Arquitectura de Bus para Protección contra Hardware Trojan en Chips de Seguridad

Según Changlong et al. (2011), en el artículo "A System-On-Chip bus architecture for hardware Trojan protection in security chips"[1], se propone una arquitectura de bus Segura que protegerá al usuario de Hardware Troyano, además nos indica lo siguiente:

Los circuitos integrados actuales, dependen aun más de núcleos IP de terceros y de desarrollos externos para su desarrollo. Esta es la razón que los vuelve muy vulnerables a modificaciones maliciosas, conocidas como "Hardware Trojans", las cuales pueden ser el motivo de las filtraciones de datos o fallos extremos en el sistema.

Los Hardware Troyanos se clasifican dependiendo de sus características físicas, método de activación y accionamiento. Sin embargo, existen métodos para detectarnos antes de que logren hacer algún daño, aunque siguen siendo muy peligrosos.

En el área de chips de seguridad, si un virus troyano contamina el diseño o fabricación de este, puede evitar casi todas las medidas de seguridad, y extraer información como usuarios y contraseñas, y a veces sin ser detectados.

Antes de que el árbitro del bus permita la comunicación entre un maestro y un esclavo, se genera un número aleatorio mediante un Generador de Números Aleatorios (RNG). Este número se envía simultáneamente a ambos dispositivos para sincronizar la transmisión. Los datos reales se combinan con pseudo-datos, de forma que solo el receptor autorizado, usando el mismo número aleatorio, pueda extraer la información auténtica.

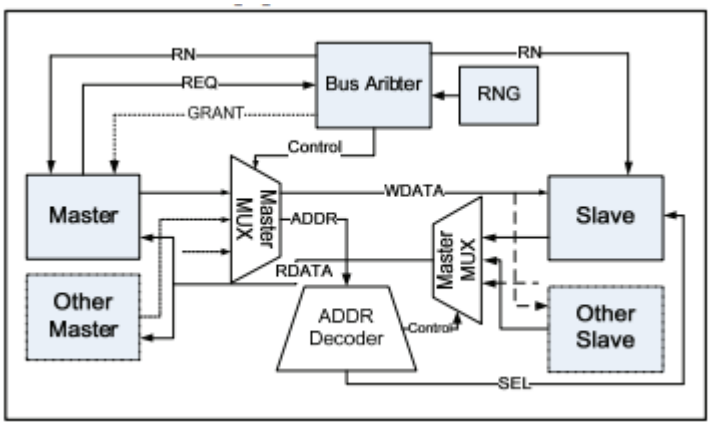


Figura 1 La estructura de la arquitectura de bus propuesta.

El RNG tiene una estructura híbrida compuesta por una Fuente Aleatoria Analógica (ARS), que emplea osciladores de alta frecuencia no correlacionados y un muestreo con reloj principal de baja frecuencia con jitter. También incluye un Registro de Corrimiento con Retroalimentación Lineal (LFSR), que utiliza un polinomio característico específico y posee un periodo de $2^{128} - 1$.

Esta arquitectura busca impedir que un Troyano pueda interceptar y filtrar la comunicación interna del chip.

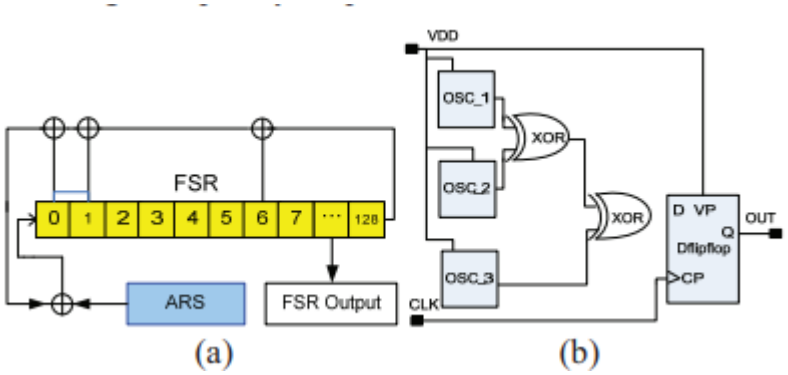


Figura 2 (a) La estructura del RNG-(b) El diagrama de bloques detallado del ARS.

Bus de componentes

En la conferencia dada por M. Yasuda et al. (2002). Titulada "A top-down hardware/software co-simulation method for embedded systems based upon a component logical bus architecture" [2], nos indica lo siguiente.

La arquitectura lógica de bus de componentes se conforma por tres tipos de señales. Señales del bus del sistema, señales de puerto y señales de interrupción. El objetivo de esta arquitectura es servir de interfaz entre componentes de software y hardware.

Señales del bus del sistema

Incluyen dirección, datos y control. Los datos se leen o escriben según la señal de control y la dirección. El espacio de direcciones se divide en espacio de memoria y espacio de entrada/salida (I/O). La memoria de los componentes de software se asigna en el espacio de memoria, mientras que los puertos de software y los registros de hardware están en el espacio de I/O mapeado en memoria.

Señales de puerto

Son señales específicas que permiten la comunicación directa entre componentes de software y hardware. Los puertos del software también se asignan en el espacio de I/O.

Señales de interrupción

Se usan para notificar a un componente de software cuando un componente de hardware o software ha completado una operación o ha detectado un error.

El intercambio de datos ocurre mediante la escritura en puertos y registros del hardware por parte del software para iniciar operaciones y transferir datos. El hardware notifica la finalización mediante interrupciones o estableciendo una bandera de estado. El software lee los puertos y registros para obtener datos e información de estado.

Bus puente

Segun T. Wang et al. (2008) en el articulo "A Hardware Implement of Bus Bridge Based on Single CPU and Dual Bus Architecture" [3], describe la función y estructura del bus Puente.

Las arquitecturas tradicionales utilizan una sola CPU y un solo bus, como la arquitectura de Von Neumann. Esta arquitectura es escasa en los ordenadores de red los cuales son exigentes en los métodos de mantener seguros sus datos. Además, la red y el almacenamiento local están conectados al mismo bus, esto representa un peligro, ya que con software malintencionado los datos de ese almacenamiento podrían ser robados y llegan a controlar el bus del sistema.

Arquitectura mínima del sistema de hardware

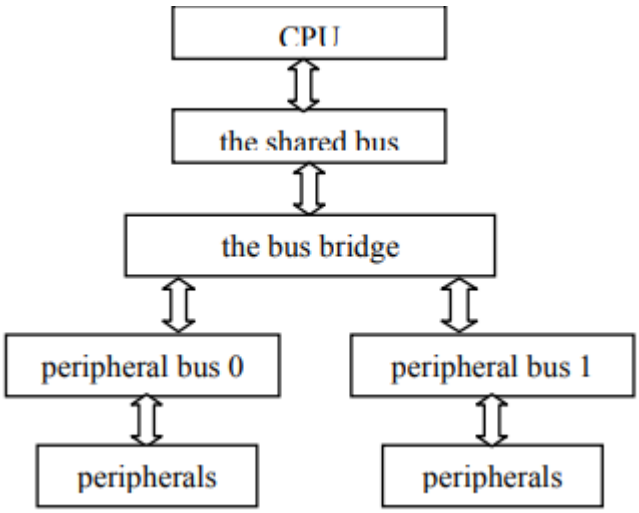


Figura 3 Arquitectura mínima del sistema de hardware

Como se puede observar en la figura 3 el sistema consta de cuatro partes: CPU, buses, el puente de bus y los periféricos. El bus puente se conecta a la CPU a través del bus compartido. Los buses periféricos se conectan a los puertos maestros del bus puente.

Además, los dos buses periféricos están aislados y solo uno de ellos contiene el módulo del CPU y puede conectarse a los buses compartidos.

Función del bus puente

La función del bus puente es conectar el bus compartido con uno de los buses periféricos. El bus bridge debe asegurar que el bus compartido pueda conectarse en cualquier momento a uno de los buses periféricos. Además, debe garantizar que solo un bus periférico esté conectado al bus compartido simultáneamente. Cuando la CPU necesita cambiar de bus periférico, el bus bridge corta la conexión actual y conecta el bus compartido con el otro bus periférico.

Antes del diseño del bus puente, se definen dos términos importantes. El área de trabajo indica cuál bus periférico está conectado actualmente al bus compartido. Si el bus periférico 0 está conectado, el sistema trabaja en el área 0; si es el bus periférico 1, entonces trabaja en el área 1. Las señales correspondientes son aquellas con la misma definición en el bus compartido y en los buses periféricos, pero con nombres distintos para diferenciarlas. Por ejemplo, la señal write_n en el bus compartido corresponde a ava0_write_n en el bus periférico 0 y ava1_write_n en el bus periférico 1.

Estructura del bus puente

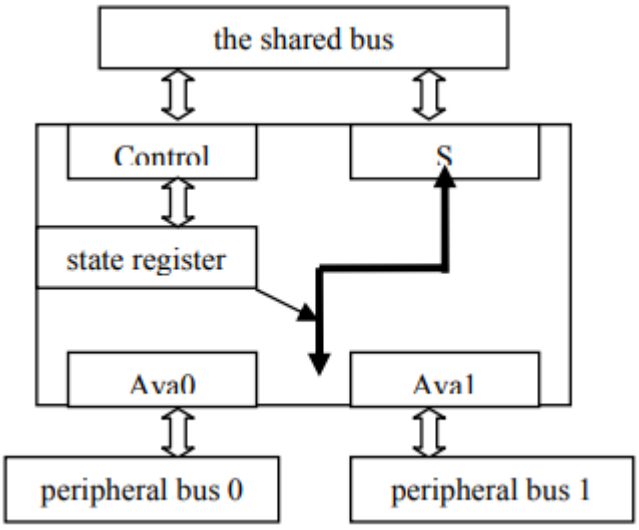


Figura 4 Estructura del bus puente

La estructura del bus bridge incluye dos puertos esclavos llamados Control y S, dos puertos maestros denominados Ava0 y Ava1, y un registro de estado de cinco bits. El puerto Control recibe instrucciones del CPU para el bus bridge y envía los datos del registro de estado al CPU. El puerto S recibe y transmite todas las señales Avalon del bus

compartido. Estas señales se conectan con sus señales correspondientes en los puertos Ava0 o Ava1, que están vinculados a los buses periféricos.

El registro de estado tiene cinco bits: los cuatro bits inferiores almacenan los modos de trabajo, y el bit superior, llamado bit bandera, indica qué bus periférico está conectado actualmente al bus compartido a través del bus bridge. Cuando el bit bandera está en 0, todas las señales del puerto S se conectan con las del puerto Ava0, por lo que el sistema opera en el área 0. Si el bit bandera está en 1, el sistema opera en el área 1 conectando el bus compartido con el bus periférico 1.

Bus de direcciones DDR4

Según K. Pandey (2018), en el artículo: "Signal and power integrity analysis of DDR4 address bus of onboard memory module" [4], explica que para garantizar la integridad de señal y potencia en el bus de direcciones DDR4, es fundamental usar una topología "fly-by" y terminaciones adecuadas para reducir ruido y reflexiones, asegurando una transmisión confiable a altas velocidades.

En el diseño de memorias DDR4, el bus de direcciones es fundamental para transmitir la información con precisión a altas velocidades, que pueden llegar hasta 3.2 Gbps. Para manejar estas velocidades, se utiliza una topología llamada "fly-by", donde la señal de dirección pasa en cadena desde el controlador hacia cada módulo de memoria, lo que ayuda a reducir problemas como las reflexiones y el ruido. Además, en el extremo final del bus se coloca una terminación pull-up de 40 ohmios que mejora la integridad de la señal.

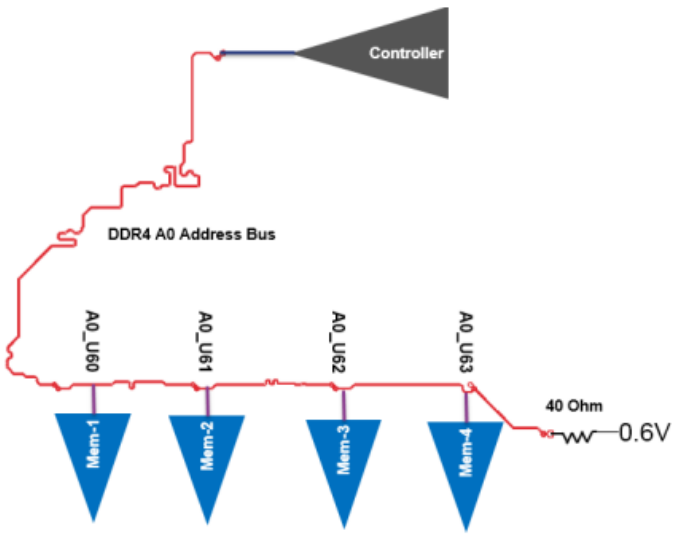


Figura 5 Topología de buses de dirección en memoria DDR4

Cada línea en el bus de direcciones representa un bit, por lo que la cantidad de líneas determina la capacidad máxima de memoria que puede direccionarse directamente. El diseño del enrutamiento es crucial; se recomienda que las señales de reloj diferencial, dirección, control y comando estén en capas cercanas para minimizar interferencias y garantizar una buena calidad de señal.

El ruido generado por la conmutación simultánea de varias líneas, conocido como ruido SSN, puede afectar negativamente la señal, por lo que se emplean terminaciones pasivas, como resistencias en serie y paralelo, para controlar estas interferencias. Además, el plano de potencia debe diseñarse cuidadosamente para mantener la impedancia dentro de límites adecuados y reducir el ruido eléctrico.

Para evaluar el desempeño del bus de direcciones, se usan simulaciones con modelos IBIS que permiten analizar el comportamiento dinámico de la señal y generan diagramas de ojo que muestran la calidad en diferentes módulos de memoria. Estos análisis ayudan a identificar el punto más débil en la cadena de memoria, asegurando que el sistema funcione de forma estable y eficiente.

Buses de serie Universal

según L. Ramadoss y J. Y. Hung (2008), en el artículo: “A study on universal serial bus latency in a real-time control system” [5], nos explica lo siguiente acerca del bus de serie universal.

Cuando se conecta un bus de serie universal (USB) al computador, el sistema operativo automáticamente lo detecta y se encarga de configurarlo y el controlador USB detecta las características del dispositivo a través de un dispositivo descriptor.

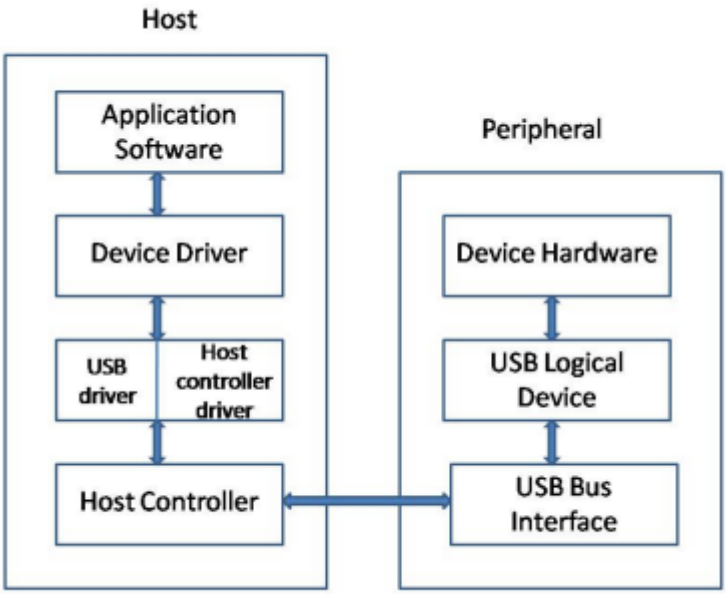


Figura 6 Comunicación USB

Un dispositivo descriptor es una estructura de datos la cual contiene información sobre las propiedades de los dispositivos conectados. Inmediatamente el controlador USB localiza el driver del dispositivo y lo carga o actualiza. Una vez realizado esto establece una conexión lógica llamada canalización entre el driver y el punto final. El punto final es el registro por donde entran y salen datos del dispositivo.

Comunicación del bus de serie universal

La comunicación USB se basa en una relación maestro-esclavo, donde la computadora actúa como maestro y los dispositivos USB como esclavos. Solo puede haber un maestro en el sistema y los dispositivos USB no se comunican directamente entre sí, sino siempre a través del host.

El software del sistema USB incluye dos componentes principales: el controlador USB y el controlador del host. Cuando un dispositivo USB necesita enviar o recibir datos, el controlador del dispositivo envía una solicitud de transferencia al controlador USB, llamada paquete de solicitud de entrada/salida (IRP).

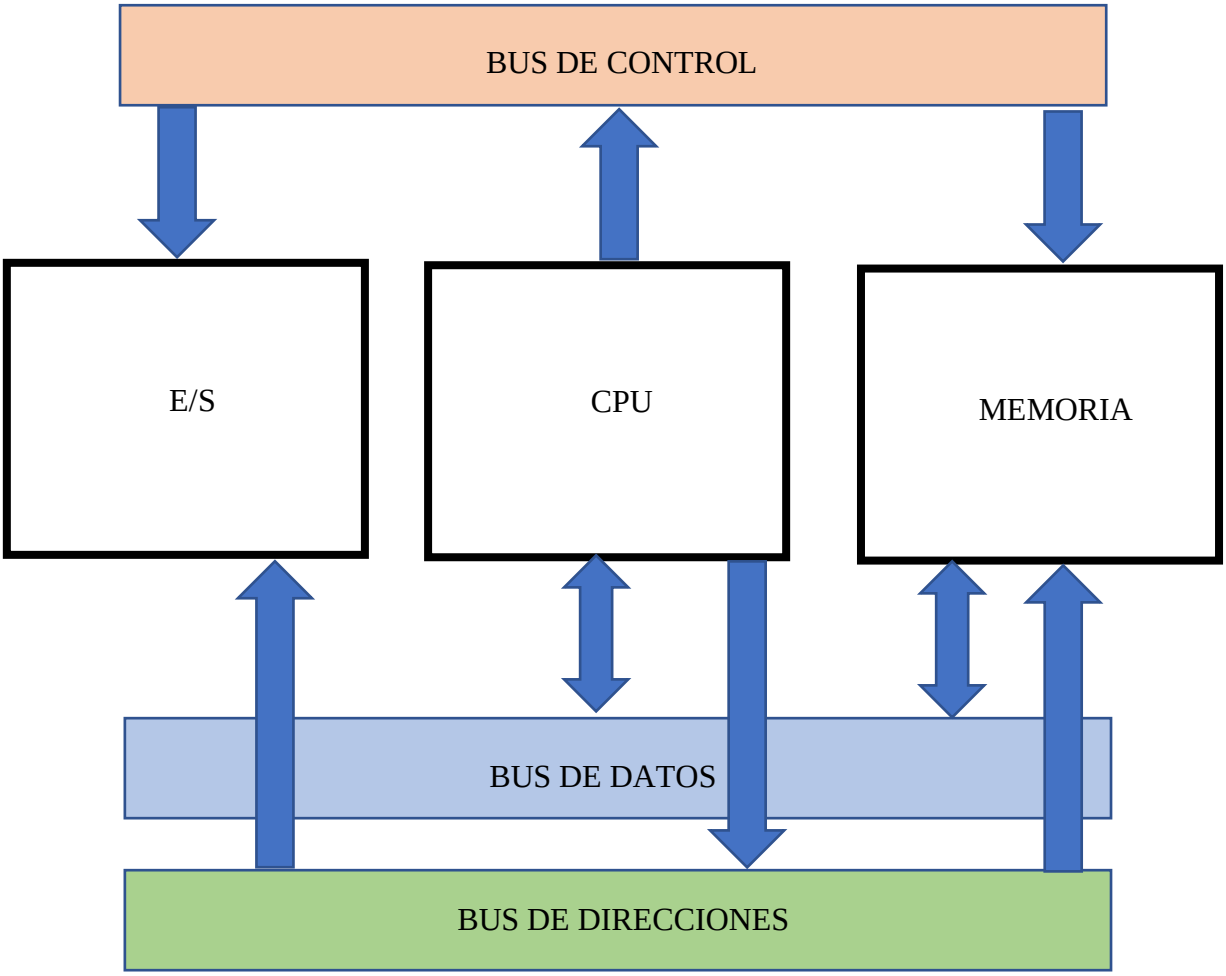
Luego, el controlador del dispositivo proporciona un área de memoria para almacenar los datos durante la transferencia. El controlador USB divide esa solicitud en varias transacciones pequeñas que se ejecutan en intervalos de 1 ms. Estas transacciones se organizan según las necesidades del dispositivo y la capacidad del USB.

PROCEDIMIENTOS

Realizar diagramas que representen la estructura y función de un bus.

Un bus es un conjunto de líneas o canales de comunicación que trasmiten información, direcciones y señales trasportándola mediante estos canales entre los componentes del sistema como la CPU, memoria, periféricos, entre otros.

Diagrama de buses en arquitectura de computadoras



Si el CPU quisiera leer un dato en la memoria se daría el siguiente proceso:

1. La CPU coloca la dirección a leer en el Bus de Direcciones.
2. La CPU envía la señal de lectura por el Bus de Control.
3. La memoria reconoce la dirección y la señal de lectura.
4. La memoria pone el dato solicitado en el Bus de Datos.
5. La CPU lee el dato desde el Bus de Datos.

Identificar y describir los diferentes tipos de buses y su impacto en la velocidad de transferencia.

A continuación, se muestra un cuadro comparativo con los tipos de buses, su descripción, ventajas y desventajas y como impacta a la velocidad de transferencia.

Tipo de Bus	Descripción	Ventajas	Desventajas	Impacto en la Velocidad de Transferencia

Bus de Datos	Transporta los datos entre componentes.	Mayor ancho permite transferir más bits a la vez.	Mayor complejidad con más líneas.	Ancho mayor = mayor cantidad de datos transferidos simultáneamente, aumenta la velocidad.
Bus de Direcciones	Transporta las direcciones de memoria o dispositivos.	Permite direccionar gran cantidad de memoria.	No afecta directamente la velocidad de datos.	Limita la cantidad máxima de memoria direccionable, no impacta velocidad directa.
Bus de Control	Señales para controlar y sincronizar la transferencia.	Controla la coordinación y sincronización.	Requiere buena gestión para evitar retrasos.	Afecta la sincronización y timing, impactando indirectamente la velocidad.
Bus Paralelo	Varias líneas transportan bits simultáneamente.	Transferencia rápida, envía muchos bits a la vez.	Limitado a distancias cortas; interferencias y costos elevados.	Alta velocidad por transferencia simultánea, pero menos eficiente a largas distancias.
Bus Serie	Bits enviados uno tras otro por pocas líneas.	Menor interferencia, permite distancias largas.	Velocidad menor para grandes volúmenes de datos.	Velocidad menor comparado con bus paralelo en volumen, pero más estable y confiable en distancias.

Bibliografía

[1] L. Changlong, Z. Yiqiang, S. Yafeng, and G. Xingbo, “A System-On-Chip bus architecture for hardware Trojan protection in security chips,” in *2011 IEEE International Conference of Electron Devices and Solid-State Circuits*, IEEE, Nov. 2011, pp. 1–2. doi: 10.1109/EDSSC.2011.6117727.

[2] M. Yasuda, K. Seo, H. Koizumi, B. Shackleford, and F. Suzuki, “A top-down hardware/software co-simulation method for embedded systems based upon a component logical bus architecture,” in *Proceedings of 1998 Asia and South Pacific Design Automation Conference*, IEEE, pp. 169–175. doi: 10.1109/ASPDAC.1998.669438.

[3] T. Wang, F. Shao, R. Sun, and H. Huang, “A Hardware Implement of Bus Bridge Based on Single CPU and Dual Bus Architecture,” in *2008 International Symposium on Computer Science and Computational Technology*, IEEE, 2008, pp. 17–20. doi: 10.1109/ISCSCT.2008.106.

[4] A. K. Pandey, “Signal and power integrity analysis of DDR4 address bus of onboard memory module,” in *2018 IEEE Electrical Design of Advanced Packaging and Systems Symposium (EDAPS)*, IEEE, Dec. 2018, pp. 1–3. doi: 10.1109/EDAPS.2018.8680896.

[5] L. Ramadoss and J. Y. Hung, “A study on universal serial bus latency in a real-time control system,” in *2008 34th Annual Conference of IEEE Industrial Electronics*, IEEE, Nov. 2008, pp. 67–72. doi: 10.1109/IECON.2008.4757930.