Czas na przerwanie

Obsługa przerwań jest jedną z kluczowych operacji, z jaką musi poradzić sobie system operacyjny. Implementacja przerwań w Linuksie jest zależna od architektury całego systemu, a w szczególności od procesora, kontrolera/kontrolerów przerwań i sposobu ich połączenia. Niniejszy artykuł podzielony jest na dwie części. W pierwszej krótko opisano popularnie stosowany w systemach ARMowych kontroler przerwań GIC. Druga natomiast traktuje o obsłudze przerwań w Linuksie, przybliżając takie zagadnienia jak: domeny przerwań, połączenia kaskadowe, rozdzielenie obsługi przerwania na części (tzw. top, bottom halves), a także trasowanie przerwań. W artykule przybliżono zarówno mechanizmy generyczne, jak i zależne od architektury – skupiono się głównie na SoCach ARM/ARM64 oraz kontrolerach przerwań typu GIC.

KONTROLERY PRZERWAŃ – GICvX

Firma ARM ciągle ulepsza i rozszerza listę oferowanych produktów. Tak jest również w przypadku kontrolerów przerwań GIC (ang. *Generic Interrupt Controller*), których zadaniem jest zarządzanie, priorytetyzowanie oraz trasowanie (ang. *routing*) przerwań. Architektura GICa ewoluowała z wersji GICv1 do najnowszej GICv4. Poniżej przedstawiono nowe funkcje wprowadzane do kolejnych wersji wspomnianych kontrolerów.

Wersja	Funkcjonalność	Typowo używane w
GICv1	+ Obsługa do 8 PE* + Obsługa do 1020 przerwań (*interrupt ID*) + Dwa poziomy zabezpieczeń (ang. *security state*)	ARM Cortex-A9 MPCore
GICv2	Funkcjonalność GICv1 + Wsparcie dla wirtualizacji	ARM Cortex-A53 MPCore ARM Cortex-A57 MPCore
GICv3	Funkcjonalność GICv2 + Obsługa powyżej 8 PE* (126 GIC-500, 512 GIC-600) + Wsparcie dla przerwań opartych na wiadomościach (*Message-based interrupt*) + Obsługa powyżej 1020 przerwań (zależne od implementacji) + Dostęp do tzw. *CPU interface* poprzez rejestry systemowe + Ulepszony model zabezpieczeń (rozdzielenie grupy 1 na *Secure* i *Non-secure*)	ARM Cortex-A57 MPCore ARM Cortex-A72 MPCore
GICv4	Funkcjonalność GICv3 + Bezpośrednie wstrzykiwanie przerwań wirtualnych	ARM Cortex-A57 MPCore ARM Cortex-A72 MPCore

Tabela 1.Funkcjonalność kontrolerów GIC w zależności od wersji

**PE (ang. Processing Element) – w nomenklaturze ARMowej termin odnoszący się do jednostki przetwarzającej, w szczególności identyfikowany z rdzeniem.*

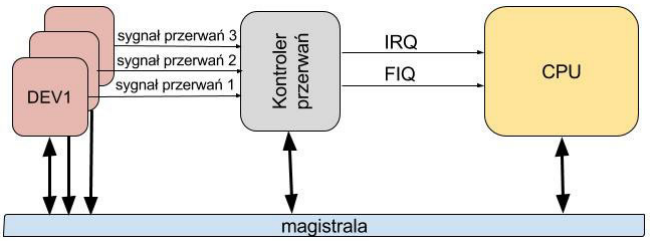
Przykładowe implementacje poszczególnych architektur to:

- GIC-600 (implementuje architekturę GICv3)
- GIC-500 (implementuje architekturę GICv3)
- GIC-400 (implementuje architekturę GICv2)

Oprócz wersji wymienionych w powyższej tabeli firma ARM wypuściła na rynek wersję pośrednią GICv2m, która rozszerza GICv2 o wsparcie dla przerwań bazujących na wiadomościach (*ang. message based interrupts*). Ciekawostką jest fakt, że wersja ta jest jedną z wymaganych opcji, którą musi spełniać serwer ARMowy, aby zadeklarować jego zgodność ze specyfikacją SBSA (*ang. Server Base System Architecture*) na poziomie pierwszym. Niniejszy artykuł nie traktuje o serwerach ARMowych, ale zainteresowany czytelnik może sięgnąć do wspomnianej specyfikacji (odnośnik do specyfikacji SBSA w sekcji „W sieci").

Jak sygnalizowane są przerwania

Tradycyjnie w systemach komputerowych przerwania od poszczególnych peryferiów do kontrolera przerwań sygnalizowane były przy użyciu dedykowanych linii.



Rysunek 1. Sygnalizacja przerwań przy użyciu dedykowanych linii

Z uwagi na fakt, że z biegiem czasu w systemach komputerowych stosowano coraz więcej komponentów, liczba fizycznych połączeń potrzebnych do realizacji przerwań wzrastała. Sytuacja taka sprawiała problemy projektantom zwłaszcza w systemach, w których liczba przerwań dochodziła do setki lub nawet przekraczała ją. W celu rozwiązania tej niedogodności kontroler GIC od wersji 3 (w zasadzie od wersji v2m) dodał wsparcie dla przerwań bazujących na wiadomościach. Są one generowane i zatwierdzane poprzez wpisy do odpowiednich rejestrów.