

Organización y estructura del lenguaje VHDL.

PROGRAMACIÓN DE SISTEMAS EMBEBIDOS

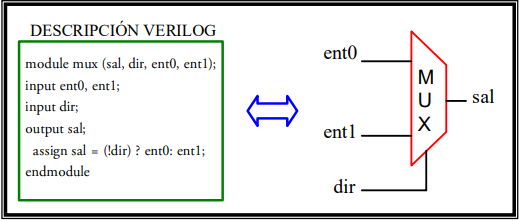


24 de marzo de 2020

nadia sarahi murguia chavez

ING. MECATRONICA 8vo A

Los lenguajes de descripción hardware (HDLs, Hardware Description Languages) viene utilizándose desde los años 70 en los ciclos de diseño de sistemas digitales asistidos por herramientas de CAD electrónico. Al principio sugirieron una serie de lenguajes que no llegaron a alcanzar un éxito que permitiera su consolidación en el campo industrial o académico. En los años 80 apresen los lenguajes Verilog y VHDL que, aprovechando la disponibilidad de herramientas hardware y software cada vez más potentes y accesibles y los adelantos en las tecnologías de fabricación de circuitos integrados, logran imponerse como herramientas imprescindibles en el desarrollo de nuevos sistemas. En la actualidad ambos lenguajes están normalizados y comparten, en poco tiempo eliminando del mercado, al resto de lenguajes que de un modo u otro todavía son soportados por algunas herramientas CAD.



Estos lenguajes son sintácticamente similares a los de programación de alto nivel Verilog tiene una sintaxis similar al C y VHDL a ADA y se diferencian de estos en sus semánticas estas orientadas al modelado del hardware. Su capacidad para permitir distintos enfoques en el modelado de los circuitos y su independencia de la tecnología y metodología de diseño permite extender su uso a los distintos ciclos de diseño que puedan utilizarse. Por ello, para los profesionales relacionados de alguna manera con el diseño o mantenimiento de sistemas digitales resulta hoy en día imprescindible su conocimiento.

El lenguaje VHDL.

Los estudios para la creación del lenguaje VHDL (VHSIC HDL) comenzaron en el año 1981, bajo la cobertura de un programa para el desarrollo de circuitos integrados de muy alta velocidad (VHSIC), del departamento de defensa de los Estados Unidos. En 1983 las compañías intermetrics, IBM y Texas Instruments obtuvieron la concesión de un proyecto para la realización del lenguaje y de un conjunto de herramientas auxiliares para su aplicación. Finalmente, en el año 1987, el lenguaje VHDL se convierte en la norma IEEE-1076 como todas las normas IEEE, se somete a revisión periódicas, por lo que en 1993 sufrió algunas leves modificaciones.

Características del lenguaje.

El lenguaje VHDL fue creado con el propósito de especificar y documentar circuitos y sistemas digitales utilizando un lenguaje formal. En la práctica se ha convertido, en un gran número de entornos de CAD, en el HDL de referencia para realizar modelos sintetizables automáticamente. Las principales características del lenguaje VHDL se explican en los siguientes puntos:

* Descripción textual normalizada: El lenguaje VHDL es un lenguaje de descripción que especifica los circuitos electrónicos en un formato adecuado para ser interpretado tanto por maquinas como por personas. Se trata además de un lenguaje formal, es decir, no resulta ambiguo a la hora de expresar el comportamiento o representar la estructura de un circuito. Esta, como ya se ha dicho, normalizado, o sea, existe un único modelo para el lenguaje, cuya utilización está abierta a cualquier grupo que quiera desarrollar herramientas basadas en dicho modelo, garantizado su compatibilidad con cualquier otra herramienta basada que respete las indicaciones específicamente en la norma oficial. Es, por último, un lenguaje ejecutable, lo que permite que la descripción textual del hardware se materialice en una representación del mismo utilizable por herramientas auxiliares tales como simuladores y sintetizadores lógicos, compiladores de silicio, simuladores de tiempo, de cobertura de fallos, herramientas de diseño físico, etc.
* Amplio rango de capacidad descriptiva: EL lenguaje VHDL posibilita la descripción del hardware como distintos niveles de abstracción, pudiendo adaptarse a distintos propósitos y utilizarse en las sucesivas fases que se dan en el desarrollo de los diseños. Además, es un lenguaje adaptable a distintas en el primer caso, cubrir el tipo de necesidades de los distintos géneros de instituciones, compañías y organizaciones relacionadas con el mundo de la electrónica digital; y, en el segundo, facilita la actualización y adaptación de los diseños a los avances de la tecnología en cada momento.
* Otras ventajas: Además de las ventajas ya reseñadas también es destacable la capacidad del lenguaje para el manejo de proyectos de grandes dimensiones, las garantías que comportan su uso cuando, durando el ciclo de mantenimiento del proyecto, hay que sustituir componentes o realizar modificaciones en los circuitos, y el hecho de que, para muchas organizaciones contratantes, sea parte indispensable de la documentación de los sistemas.

Modela VHDL.

1. Caracterización de circuitos.

Para entender en que consiste el modelado lógico de un circuito, hay que apreciar los dos aspectos que los caracterizan:

1. Un interfaz externo: Una puerta AND de dos entradas, por ejemplo, tiene tres terminales, dos entradas y una salida, y se diferencia de una puerta AND de tres entradas, en dos cosas: el número de terminales y el nombre (el nombre de una es AND de dos entradas, el de la otra AND de tres entradas)
2. Un algoritmo de procesamiento de la información: Cada dispositivo digital realiza una determinada operación que le permite obtener ciertos niveles lógicos en sus terminales de salida a partir de los aplicados en sus entradas: una puerta XOR pone un uno cuando sus dos entradas son distintas, una puerta NOR cuando todas sus entradas tienen un cero.

En los planos de un circuito la representación del interfaz de un dispositivo se realiza gráficamente mediante el dibujo de un determinado símbolo: son característicos y conocidos por todos los de las puertas lógicas o los flip-flops; los bloques funcionales se representan normalmente mediante rectángulos con algún texto distintivo asociado (sumador, decodificado, etc.) y líneas y etiquetas para la especificación de los terminales. Con ese sistema parece, a veces, que el símbolo aporta alguna información sobre el funcionamiento: si vemos el símbolo de una puerta AND sabemos cuál es la función lógica que realiza, pero este conocimiento es aprendido y procede de la asociación de una determinada tabla de verdad, la de función lógica que realiza, pero este de verdad, la de la función lógica AND, con la forma del símbolo que representa el interfaz de la puerta AND.

La función lógica que realiza un circuito o dispositivo identificable mediante un símbolo se describe (en catálogos de fabricantes, libros de texto o documentaciones de proyectos) mediante tablas de verdad y cronogramas y, cuando resulta conveniente, mediante anotaciones adicionales en lenguaje corriente.

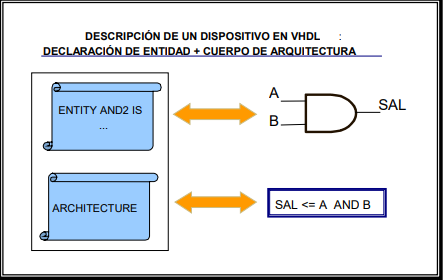
El modelo de un dispositivo o circuito digital se completa añadiendo al modelo lógico información referente a su comportamiento dinámico (indicando tiempos de retardo entre entradas y salidas, tiempos de set-up, etc.), y sus características eléctricas (corrientes, niveles de tensión, carga, disipación de potencia) y físicas (encapsulados, dimensiones).

Los lenguajes de especificación hardware como VHDL están orientados al modelo de las características lógicas y dinámicas del hardware. El modelo de un dispositivo “simple” (uno que no está compuesto por dos o más conectados entre sí) los dispositivos “complejos” pueden describirse mediante la conexión de otros (con un lenguaje de descripción hardware consiste en la descripción de su interfaz y su comportamiento) incluyendo características dinámicas, haciendo uso de las construcciones de que disponga el lenguaje y respetando las reglas que se impongan.

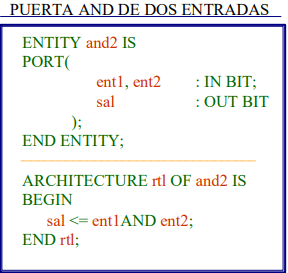
1. La declaración de entidad y el cuerpo de arquitectura.

La realización del modelo hardware del modelo hardware de un dispositivo en VHDL consiste en la elaboración de dos unidades de código VHDL: una declaración de entidad y el cuerpo de arquitectura.

* La declaración de entidad es la unidad de diseño VHDL que sirve para especificar el interfaz de los dispositivos. Cumple, por tanto, funciones equivalentes a las de los símbolos en las representaciones gráficas.
* El cuerpo de arquitectura es la unidad de diseño VHDL que sirve para especificar el funcionamiento de un dispositivo identificado por una determinada declaración de entidad, por lo que se puede considerar el equivalente a las tablas de verdad o a los cronogramas.



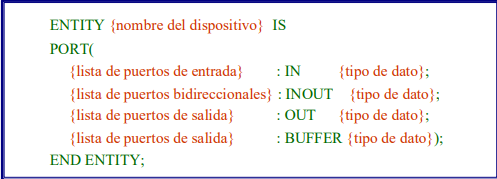
En la declaración de entidad se define el nombre del dispositivo y sus puertos; en el cuerpo de arquitectura su funcionamiento, en este caso mediante una sentencia que utiliza un operador lógico.



La construcción del modelo de un dispositivo en un entorno VHDL finaliza, en principio, en el momento en que las unidades VHDL que lo describen quedan almacenadas en una librería VHDL. Para ello hay que editar las unidades y compilarlas. En el proceso de compilación se comprueba que no se incumplen una serie de reglas sintácticas y semánticas.

1. Sintaxis básicas de la declaración de entidad.

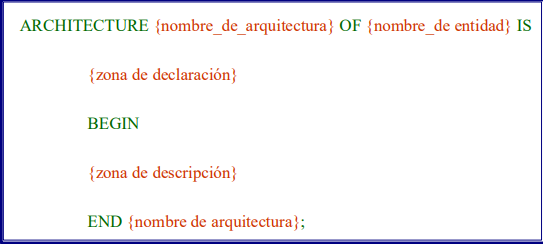
La sintaxis básica de la declaración de entidad es la siguiente:



Obviamente, el identificador que sigue a la palabra ENTITY es el nombre del dispositivo. Los puertos del dispositivo se especifican, entre paréntesis, en la lista de interfaz de la declaración de entidad, que es el campo señalado por la palabra clave PORT. Para cada puerto hay que declarar:

1. Su nombre: Los nombres de los puertos son etiquetas definibles por el usuario. No pueden utilizarse palabras reservadas del lenguaje.
2. Su dirección: La dirección del puerto se determina a partir de las características del terminal del dispositivo que se desea modelar: los pines de entrada se definen como un tipo IN, los de salida como OUT y los bidireccionales como INOUT.
3. El tipo de datos que maneja el puerto: En los modelos VHDL hay que definir el tipo de datos de los objetos. La elección del tipo de datos es muy importante, pues determina el conjunto de valores que puede tomar el objeto declarado, así como las operaciones (lógicas, aritméticas y de conversión de tipos) que se le pueden aplicar, cuestión fundamental a la hora de describir el funcionamiento de los dispositivos.
4. Cuerpos de arquitectura. Nociones básicas.

La sintaxis básica de un cuerpo de arquitectura es la siguiente:



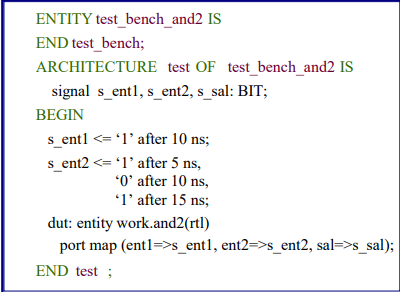
Las dos etiquetas que forman parte de la cabecera nombran a la propia arquitectura y a la declaración de entidad a la que está asociada. La primera vuelve a parecer en la línea que cierra el cuerpo de arquitectura.

Por lo demás, se distinguen dos áreas para la inclusión de código con dos propósitos distintos:}

1. La comprendida entre la cabecera y la palabra BEGIN está destinada a la declaración de objetos que se precisen para realizar la descripción del funcionamiento del dispositivo.
2. Entre la palabra BEGIN y END {arquitectura} se describe el funcionamiento.
3. Simulación del modelo VHDL.

La compilación exitosa de la declaración de entidad y la arquitectura VHDL garantiza que se cumple una serie de reglas sintácticas y semánticas, pero no que el hardware se haya modelado correctamente. Para probar el modelo es preciso simularlo.

Una simulación VHDL se realiza conectando el modelo en pruebas a un conjunto de estímulos que permite observar la respuesta del mismo. Para ello hay que construir un test-bench (banco de test).



Como puede verse, consta, al igual que el modelo de un dispositivo, de una declaración de entidad y un cuerpo de arquitectura, pero con las siguientes peculiaridades:

1. La declaración de entidad no tiene puertos.
2. En el cuerpo de arquitectura no se describe el funcionamiento de un dispositivo, sino que se conecta el dispositivo a probar a un conjunto de señales (estímulos y salidas).

El conjunto de valores que toman los estímulos a lo largo del tiempo constituyen los vectores de test que se aplican al modelo en pruebas.

Un simulador VHDL es capaz de ejecutar test-beches disponiendo normalmente, de un conjunto de utilidades que facilitan la depuración de los modelos y la revisión de resultados.

1. Unidades de diseño y librerías VHDL.

Un modelo VHDL se compone de un conjunto de unidades de diseño. Una unidad de diseño es la mínima sección de código compilable separadamente. Es un concepto muy simple, puede entenderse recurriendo a conocimientos básicos sobre lenguaje de programación: no se puede editar un fichero, escribir únicamente una sentencia IF o CASE y compilar el fichero, porque una sentencia no es una unidad de código compilable. Podrá compilarse un programa, una función o un conjunto de declaraciones, o cualquier cosa que el lenguaje especifique.

Las unidades de diseño VHDL se construyen combinando construcciones del lenguaje (sentencias, declaraciones, etc.). En VHDL existen cinco tipos distintos de unidades de diseño, cada una de las cuales está pensada para el desempeño de una determinada función en el modelado del hardware. Las unidades se clasifican en primarias y secundarias (de acuerdo con las relaciones de dependencia jerárquica que mantienen entre sí: una unidad secundaria esta asociada siempre a una unidad primaria), y son:

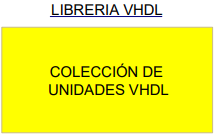
1. La declaración de entidad: Es la unidad primaria del lenguaje que identifica a los dispositivos, de finiendo su interfaz (nombre, terminales de conexión y parámetros de instancia miento). Puede decirse (simplificando bastantemente) que desempeña una función equivalente a la del símbolo de un dispositivo en los esquemas de circuitos.
2. El cuerpo de arquitectura: Es una unidad secundaria del lenguaje. Está asociado a una determinada declaración de entidad, describiendo el funcionamiento lógico del dispositivo identificado por esta. Aporta una información equivalente a la de las tablas de verdad o los cronogramas de los circuitos lógicos.
3. La declaración de paquete: Los paquetes desempeñan en VHDL funcione similares a las de las librerías en lenguajes de programación de alto nivel. La declaración de paquete es una unidad primaria que contiene la “vista pública” de los paquetes.
4. El cuerpo del paquete: Es una unidad secundaria asociada a una declaración de paquete. Se utiliza, si resulta necesario, para definir los elementos declarados en este.
5. La declaración de configuración: Es una unidad primaria que sirve para manejar el emplazamiento de componentes en modelos estructurales.

Las más importantes, porque son las únicas que siempre deben existir en cualquier modelo hardware y exigen un mayor esfuerzo de desarrollo durante las tareas de diseño, son las dos primarias: la declaración de entidad y el cuerpo de arquitectura. También son muy importantes las unidades con que se construyen los paquetes VHDL (la declaración y el cuerpo de paquete). Los paquetes VHDL, como ya se ha dicho, realizan una función equivalente a la de las librerías en los lenguajes de programación, es decir, son unidades de código cuyo contenido puede utilizarse desde otras unidades mediante una serie de cláusulas de visibilidad (como #include en lenguaje C).

La quinta unidad de código, la declaración de configuración, se utiliza rara vez en modelos de baja mediana complejidad y su conocimiento no resulta imprescindible para iniciarse en el lenguaje.

Las unidades de diseño VHDL se almacenan en librerías VHDL; dicho de otra manera: una librería VHDL es una colección de unidades de diseño. Una unidad queda almacenada en una librería cuando ha sido compilada correctamente. Las librerías VHDL tienen un nombre lógico que permite identificarlas.

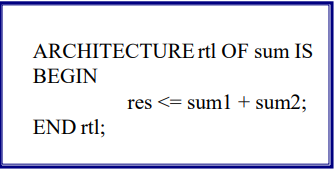
Es importante no confundir las librerías VHDL con las de los lenguajes de programación: como ya se ha dicho, el equivalente a esta ultimas en VHDL son los paquetes.



Las unidades de diseño almacenadas en una misma librería deben cumplir una serie de normas:

1. Una unidad primaria y cualquier unidad secundaria asociada deben almacenarse en las mismas librerías. Además, primero debe compilarse la unidad primaria (declaración de entidad o declaración de paquete) y después la secundaria (cuerpo de arquitectura o paquete).

Que sea necesario compilar antes la unidad primaria que cualquier secundaria asociada viene dado porque en esta última puede utilizarse cualquier objeto visible o declarado en la primaria. La necesidad resulta entonces evidente: la compilación de un cuerpo de arquitectura.



La segunda regla se debe a esta misma razón:

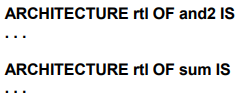
1. Si una unidad de diseño tiene visibilidad sobre otra, esta tiene que compilarse antes que la primaria.

Esto afecta fundamentalmente a los paquetes. Antes de compilar una unidad que utilice un paquete, la declaración y el cuerpo de paquete deben haberse almacenando en una librería.

La ultima regla establece algunas condiciones sobre los nombres de las unidades:

1. No pueden existir dos unidades primarias con el mismo nombre en una librería. Si puede haber de hecho ocurre con frecuencia, unidades secundarias con el mismo nombre, solo cuerpos de arquitectura, porque los de paquete no tienen nombre, la única condición que debe darse en este caso es que estén asociados a distintas unidades primarias.

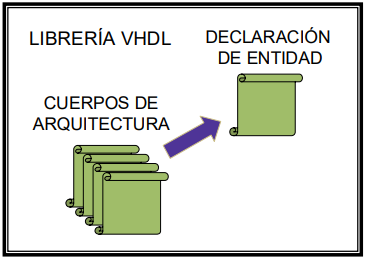
El hecho de que los cuerpos de arquitectura tengan el mismo nombre no ocasiona ningún problema, puesto que se distinguen por la declaración de entidad cuyo comportamiento describen:



Para determinar con las cuestiones relativas a las unidades de diseño y las librerías hay que mencionar tres detalles importantes:

* Una declaración de entidad puede tener asociados varios, cuerpos de arquitectura.

El lenguaje VHDL admite la posibilidad de que resulte necesario tener que describir de varias maneras el comportamiento de un dispositivo con el objeto, fundamentalmente, de que se puede disponer de versiones que lo modelen con distintos grados de abstracción, por ello una declaración de entidad puede tener asociado cualquier número de cuerpos de arquitectura.



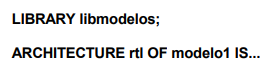
* Un paquete puede estar formado únicamente por una declaración de paquete, es decir, el cuerpo de paquete puede no existir.

Esto es debido a que los cuerpos de paquete suelen contener la especificación de subprogramas en realidad pueden contener más cosas cuya declaración aparece en la declaración de paquete. Cuando un paquete no contiene procedimientos o funciones puede que no exista el cuerpo de paquete.

* Como ya se ha indicado anteriormente, cada librería VHDL tiene un nombre lógico. Además, la librería de trabajo que se esté utilizando en un determinado momento puede identificarse mediante la palabra WORK.

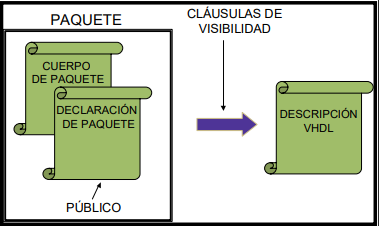
1. Cláusulas de visibilidad

Los modelos hardware y el contenido de los paquetes VHDL almacenados en una determinada librería pueden utilizarse en unidades de diseño de otras. Para ello es preciso utilizar cláusulas de visibilidad sobre las unidades que componen una librería se obtiene mediante la cláusula LIBRARY. Por ejemplo:

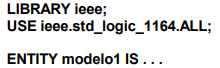


Las clusulas de visibilidad se usan muchas veces para poder utilizar los objetos y subprogramas declarados en los paquetes. Para poder utilizar el contenido de un paquete resulta ser necesario

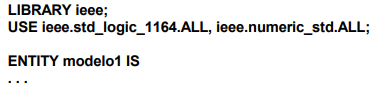
* Obtener visibilidad sobre la librería donde esta almacenando.
* Obtener visibilidad sobre los contenidos del propio paquete.



En el desarrollo de modelos VHDL resulta muy frecuente el uso de los paquetes de la librería IEEE. Dos del paquete de esa librería se llaman std\_logic\_1164 y numeric\_std.



Permitiría el uso del contenido del paquete std\_logic\_1164 en la declaración de entidad modelo 1 y en cualquiera de los cuerpos de arquitectura que tenga asociados (debido a las reglas de herencia de visibilidad entre unidades primarias y secundarias).



Daría acceso al contenido de los dos paquetes.

Dentro del paquete std\_logic\_1164 se declara un tipo de datos llamado std\_ulogic. Si solo queremos obtener visibilidad sobre ese objeto, podríamos utilizar la fórmula:



Hay algunas excepciones a las reglas de visibilidad descritas:

1. La librería WORK siempre es visible.

Esto quiere decir que una clausula (por lo demás, perfectamente validad) como:



Es superflua. Sin embargo, si existiera un paquete (PARKWORK, por ejemplo) en la librería WORK cuyo contenido se deseará utilizar, si sería necesario utilizar la fórmula:



1. La librería STD y el contenido del paquete STANDARD son siempre visibles.

La librería STD contiene los paquetes STANDARD y TEXTIO; en realidad estos paquetes forman parte del lenguaje, ya que contienen la declaración de los tipos de datos predefinidos y las operaciones de entrada-salida. La visibilidad sobre el primero es automática; para utilizar el paquete TEXTIO hay que hacer uso de la cláusula USE.

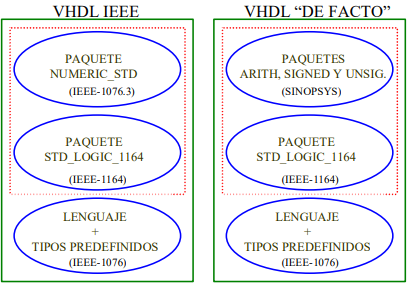
1. Nombres VHDL

Esta es la manera en que se construyen los nombres en VHDL y permite, por ejemplo, que se puede utilizar el mismo nombre para dos unidades de diseño situadas en distintas librerías sin q ue existe ambigüedad a la hora de identificarlos.

1. Estructura básica del lenguaje VHDL.

Las unidades de diseño VHDL se realizan componiendo código con las construcciones que se definen en el lenguaje: cabeceras de unidades, declaraciones de los objetos, sentencias, etc. Todas ellas están definidas en la norma IEEE-1176. Los diseñadores del lenguaje tomaron la decisión de que los tipos de datos predefinidos (INTEGER, REAL, etc.) y las operaciones de entrada-salida sobre ficheros se declaran en paquetes VHDL: Los paquetes STANDARD y TEXTIO de la librería STD. Debido a que la visibilidad sobre estos paquetes es automática, puede trabajarse con ellos obviando esta curiosa circunstancia.

Cualquier entorno VHDL standard actual incorpora, además, los paquetes de la librería IEEE. Estos paquetes se crearon para dar soporte a las necesidades descriptivas que surgen en los modelos VHDL orientados a la operación lógica automática. En ellos se define nuevos tipos de datos, operaciones lógicas y aritméticas para esos tipos y funciones de conversación entre tipos. Existe un problema con estos paquetes: hay dos versiones, la normalizada por IEEE y el estándar de facto realizado por Synopsys. La mayoría parte de los entornos utiliza el segundo, en algunos, como VERIBEST, el usuario puede elegir entre ambos. Ambas versiones comparten un paquete (el llamado ieee.std\_logic\_1164), el resto son distintos.



Es preferible elegir la versión de Synopsys, porque es la única que soporta la mayoría de los simuladores y sintetizadores VHDL.

Además, algunos entornos proporcionan paquetes desarrollados por ellos mismos, sus usos pueden acarrear problemas de portabilidad, por lo que debe, en la medida de lo posible evitarse.