em programas objeto relocáveis, é denominado **endereçamento simbólico**, e corresponde a **referências nominais** a posições de memória cujos endereços são desconhecidos no instante em que o programa é escrito, presumindo-se a sua existência em um dos demais módulos de que se compõe o programa.
- Para que esse tipo de referência possa ser utilizado de forma prática, o módulo **portador da declaração do símbolo** deve informar o fato, da mesma forma que o módulo que **deseja fazer referência** a um desses símbolos deve acusar essa intenção ao montador.
- Esse protocolo costuma ser estabelecido através do uso de **duas pseudo-instruções**, uma para informar os símbolos que o módulo **disponibiliza** aos demais, e outra, para indicar os símbolos que o módulo **deseja acessar**, mas que pertençam a outros módulos.

Referências externas

- Seria inútil a possibilidade de se efetuar programação modular se fossem completamente estanques os módulos de que o programa se compõe.
- Assim, é conveniente que cada módulo possa utilizar-se das funções executadas pelos demais, o que se faz mediante referência mútua, com o auxílio de instruções de referência à memória com operandos especiais.
- Para isso, usa-se o conceito de ***referências externas,*** que são operandos que fazem referência a símbolos externos aos módulos, ou seja, endereços simbólicos declarados em outro módulo do programa, diferente daquele no qual são referenciados.

Ponto de acesso (*export*)

- Globalmente, a referência mútua entre os módulos é viável apenas para determinados pontos de cada módulo, indicados pelo programador para serem visíveis aos demais módulos.
- Estes símbolos são declarados, em cada módulo, como sendo ***exports, entry points*** ou ***pontos de acesso*** de entrada ao módulo.
- Cada módulo pode, usualmente, apresentar um ou mais pontos de acesso, cada qual designado por um nome simbólico.

Endereço externo (*import*)

- Estes nomes simbólicos funcionam como se fossem **endereços simbólicos globais**, acessíveis a qualquer dos módulos que deles necessite.
- Nestes, os endereços simbólicos globais a serem referenciados devem ser declarados como sendo ***imports*** ou **endereços externos** (***external addresses***).

Referências entre módulos

- Com esse mecanismo, cria-se um protocolo de comunicação entre módulos desenvolvidos em linguagem simbólica, através da **referência mútua entre os módulos**, efetuada por meio do endereçamento simbólico.
- Tais endereçamentos são **resolvidos fora do montador**, pelos mecanismos do Sistema de Programação responsáveis pela operação de **ligação** (***linking***, ***binding***)

Resolução de referências entre módulos

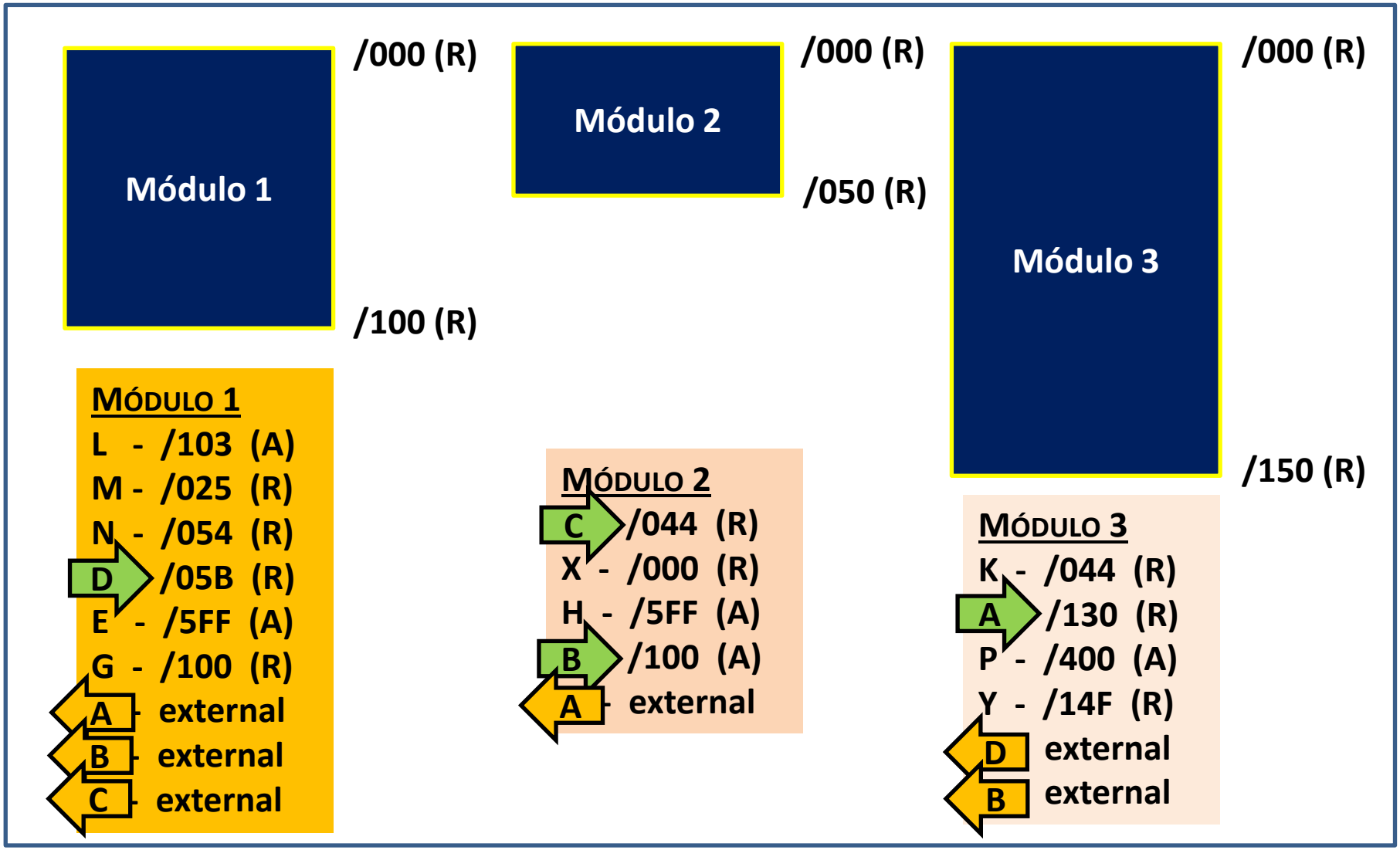
- Para que seja possível a execução de um programa que apresente endereçamento simbólico para referências entre módulos, é necessário converter tais endereços simbólicos em endereços relativos ou absolutos, recaindo então nos casos anteriores.
- Um programa nessas condições se torna **apto para a execução** quando **cada uma das suas referências** do tipo ***import*** tiver sido **satisfeita** pela presença de declarações únicas das respectivas referências, na forma de símbolos do tipo ***export***, além das ações de alocação e de relocação.
- A estratégia, portanto, para recair nas situações já estudadas, consiste apenas em **efetuar em primeiro lugar uma relocação estática intermediária**, com a finalidade de remover as referências simbólicas entre os módulos.
- Para finalizar, basta apenas efetuar as operações de alocação e de relocação, da forma como foi visto anteriormente.

EXEMPLO COMPLETO

Um exemplo detalhado completo

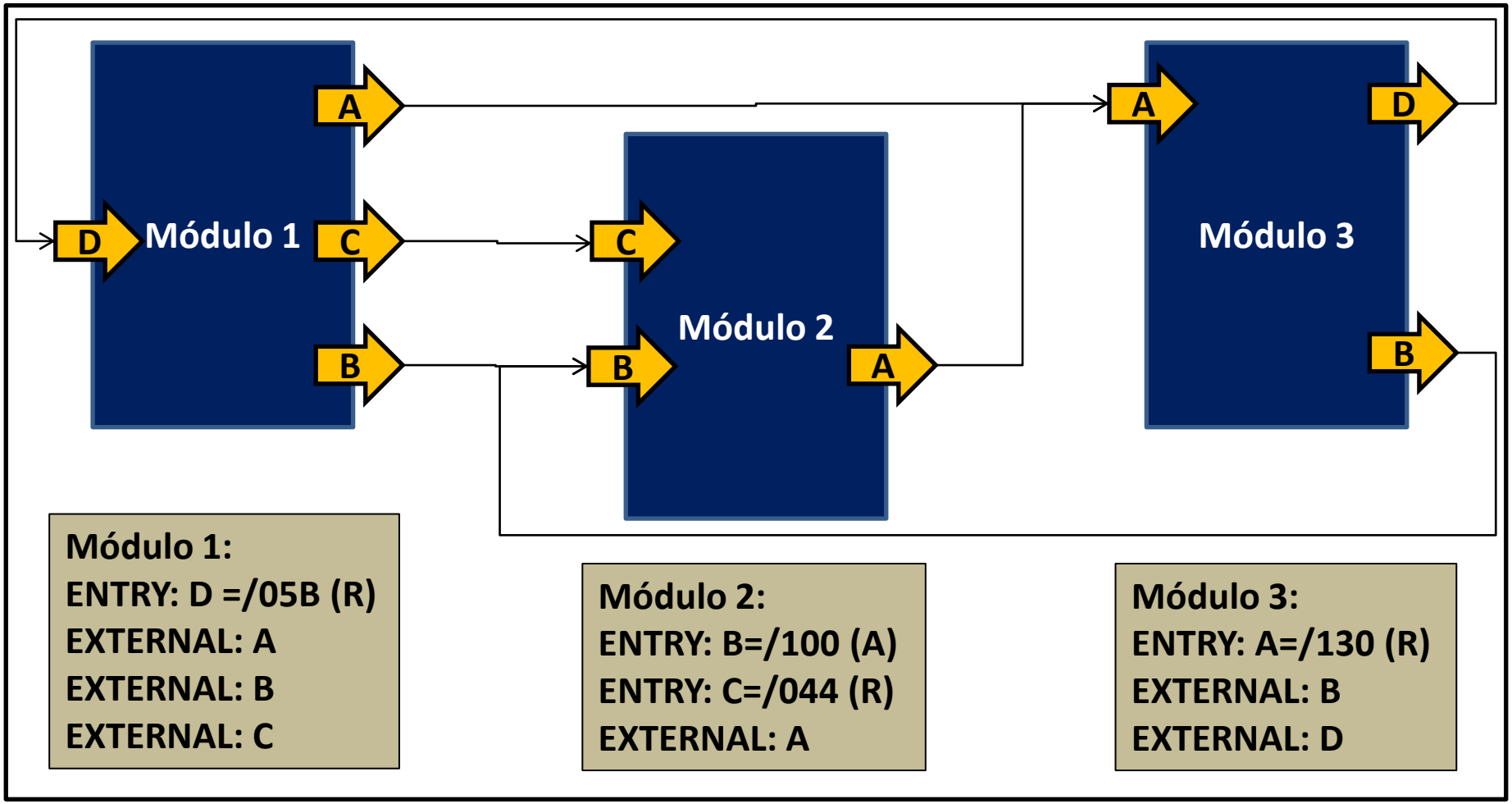
- Neste exemplo, são usados **os mesmos três módulos** dos casos ilustrativos anteriores
- Foram apenas **incorporadas algumas referências simbólicas** entre módulos:
  - O módulo 1 tem D como *entry point,* e como *externals*, A, B e C
  - O módulo 2 tem B e C como *entry points*, e como *external*, A
  - O módulo 3 tem A como *entry point*, e como *externals*, B e D
- Agregando essas informações à situação inicial já apresentada, temos a **situação inicial** deste exemplo.
- Assim, tem-se também, além das referências internas absolutas e relocáveis, as **quatro referências simbólicas** A, B, C e D introduzidas como entry points e externals, e que precisam ser resolvidas.
- A primeira operação a realizar é a **relocação estática intermediária**, com a finalidade de eliminar as referências simbólicas entre módulos através da construção de um só módulo relocável a partir dos três inicialmente apresentados, e em seguida, alocar e relocar o módulo relocável assim obtido.
- ***Observação:*** adotada aqui apenas por razão didática, **a realização em separado** da relocação estática intermediária **poderia ser evitada na prática** construindo-se de uma só vez o código absoluto desejado, para isso **juntando** as operações de **relocação estática intermediária, de alocação e de relocação** final a partir de uma base absoluta de relocação.

Situação inicial, c/referências entre módulos

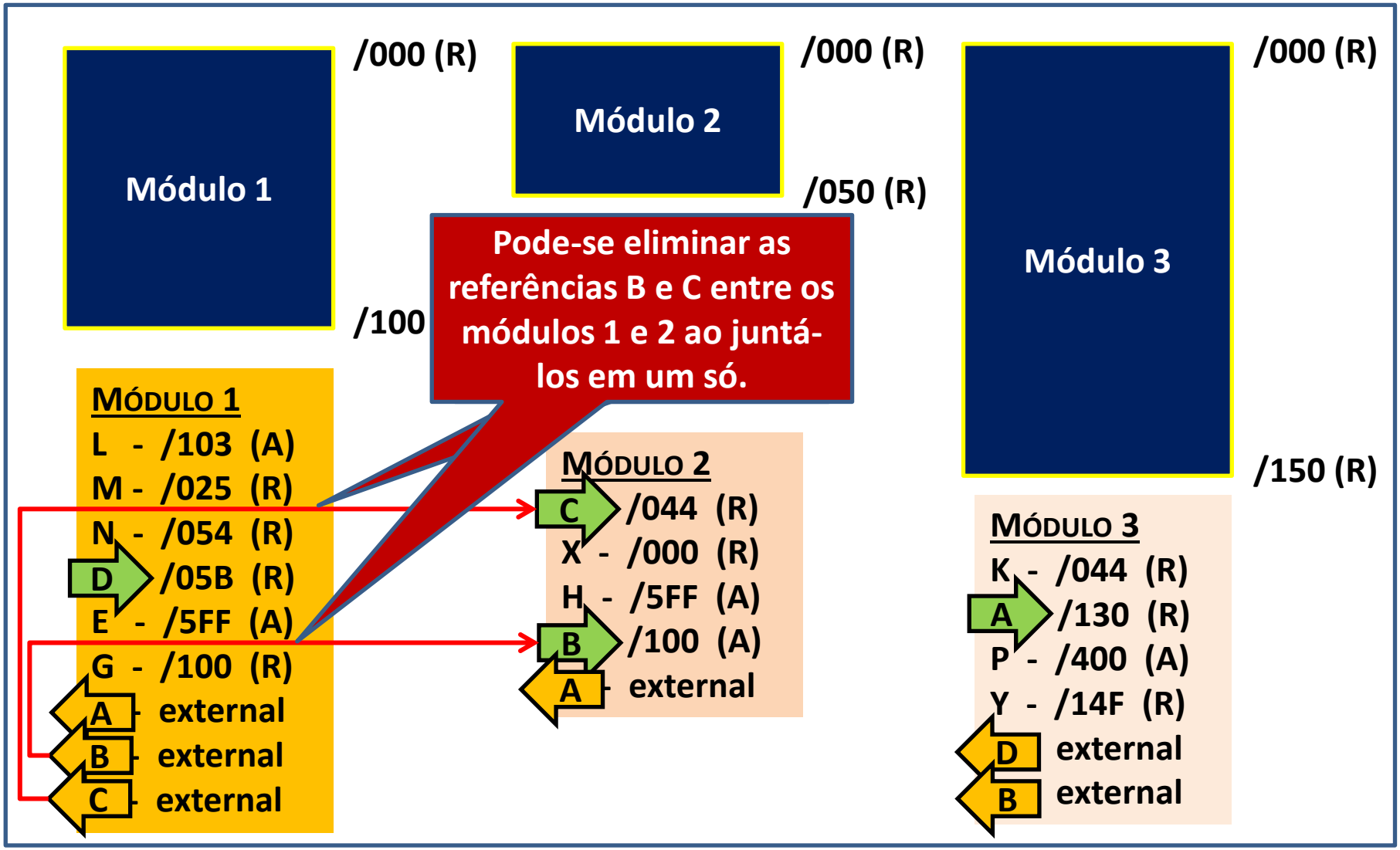


Esquema da ligação das referências simbólicas

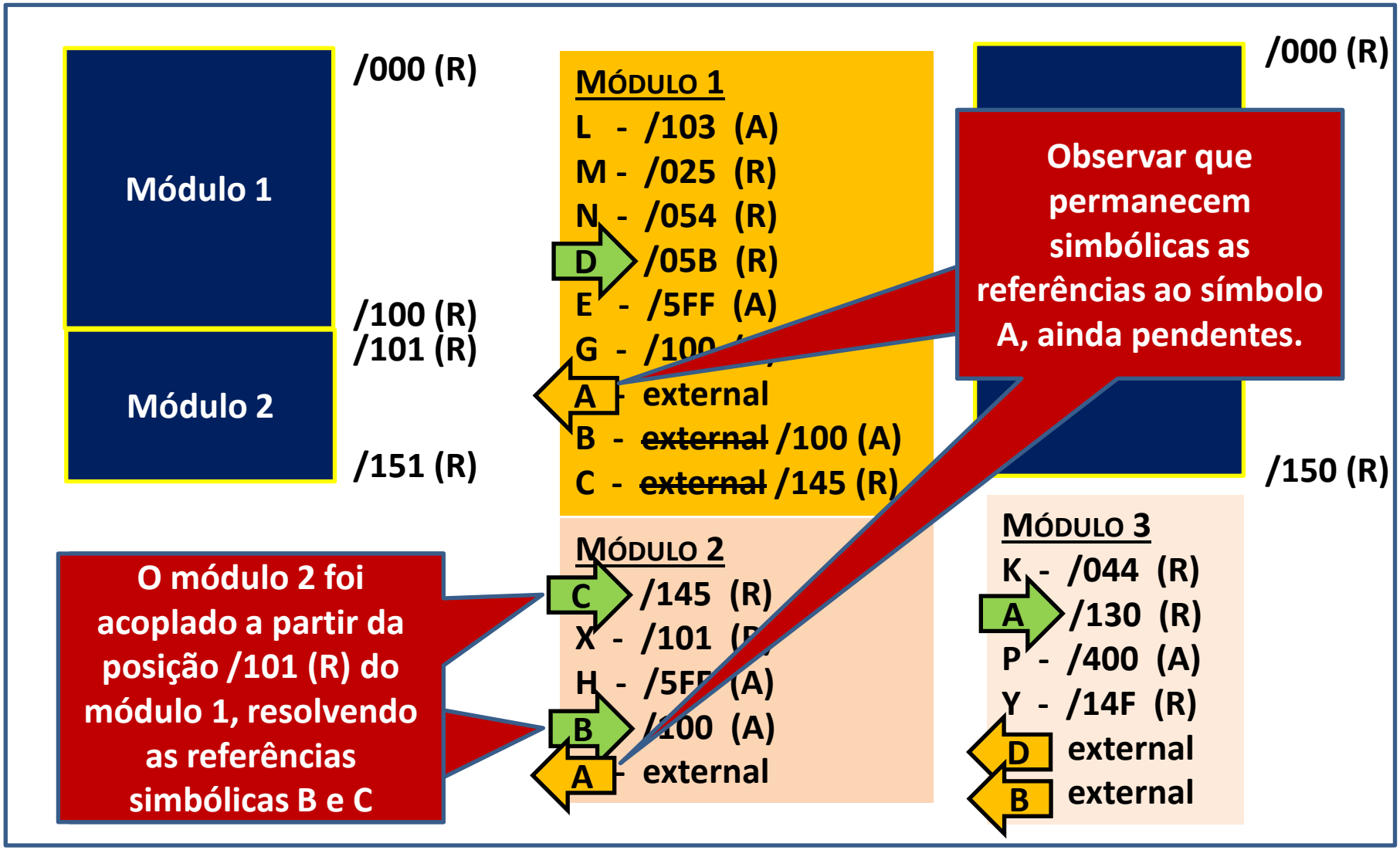
Este diagrama mostra apenas as conexões simbólicas existentes entre os três módulos iniciais, e os endereços associados a elas no módulo a que pertencem.



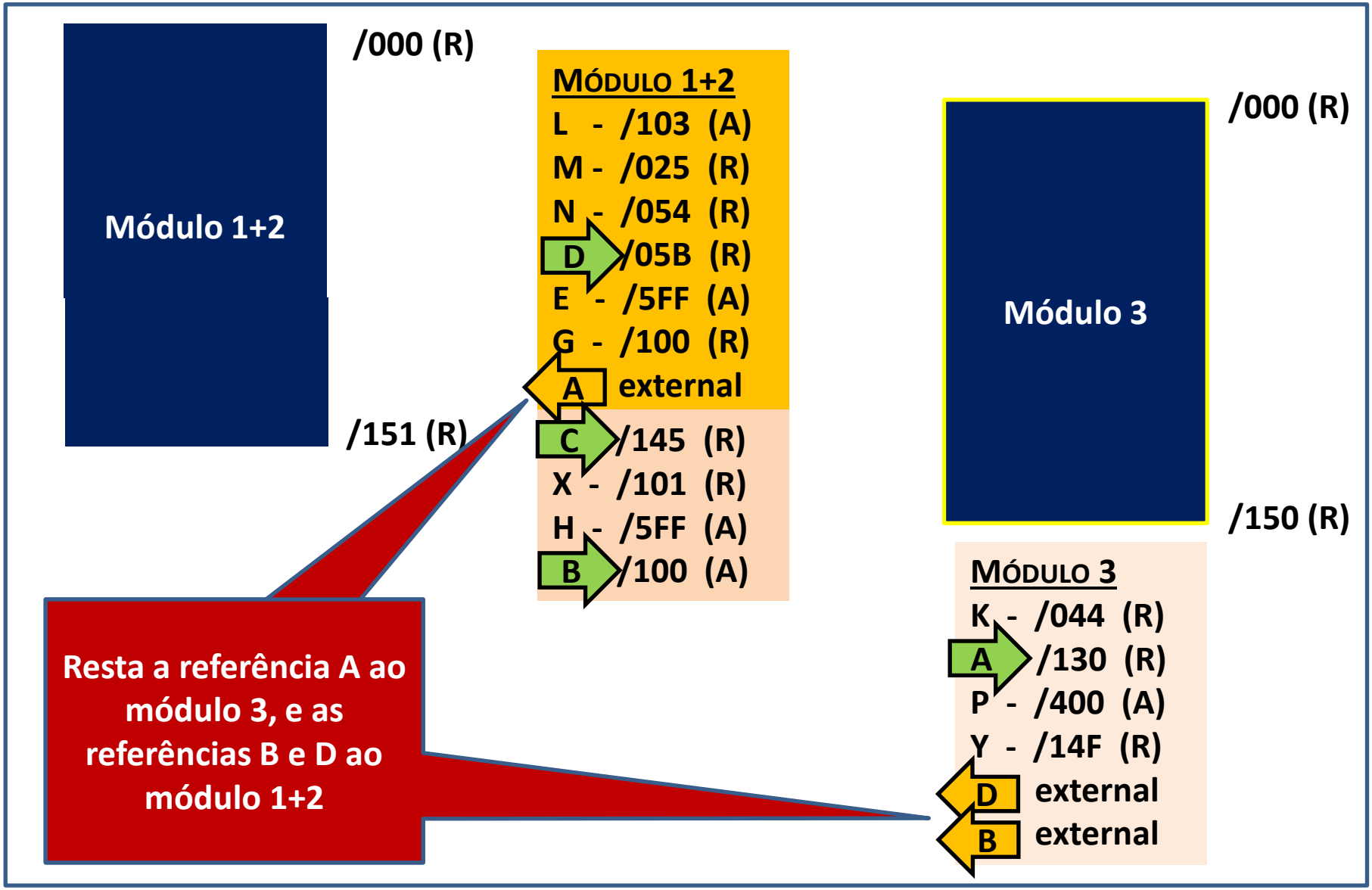
Resolução de referências simbólicas entre os módulos 1 e 2



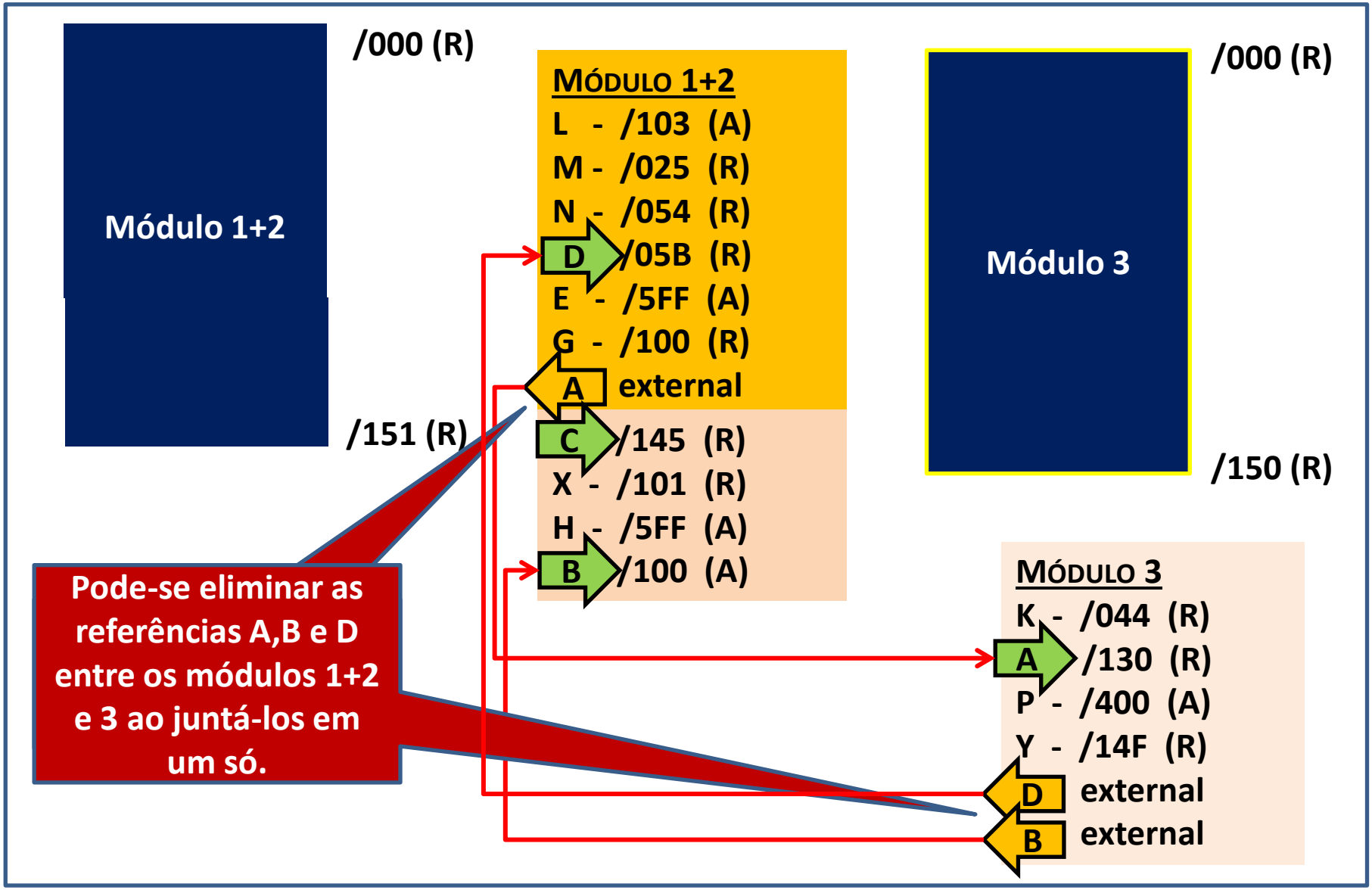
Incorporação do módulo 2 ao módulo 1



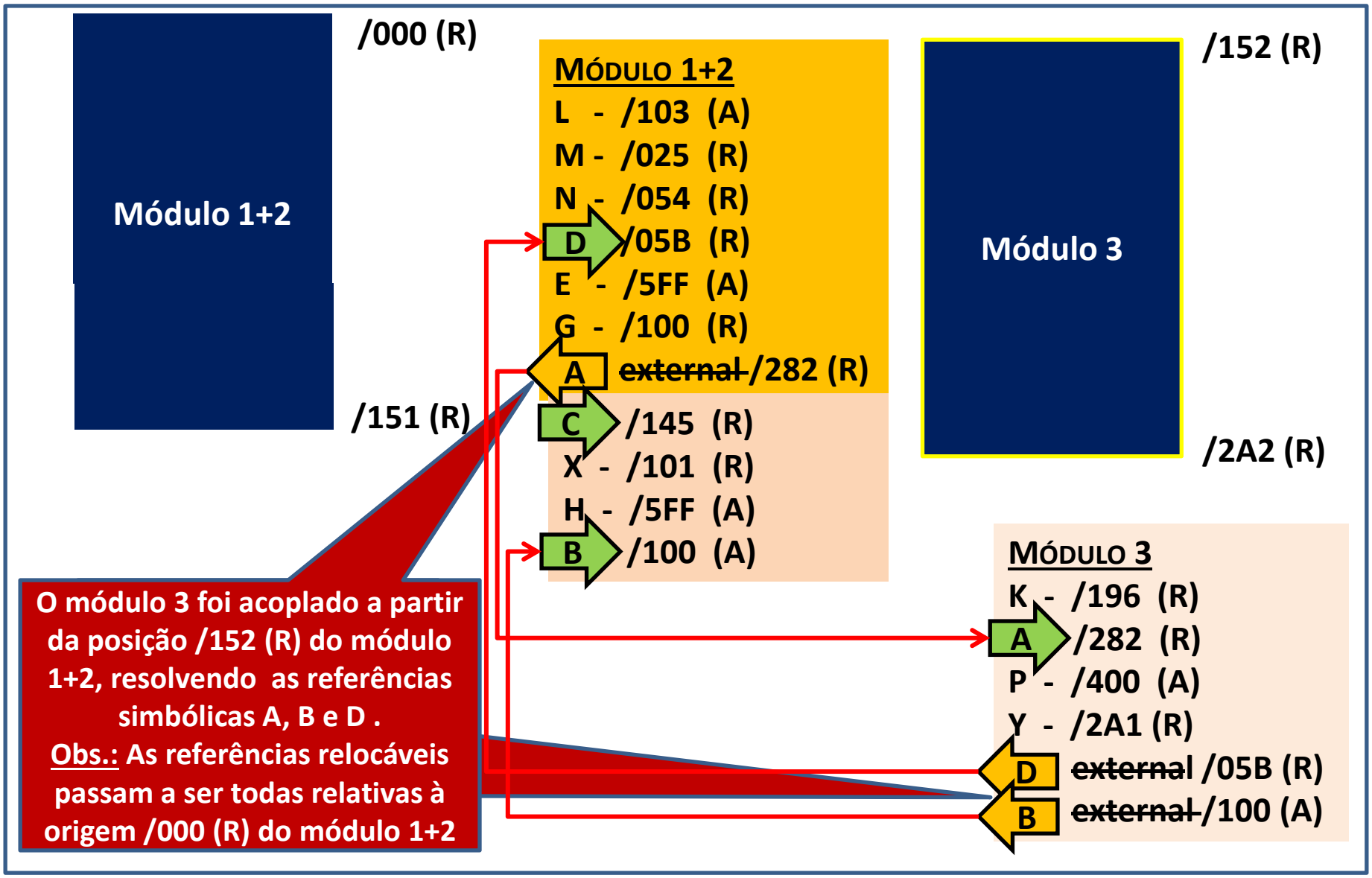
Resultado da fusão dos módulos 1 e 2



Resolução das referências entre 1+2 e 3



Resolução das referências entre 1+2 e 3



Módulo relocável único resultante

MÓDULO 1+2+3

L - /103 (A)
M - /025 (R)
N - /054 (R)
D - /05B (R)
E - /5FF (A)
G - /100 (R)
C - /145 (R)
X - /101 (R)
H - /5FF (A)
B - /100 (A)
K - /044 (R)
A - /130 (R)
P - /400 (A)
Y - /2A1 (R)

Alocação em /500 (A) e relocação final

ENDEREÇOS ABSOLUTOS FINAIS

L - /103 (A)
M - /525 (A)
N - /554 (A)
D - /55B (A)
E - /5FF (A)
G - /600 (A)

C - /645 (A)
X - /601 (A)
H - /5FF (A)
B - /100 (A)

K - /544 (A)
A - /630 (A)
P - /400 (A)
Y - /7A1 (A)

Comentários sobre o exemplo

- Estão envolvidos no processo **três módulos (1, 2 e 3)**
  - O módulo 1 tem D como *entry point* e A,B,C como *externals*
  - O módulo 2 tem como *entry points* B,C e como *external*, A
  - O módulo 3 tem como *entry point* A, e como *externals*, B e D
- Essas relações simplesmente **associam** os ***entry points*** aos correspondentes ***externals,*** em outro módulo.
- Notar que, feitas todas as associações, **nenhum *external*** fica **sem conectar-se ao *entry point*** correspondente.
- **Este mecanismo**, similar ao desempenhado no primeiro passo do montador para a construção da tabela de símbolos, **é executado por um** módulo do sistema de programação que desempenhe o papel de **ligador (*linker*).**

FIM