

	TITLE Pułapki i sterowanie przerwaniem RV32I										VER.	v00.01		DATE/TIME	2026-07-20T00:00:00+02:00										
	PROJECT Freestanding C na RV32I										SERIES Freestanding RV32I and K&R		CARD 10/13		SHEET 1/8										
	SUBJ. Inf.										PROG. -		CORE -		SCOPE -			LEVEL -		POS. -		GITEA URL https://zsl-gitea.mpabi.pl/edu-inf/lab-rv32i-c-interrupts		ec04ecce-1c4d-52ba-a2a2-45f6a4707056	
	GITEA https://zsl-gitea.mpabi.pl/edu-inf/lab-rv32i-c-interrupts																								
	HTML https://dce7fb9d-7b2f-5d49-96a2-3a30d3070b84.mpabi.pl/8fb9c110-557b-5081-8ae2-a56800f872e8																								

Cel karty

Uczeń odróżnia synchroniczny wyjątek od asynchronicznego przerwania, wyprowadza adres wznowienia z mepc oraz dowodzi, że samo pending nie wystarcza bez indywidualnej i globalnej bramki enable.

Zakres i zachowane przykłady

Trzy obrazy RV32I działają na rzeczywistym modelu Hazard3. Task01 wykonuje ecall; Task02 używa kontrolowanego machine software interrupt; Task03 mierzy macierz pending, mie.MSIE i mstatus.MIE. Peryferia pojawiają się dopiero w C12 i C13.

Układ każdego przykładu: pełny profil A1–A8; widok N/D ma jawny powód, a diagram nie jest wymagany, gdy tekst daje lepszy dowód.

Typ	Task	Idea	Waga
ECALL	Task01	synchroniczny cause 11, mepc i mret	kluczowe
MSIP	Task02	MSIP, MSIE, MIE, vector 3 i mret	kluczowe
GATE	Task03	to samo źródło, dwie niezależne bramki	kluczowe

		TITLE Pułapki i sterowanie przerwaniem RV32I										VER. v00.01		DATE/TIME 2026-07-20T00:00:00+02:00					
		PROJECT Freestanding C na RV32I										SERIES Freestanding RV32I and K&R		CARD 10/13				SHEET 2/8	
URL		Inf.		PRG.		CORE		SCOPE		LEVEL		POS.		GITEA UUID 8fb9c110-557b-5081-8ae2-a56800f872e8		ec04ecce-1c4d-523baa62-234f9-40700e		NOTHUR W. PAB182C3E	
GITEA https://zsl-gitea.mpabi.pl/edu-inf/lab-rv32i-c-interrupts																			
HTML https://dce7fb9d-7b2f-5d49-96a2-3a30d3070b84.mpabi.pl/8fb9c110-557b-5081-8ae2-a56800f872e8																			

1 1 — Wyjątek synchroniczny wskazuje instrukcję

`ecall` jest wykonywany przez bieżący strumień instrukcji, więc `mcause=11` nie ma ustawionego bitu interrupt, a `mepc` wskazuje samo `ecall`. Handler `Task01` świadomie zapisuje `mepc+4`; bez tego `mret` wykonałby ponownie tę samą instrukcję. Taka polityka jest poprawna dla tego kontrolowanego ECALL, ale nie wolno automatycznie pomijać instrukcji przy każdym błędzie.

2 2 — Software IRQ jest asynchronicznym źródłem poziomowym

Testbench ustawia MSIP hart0, a rdzeń widzi `mip.MSIP`. Dostarczenie wymaga dodatkowo `mie.MSIE=1` i `mstatus.MIE=1`. Wektor 3 prowadzi do `isr_machine_softirq`; handler odczytuje `mcause=0x80000003`, usuwa źródło i dopiero potem publikuje licznik. `mret` wraca do przerwanej instrukcji, a nie do instrukcji wybranej ręcznie przez aplikację.

3 3 — Pending i enable odpowiadają na inne pytania

Pending mówi, że źródło żąda obsługi. `mie.MSIE` dopuszcza konkretną klasę, a `mstatus.MIE` otwiera globalną bramkę machine mode. `Task03` utrzymuje to samo pending i kolejno sprawdza dwa stany z jedną zamkniętą bramką oraz stan z obiema otwartymi. Dzięki temu kod 0 dowodzi całej macierzy, a nie tylko tego, że ISR kiedyś się wykonał.

	TITLE Pułapki i sterowanie przerwaniem RV32I						VER. v00.01	DATE/TIME 2026-07-20T00:00:00+02:00		
	PROJECT Freestanding C na RV32I						SERIES Freestanding RV32I and K&R	CARD 10/13		SHEET 3/8
	URL: Inf.	PRG.	CORE	SCOPE	LEVEL	POS.	GITEA UUID 8fb9c110-557b-5081-8ae2-a56800f872e8 ec04eccc-1c4d-523baa62-23f9b407009e			
	GITEA https://zsl-gitea.mpabi.pl/edu-inf/lab-rv32i-c-interrupts						AUTHOR W. PASIECZKA			
	HTML https://dce7fb9d-7b2f-5d49-96a2-3a30d3070b84.mpabi.pl/8fb9c110-557b-5081-8ae2-a56800f872e8									

4 Zadania — zachowane przykłady w profilu A1–A8

TASK TASK01 · ECALL i wznowienie pod mepc+4 2cd1da12-b916-5def-b868-da4b13ff9089
BLOCK A1 CONTEXT · AKTYWNE - kontrolowany wyjątek M-mode Zachowany przykład: <code>src/tasks/task01_ecall_resume.c</code>. Granica zaczyna się na instrukcji <code>ecall</code> w <code>main</code> i kończy po powrocie do kolejnej instrukcji. Nie ma zewnętrznego peryferium ani pending IRQ; rdzeń sam tworzy synchroniczną pułapkę.
BLOCK A2 STRUCTURE · AKTYWNE - mcause, mepc i mtvec <code>mcause</code> zapisuje klasę 11 bez bitu <code>interrupt</code>, <code>mepc</code> zapisuje adres ECALL, a baza <code>mtvec</code> prowadzi wszystkie synchroniczne wyjątki do <code>handle_exception</code>. Handler publikuje oryginał i osobny adres wznowienia.
BLOCK A3 DISPATCH · AKTYWNE - synchroniczny slot mtvec Sprzęt wybiera bazowy slot wektora, ponieważ zdarzenie jest wyjątkiem, nie IRQ. <code>handle_exception</code> odczytuje <code>mcause</code> i obsługuje jedyny dopuszczony przypadek ECALL. To granica <code>trap dispatch</code>, choć nie ma tablicy <code>callbacków</code>.
BLOCK A4 APPLICATION · AKTYWNE - cause 11 i stały krok 4 ECALL w RV32I jest instrukcją 32-bitową, dlatego dla tego przykładu poprawny <code>resume</code> ma wartość <code>mepc+4</code>. Gate wymaga jednego wejścia, <code>cause 11</code>, <code>delta 4</code> i ustawienia <code>resumed</code> dopiero po <code>mret</code>.
BLOCK A5 FLOW · AKTYWNE - ecall, save, patch, mret Normalny kod wykonuje ECALL. Rdzeń zapisuje <code>mepc/mcause</code>, skacze do wektora, handler odczytuje oba CSR i zapisuje nowe <code>mepc</code>. Epilog <code>interrupt("machine")</code> kończy się <code>mret</code>; dopiero potem <code>main</code> zapisuje <code>resumed</code>.
BLOCK A6 STATE · AKTYWNE - RUNNING, TRAP, RESUMED Stan przechodzi <code>RUNNING -> TRAP(mepc=ecall,cause=11) -> RESUMED(pc=mepc+4)</code>. Licznik 1 odróżnia pojedyncze wznowienie od pętli ponownego wykonywania ECALL.
BLOCK A7 RUNTIME · AKTYWNE - ecall i mret w ELF Oracle porównuje raportowane adresy i niezależnie wymaga różnicy 4. Objdump musi pokazać <code>ecall</code> w <code>main</code> oraz <code>mret</code> w handlerze. Segmenty LOAD pozostają rozdzielone R-X/R-W.
BLOCK A8 PATTERNS · AKTYWNE - explicit resume policy Wzorzec handlera wyjątku obejmuje jawną decyzję: <code>retry</code>, <code>skip</code> albo <code>terminate</code>. <code>Task01</code> wybiera <code>skip</code> tylko dla znanego ECALL. Dla faultu pamięci bezwarunkowe <code>mepc+4</code> mogłoby ukryć błąd i jest niedozwolone.
BLOCK Ćwiczenie · Przewidywanie → wykonanie → wniosek Przed uruchomieniem przewidź bit <code>interrupt</code> i kod <code>mcause</code>, wskaż wartość <code>mepc</code> oraz adres wznowienia. Wyjaśnij, co stałoby się bez zapisu <code>mepc+4</code>. Evidence: Adres <code>ecall</code> i dwa adresy <code>mepc</code> z raportu/listingu, <code>cause 11</code>, instrukcja <code>mret</code> oraz kod wyjścia <code>Hazard3</code>. Acceptance: <code>Hazard3: count=1, mcause=0x0000000b, resume_pc-mepc=4, resumed=1, pass=1</code> i kod 0.



TITLE		Pułapki i sterowanie przerwaniem RV32I		VER.	v00.01		DATETIME		2026-07-20T00:00:00+02:00	
PROJECT				SERIES		CARD		SHEET		
Freestanding C na RV32I				Freestanding RV32I and K&R		10/13		4/8		
DIR.	Inf.	PRG.	CORE	SCOPE	LEVEL	POS.	GITEA UUID 8fb9c110-557b-5081-8ae2-a56800f872e8 ec04ecce-1c44-523baa62-23f9b407009e			
GITEA https://zsl-gitea.mpabi.pl/edu-inf/lab-rv32i-c-interrupts							AUTHOR W. PAB18223E			
HTML https://dce7fb9d-7b2f-5d49-96a2-3a30d3070b84.mpabi.pl/8fb9c110-557b-5081-8ae2-a56800f872e8										



BLOCK TASK01 · Zapis dowodu ucznia

1. Przewidywanie przed uruchomieniem

2. mcause, mepc i adres wznowienia

3. ecall/mret w listingu i kod wyjścia

4. Wniosek: reguła języka lub kontrakt targetu potwierdzony przez pomiar

	TITLE		Pułapki i sterowanie przerwaniem RV32I		VER.	v00.01		DATE/TIME	2026-07-20T00:00:00+02:00																	
	PROJECT		Freestanding C na RV32I		SERIES		Freestanding RV32I and K&R		CARD	10/13		SHEET	5/8													
	REV.	Inf.		PRG.			CORE			SCOPE				LEVEL			POS.			GITEA UUID	8fb9c110-557b-5081-8ae2-a56800f872e8		e04ecee-1c4d-52ba-a6d2-34f8-40700e		NOTION W. PABIEZCZE	
	GITEA https://zsl-gitea.mpabi.pl/edu-inf/lab-rv32i-c-interrupts																									
	HTML https://dce7fb9d-7b2f-5d49-96a2-3a30d3070b84.mpabi.pl/8fb9c110-557b-5081-8ae2-a56800f872e8																									

TASK TASK02 · Machine software IRQ i clear źródła a4b503ba-cacc-5af1-9ecd-31f51563ca76
BLOCK A1 CONTEXT · AKTYWNE - kontrolowane źródło bez peryferium Zachowany przykład: src/tasks/task02_machine_software_irq.c. tb_set_softirq(0) podnosi machine software interrupt hart0. Testbench jest źródłem, CSR są kontrolą rdzenia, ISR usuwa żądanie, a main ocenia wznowienie. UART/GPIO nie uczestniczą.
BLOCK A2 STRUCTURE · AKTYWNE - trzy bity i jeden wektor Źródło jest widoczne jako mip.MSIP; indywidualna bramka to mie.MSIE, globalna to mstatus.MIE. Cause 3 w trybie vectored wybiera slot 3 i symbol isr_machine_softirq.
BLOCK A3 DISPATCH · AKTYWNE - sprzęt wybiera vector 3 Bit interrupt w mcause i kod 3 powodują sprzętowy wybór slotu machine software IRQ. Nie jest to statyczne jak z main. Po wejściu symbol handlera jest już określony przez wektor, a mret odtwarza przerwany kontekst.
BLOCK A4 APPLICATION · AKTYWNE - pending 1 do 0 Przy MSIE=1 i MIE=0 źródło jest już pending, lecz licznik wynosi 0. Po otwarciu MIE handler widzi 0x80000003 i pending=1, wykonuje clear, zwiększa licznik, a po mret main mierzy pending=0.
BLOCK A5 FLOW · AKTYWNE - configure, assert, observe, enable Kolejność eliminuje wyścig pomiaru: najpierw wyłącz wszystko, ustaw MSIE, podnieś źródło, zmierz pending, dopiero potem ustaw MIE. Handler odczytuje cause/pending przed clear i publikuje licznik po clear.
BLOCK A6 STATE · AKTYWNE - MASKED_PENDING, ACTIVE, CLEARED Źródło przechodzi do MASKED_PENDING, bo globalna bramka jest zamknięta. Po MIE staje się ACTIVE; clear w ISR tworzy CLEARED. Resumed=1 jest osobnym stanem normalnego kodu po mret.
BLOCK A7 RUNTIME · AKTYWNE - cause z bitem 31 Wartość 0x80000003 łączy bit interrupt i kod 3. Listing ma pokazać CSR dla mie/mstatus/mip, dostęp do testbench clear oraz mret. Oracle wymaga całego śladu, nie tylko pass.
BLOCK A8 PATTERNS · AKTYWNE - clear level source before publish Źródło poziomowe usuwa się przed opublikowaniem ukończenia ISR. Gdyby licznik wzrósł bez clear, po mret żądanie pozostałoby aktywne i handler mógłby wejść ponownie.
BLOCK Ćwiczenie · Przewidywanie → wykonanie → wniosek Zapisz stan pending przed globalnym enable, mcause w ISR i pending po clear. Rozdziel źródło MSIP od obu bramek oraz wskaż, dlaczego handler musi usunąć źródło przed mret. Evidence: Ślad źródło→pending→MSIE/MIE→vector 3→ISR→clear→mret, raport globali i kod wyjścia Hazard3. Acceptance: Hazard3: count=1, mcause=0x80000003, pending_before=1, pending_in_handler=1, pending_after=0, resumed=1, pass=1 i kod 0.

	TITLE Pułapki i sterowanie przerwaniem RV32I		VER v00.01	DATE TIME 2026-07-20T00:00:00+02:00			
	PROJECT Freestanding C na RV32I		SERIES Freestanding RV32I and K&R	CARD 10/13	SHEET 7/8		
	DIR: Inf.	PRG: CIRE	SCOPE	LEVEL	POS:		GITEA UUID 8fb9c110-557b-5081-8ae2-a56800f872e8
	GITEA URL https://zsl-gitea.mpabi.pl/edu-inf/lab-rv32i-c-interrupts						GITEA UUID ec04ecce-1c4d-523a-a6d2-34f9-407009e NOTHUR W. PAB122232E
HTML https://dce7fb9d-7b2f-5d49-96a2-3a30d3070b84.mpabi.pl/8fb9c110-557b-5081-8ae2-a56800f872e8							

TASK TASK03 · Macierz pending, MSIE i MIE f56dba83-f7a9-5643-8f2e-84df1e8a2190 BLOCK A1 CONTEXT · AKTYWNE - jedna zmienna, dwie kontrolki Zachowany przykład: src/tasks/task03_pending_enable_matrix.c. Bodziec MSIP pozostaje ten sam podczas dwóch prób blokowanych i próby dostarczonej. Przykład zmienia wyłącznie bramki rdzenia, więc nie myli nowego zdarzenia z nową konfiguracją. BLOCK A2 STRUCTURE · AKTYWNE - macierz 1×2×2 Oś danych to pending=1. Dwie osie sterujące to MSIE i MIE. Cztery kombinacje mają jeden stan dostarczalny: oba bity 1. Przykład mierzy trzy istotne wiersze przy utrzymanym pending. BLOCK A3 DISPATCH · AKTYWNE - wejście dopiero dla 1/1 Sprzętowy dispatch do vector 3 nie zachodzi przy żadnej pojedynczej otwartej bramce. Dopiero MSIE=1 i MIE=1 pozwalają utworzyć trap, zapisać cause i wejść do ISR kończącego się mret. BLOCK A4 APPLICATION · AKTYWNE - dwa razy blocked, raz delivered Przy MIE=1/MSIE=0 licznik zostaje 0. Przy MIE=0/MSIE=1 również zostaje 0. Przy obu równych 1 rośnie dokładnie do 1, mcause ma wartość 0x80000003, a clear zeruje pending. BLOCK A5 FLOW · AKTYWNE - nie zmieniaj źródła między próbami Program podnosi MSIP raz, następnie testuje global-only, individual-only i both. Globalny bit jest zamykany przed zmianą MSIE, aby dostarczenie nie zaszło między pomiarem a zapisaniem wyniku. BLOCK A6 STATE · AKTYWNE - P10, P01, P11, CLEAR Skrót Pxy oznacza pending=1, MSIE=x, MIE=y. Ślad to P01(blocked) -> P10(blocked) -> P11(active) -> CLEAR. Licznik zmienia się tylko na trzeciej krawędzi. BLOCK A7 RUNTIME · AKTYWNE - odczyt mip, zapisy mie/mstatus Raport utrwała trzy decyzje i cause. Listing rozdziela csrs/csrs mie, csrs/csrs mstatus oraz odczyt mip. Objdump niezależnie potwierdza mret w ISR. BLOCK A8 PATTERNS · AKTYWNE - source, individual, global gate Trzywarstwowy model zapobiega typowemu błędowi: pending nie oznacza delivered. Ten sam wzorec wróci dla MTIP/MTIE oraz external IRQ z enable peryferium, kontrolera i globalnym MIE. BLOCK Ćwiczenie · Przewidywanie → wykonanie → wniosek Wypełnij trzy wiersze: pending=1 z samym MIE, pending=1 z samym MSIE oraz pending=1 z obiema bramkami. Przewidź licznik ISR i końcowy pending. Evidence: Tabela prawdy dwóch bramek dla jednego utrzymywanego pending, licznik ISR, końcowy clear, mret i kod wyjścia Hazard3. Acceptance: Hazard3: pending=1, blocked_no_msie=1, blocked_no_mie=1, delivered_both=1, count=1, mcause=0x80000003, pending_after=0, pass=1 i kod 0.

	TITLE: Pułapki i sterowanie przerwaniem RV32I		VER: v00.01	DATETIME: 2026-07-20T00:00:00+02:00			
	PROJECT: Freestanding C na RV32I		SERIES: Freestanding RV32I and K&R	CARD: 10/13	SHEET: 8/8		
	OBJ: Inf.	PRG: CURE	SCOPE: LEVEL	POS:	GITEA UUID: 8fb9c110-557b-5081-8ae2-a56800f872e8		ec04ecce-1c4d-523a-a6a2-23f0b407009e
	GITEA https://zsl-gitea.mpabi.pl/edu-inf/lab-rv32i-c-interrupts						
HTML https://dce7fb9d-7b2f-5d49-96a2-3a30d3070b84.mpabi.pl/8fb9c110-557b-5081-8ae2-a56800f872e8							

BLOCK TASK03 · Zapis dowodu ucznia
1. Przewidywanie przed uruchomieniem
.....
.....
2. Macierz pending/MSIE/MIE
.....
.....
3. Dostarczenie, clear i kod wyjścia
.....
.....
4. Wniosek: reguła języka lub kontrakt targetu potwierdzony przez pomiar
.....
.....