

## 1 MODEL

$\mu ::= \text{wk}$	(Weak)	$\zeta ::= \text{cta}$	(Thread group)
$\text{rlx}$	(Relaxed)	$\text{gpu}$	(Processor)
$\text{ra}$	(Release/Acquire)	$\text{sys}$	(System)
$\text{sc}$	(Sequentially Consistent)		

Orders/Relations in model

- $\trianglelefteq$  is the old  $\leq$  (without coherence stuff from [F4](#) and [P5B](#)).  
This provides the NO-TAR axiom.
- $\leq$  is a the *happens-before* suborder, which only includes [rf](#) when they are morally strong.  
This serves as a cross-location transitive kernel for the per-location order.
- $\sqsubseteq$  is a per-location order that relates morally strong and [poloc](#) accesses.  
This includes  $\leq$  for morally strong accesses.  
This provides the SC-PER-LOC axiom.

Write  $d \Delta e$  if they conflict (ie, read/write or write/write, same location).

Write  $d \blacktriangle e$  if they conflict and are morally strong

*Definition 1.1.* A pomset with preconditions is a tuple  $(E, \lambda, \leq, \trianglelefteq, \sqsubseteq)$  where

- (M1)  $E$  is a set of events
- (M2)  $\lambda : E \rightarrow (\Phi \times \mathcal{A})$  is a labeling from which we derive functions
  - $\Phi : E \rightarrow \Phi$  (formulae)
  - $\mathcal{A} : E \rightarrow \mathcal{A}$  (actions)
- (M3)  $\leq \subseteq (E \times E)$ ,  $\trianglelefteq \subseteq (E \times E)$ , and  $\sqsubseteq \subseteq (E \times E)$  are partial orders
- (M4)  $\bigwedge_e \Phi(e)$  is satisfiable (consistency)
- (M5) if  $d \trianglelefteq e$  then  $\Phi(e)$  implies  $\Phi(d)$  (causal strengthening)
- (M6) if  $d \leq e$  then  $d \trianglelefteq e$
- (M7) if  $d \leq e$  and  $d$  conflicts with  $e$  then  $d \sqsubseteq e$

We say  $d < e$  when  $d \leq e$  and  $d \neq e$ , and similarly for  $\triangleleft$  and  $\sqsubset$ .

*Definition 1.2 (Strong fulfillment).* We say  $\mathcal{A}(d) = (Wxv)$  fulfills  $\mathcal{A}(e) = (Rxv)$  if

- (F3A)  $d \triangleleft e$
- (F3B)  $d < e$  if  $d$  is morally strong with  $e$
- (F3C)  $d \sqsubseteq e$  (if  $d$  is not morally strong with  $e$ )
- (F4)  $\forall \mathcal{A}(c) = (Wx..)$  either  $c \sqsubseteq d$  or  $e \sqsubseteq c$ ,

*Definition 1.3 (Weak fulfillment).* We say  $\mathcal{A}(d) = (Wxv)$  fulfills  $\mathcal{A}(e) = (Rxv)$  if

- (F3A)  $d \triangleleft e$
- (F3B)  $d < e$  if  $d$  is morally strong with  $e$
- (F3C)  $e \not\sqsubseteq d$  (if  $d$  is not morally strong with  $e$ )
- (F4)  $\forall \mathcal{A}(c) = (Wx..)$  either  $c \sqsubset d$  or  $e \sqsubset c$ , where

$$d \sqsubset e \text{ when } \begin{cases} d \sqsubseteq e & \text{if } d \text{ is morally strong with } e \\ e \not\sqsubseteq d & \text{otherwise} \end{cases}$$

If all accesses are morally strong with each other, weak fulfillment degenerates to

- (F3)  $d < e$
- (F4)  $\forall \mathcal{A}(c) = (Wx..)$  either  $c \sqsubseteq d$  or  $e \sqsubseteq c$

If no accesses are morally strong with each other, weak fulfillment degenerates to

(F3)  $e \not\sqsubseteq d$

(F4)  $\nexists \mathcal{A}(c) = (Wx..) \text{ both } d \sqsubset c \text{ and } c \sqsubset e$

*Definition 1.4.* Let  $P' \in (\phi \mid a) \Rightarrow \mathcal{P}$  when  $(\exists P \in \mathcal{P}) (\forall e \in E)$

(p1)  $E' = E \cup \{d\}$

(p2)  $\leq' \supseteq \leq, \leq' \supseteq \leq$ , and  $\sqsubseteq' \supseteq \sqsubseteq$

(p3A)  $\mathcal{A}'(e) = \mathcal{A}(e)$

(p3B)  $\mathcal{A}'(d) = a$

(p4A)  $\Phi'(d)$  implies  $\phi \wedge (d \notin E \vee \Phi(d))$

(p4B) if  $d \neq (R..)$  then  $e = d$  or  $\Phi'(e)$  implies  $\Phi(e)$

(p4C) if  $d = (Rvx)$  then  $e = d$  or  $\Phi'(e)$  implies  $\Phi(e)[v/x]$

(p5A) if  $d = (R..), e = (W..)$  then  $e = d$  or  $\Phi'(e)$  implies  $\Phi(e)$  or  $d \leq' e$

(p5B) if  $d$  conflicts with  $e$  then  $d \sqsubseteq' e$

(p5C) if  $d$  is an acquire or  $e$  is a release then  $d \leq' e$

(p5D) if  $d$  is an SC write and  $e$  is an SC read then  $d \leq' e$

(p5E) if  $d$  reads, and  $e$  is an acquiring fence, then  $d \leq' e$

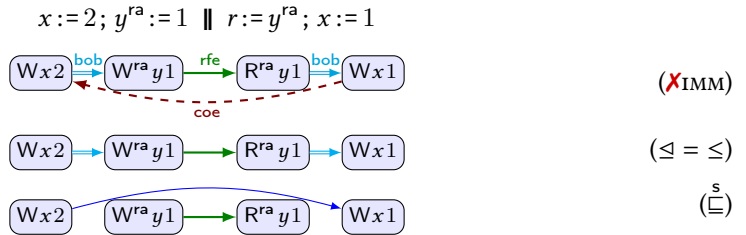
(p5F) if  $d$  is a releasing fence, and  $e$  writes, then  $d \leq' e$

## 2 IMM EXAMPLES

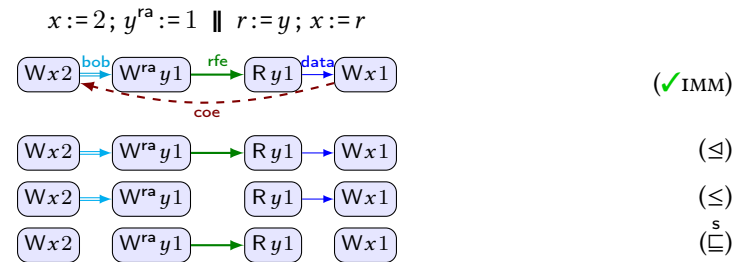
Interpreting this definition for the IMM:

- No wk, default is rlx
- All threads in same cta (only one scope)
- Actions are morally strong when both are ra/sc, mimicking happens-before
- Strong fulfillment may do the right thing

Disallowed by IMM:

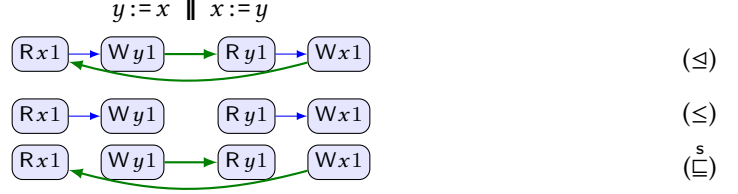


Allowed by IMM, but not by Power/ARMv7/ARMv8/TSO:

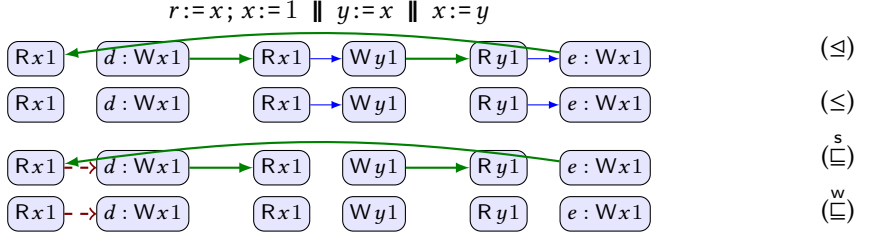


Anton says we could potentially disallow by adding an axiom to IMM forbidding cycles in  $\text{co} \cup ([W]; \text{rfe}^?; ([R^{ra}] \cup \text{po}; [FW^{ra}]); \text{ar}^*; [W])$

Need  $\leq$  to prevent thin air on rx:



Example from talk:



Comment: Two order idea from OOPSLA talk does not work for this example. In this setting it corresponds to:

- Require:  $d \sqsubseteq e$  when  $d \leq e$  and they conflict

Using this, we would have a cycle (weak/strong fulfillment not relevant here):



### 3 PTX EXAMPLES

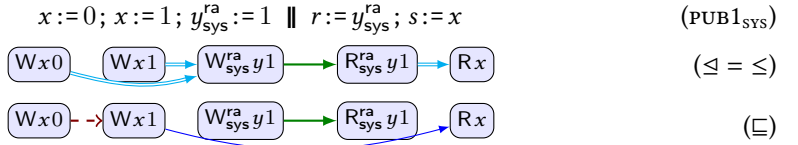
Based on [Lustig et al. 2019; NVIDIA 2020].

PTX requires weak fulfillment.

Default scope is cta. In examples, all threads in different ctas.

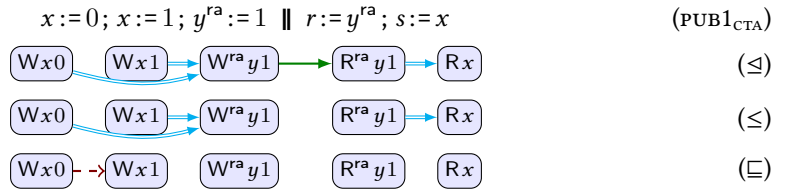
Default mode is wk.

(Rx0) must be forbidden. Before fulfilling the read:



(Wx1)  $\sqsubseteq$  (Rx) is required by m7, enforcing publication.

(Rx0) must be allowed:

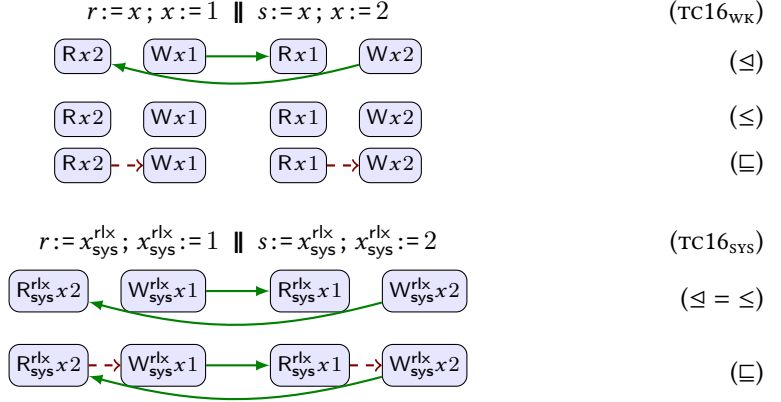


We do not have  $(W^{\text{ra}}y1) \leq (R^{\text{ra}}y1)$  since f3 only requires order for things that are morally strong.

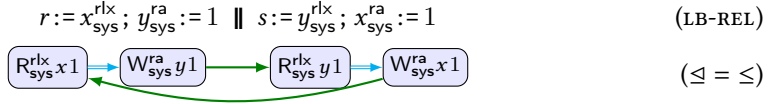
Another example that may be of interest (nothing morally strong). Can this (R<sub>x0</sub>)?

$$x := 0; x := 1 \parallel y := x \parallel \text{if}(y)\{r := x\}$$

PTX allows TC16 for events that are not mutually strong (TC16<sub>WK</sub>), but disallows it when events are mutually strong (TC16<sub>SYS</sub>). Note that  $\leq$  imposes no requirements here. Fulfillment imposes no order. This example shows that F3c cannot be strengthened to require that  $d \sqsubseteq e$ .

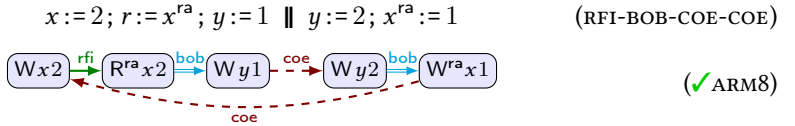


About Release-Acquire semantics. Anton confirms that the following example is allowed in C11, but disallowed in the IMM. It is apparently allowed in C11 with the intention to allow releasing writes to be downgraded to relaxed in the case that only fulfill relaxed reads.

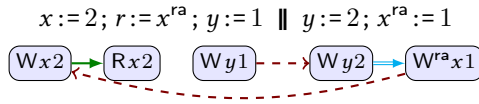


## 4 RFI EXAMPLES

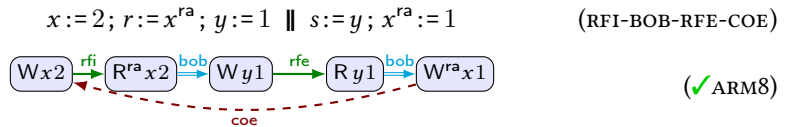
Anton example 1 (Allowed by ARM) [rfi-bob-coe-coe]



To allow this, weaken ra to rlx when read fulfilled by relaxed write of same thread (don't need to allow this when the write is part of an RMW).

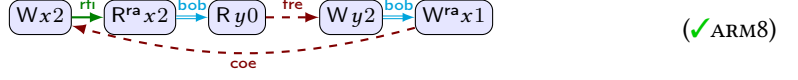


RF variant [rfi-bob-rfe-coe]:



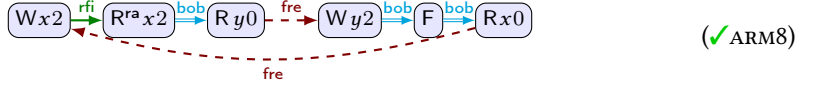
TSO variant [rfi-bob-fre-coe]:

$$x := 2; r := x^{ra}; s := y \parallel y := 2; x^{ra} := 1 \quad (\text{RFI-BOB-COE-COE})$$



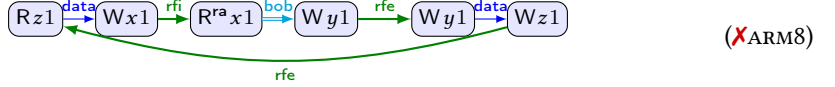
Double FRE variant [rfi-bob-fre-fre]:

$$x := 2; r := x^{ra}; s := y \parallel y := 2; F; r := x \quad (\text{RFI-BOB-FRE-FRE})$$



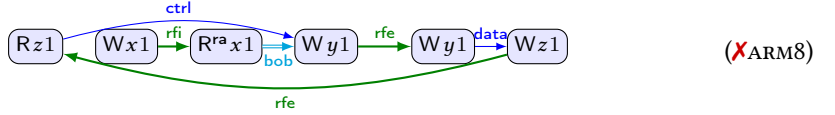
It does not seem possible to do this only with rfe. ARM disallows this [data-rfi-rfe-rfe]:

$$x := z; r := x^{ra}; y := 1 \parallel z := y \quad (\text{DATA-RFI-RFE-RFE})$$



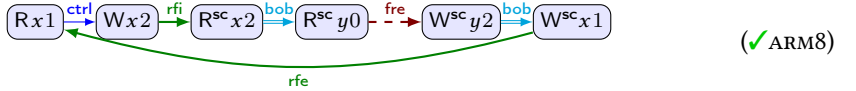
It also disallows [ctrl-rfi-rfe-rfe]:

$$\text{if}(z)\{\}; x := 1; r := x^{ra}; y := 1 \parallel z := y \quad (\text{CTRL-RFI-RFE-RFE})$$



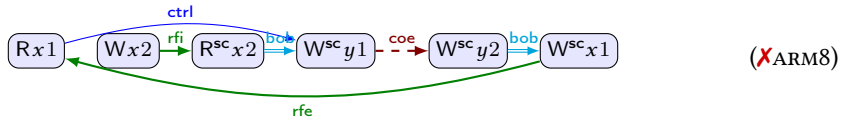
ARM allows some counterintuitive results for SC access [ctrl-rfi-fre-rfe]:

$$\text{if}(x)\{\}; x := 2; r := x^{sc}; s := y^{sc} \parallel y^{sc} := 2; x^{sc} := 1 \quad (\text{CTRL-RFI-FRE-RFE})$$



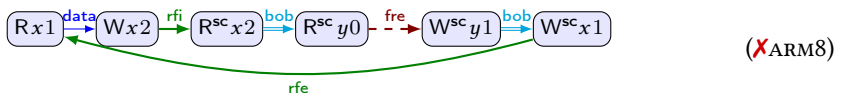
Not possible with coe [ctrl-rfi-coe-rfe]:

$$\text{if}(x)\{\}; x := 2; r := x^{sc}; y^{sc} := 1 \parallel y^{sc} := 2; x^{sc} := 1 \quad (\text{CTRL-RFI-COE-RFE})$$



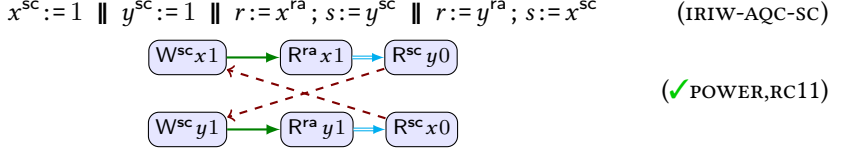
This is not allowed with a data dependency instead of a control dependency [data-rfi-fre-rfe]:

$$x := x + 1; r := x^{sc}; s := y^{sc} \parallel y^{sc} := 1; x^{sc} := 1 \quad (\text{DATA-RFI-FRE-RFE})$$

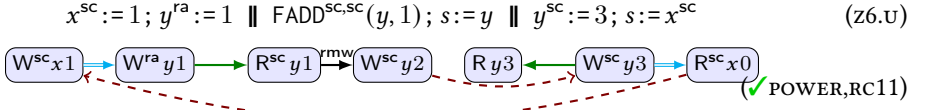


## 5 SC EXAMPLES

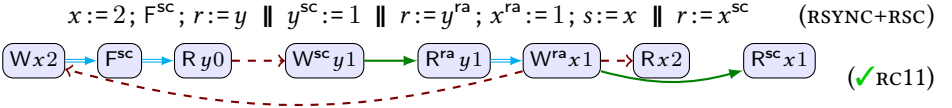
IRIW-AQC-SC is allowed by trailing-sync compilation to power [Lahav et al. 2017, §1].



Leading sync is also unsound in c11 with RMW [Lahav et al. 2017, §2.1].

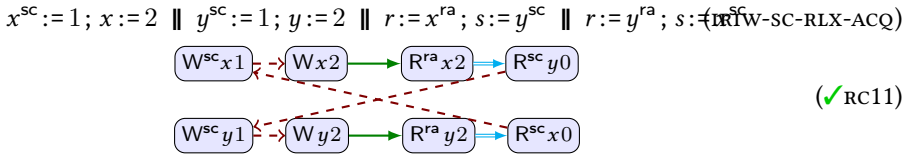


Leading sync is also unsound in c11 with SC fences [Lahav et al. 2017, §A.1].

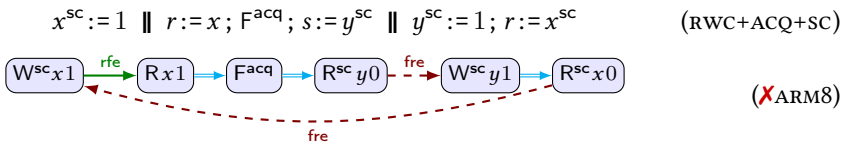


Fulfillment of  $(Rx2)$  requires that either  $(W^{ra}x1) \rightarrow (Wx2)$  or  $(Rx2) \rightarrow (W^{ra}x1)$ . It's interesting that in the pomset,  $(R^{sc}x1)$  is not needed to get a cycle.

There is a long discussion of this in [Bender and Palsberg 2019, §5.2, Fig. 17], where they also discuss this example:



[Lahav et al. 2017, §A.2] claims that ARM8 allows this [RWC+acq+sc], but herd7 rejects it. Reason: they are citing the flowing/pop model [Flur et al. 2016] rather than [Pulte et al. 2018].

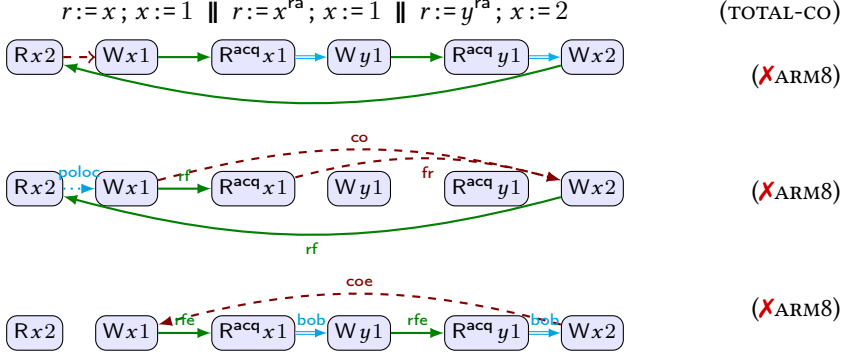


## 6 RMWS

From [Bender and Palsberg 2019, §3.3]. With partial coherence/weak fulfillment you need to be careful that RMWs are totally ordered (if that's a property you want). May not come for free.

## 7 EXAMPLE FROM JAM PAPER

From [Bender and Palsberg 2019, §B]: “Here we demonstrate that it is possible to construct a program that is only forbidden due to the total coherence order”



## 8 MORE MODEL

These definitions need to be updated to include the additional orders.

*Definition 8.1.* A pomset is  $x$ -closed if

- every  $\mathcal{A}(e) = (Rx..)$  is fulfilled
- every  $\Phi(e)$  is independent of  $x$ :  $(\forall v. \Phi(e) \models \Phi(e)[v/x] \models \Phi(e))$

*Definition 8.2.* Let  $P \in (vx.\mathcal{P})$  when  $P \in \mathcal{P}$  and  $P$  is  $x$ -closed

*Definition 8.3.* Let  $P \in (\phi \triangleright \mathcal{P})$  when  $P \in \mathcal{P}$  and  $(\forall e \in E) \Phi(e)$  implies  $\phi$

*Definition 8.4.* Let  $P' \in (\mathcal{P}[M/x])$  when  $(\exists P \in \mathcal{P})$

$E' = E, \leq' = \leq, \mathcal{A}' = \mathcal{A}$ , and  $(\forall e \in E') \Phi'(e) = \Phi(e)[M/x]$

*Definition 8.5.* Let  $P' \in (\mathcal{P}^1 \parallel \mathcal{P}^2)$  when  $(\exists P^1 \in \mathcal{P}^1) (\exists P^2 \in \mathcal{P}^2)$

$E' = E^1 \cup E^2, \leq' \supseteq \leq^1 \cup \leq^2$ , and  $(\forall e \in E')$  either

- $e \notin E^2, \mathcal{A}'(e) = \mathcal{A}^1(e)$  and  $\Phi'(e)$  implies  $\Phi^1(e)$ ,
- $e \notin E^1, \mathcal{A}'(e) = \mathcal{A}^2(e)$  and  $\Phi'(e)$  implies  $\Phi^2(e)$ , or
- $\mathcal{A}'(e) = \mathcal{A}^1(e) = \mathcal{A}^2(e)$  and  $\Phi'(e)$  implies  $\Phi^1(e) \vee \Phi^2(e)$

Language

$$\begin{aligned}
 \llbracket \text{skip} \rrbracket &\triangleq \{\checkmark\} \\
 \llbracket r := M; C \rrbracket &\triangleq \llbracket C \rrbracket [M/r] \\
 \llbracket r := x^\mu; C \rrbracket &\triangleq \bigcup_v (R^\mu xv \Rightarrow \llbracket C \rrbracket [x/r]) \\
 \llbracket x^\mu := M; C \rrbracket &\triangleq \bigcup_v (M = v \mid W^\mu xv \Rightarrow \llbracket C \rrbracket [M/x]) \\
 \llbracket F^v; C \rrbracket &\triangleq (F^v) \Rightarrow \llbracket C \rrbracket \\
 \llbracket \text{if } (M)\{C\} \text{ else } \{D\} \rrbracket &\triangleq (M \triangleright \llbracket C \rrbracket) \parallel (\neg M \triangleright \llbracket D \rrbracket) \\
 \llbracket C \parallel D \rrbracket &\triangleq \llbracket C \rrbracket \parallel \llbracket D \rrbracket \\
 \llbracket \text{var } x; C \rrbracket &\triangleq vx. \llbracket C \rrbracket
 \end{aligned}$$

$\nu$	rel	(Release)
	acq	(Acquire)
	sc	(SC)

A citation: [Jagadeesan et al. 2020]

## REFERENCES

- John Bender and Jens Palsberg. 2019. A formalization of Java’s concurrent access modes. *Proc. ACM Program. Lang.* 3, OOPSLA (2019), 142:1–142:28. <https://doi.org/10.1145/3360568>
- Shaked Flur, Kathryn E. Gray, Christopher Pulte, Susmit Sarkar, Ali Sezgin, Luc Maranget, Will Deacon, and Peter Sewell. 2016. Modelling the ARMv8 architecture, operationally: concurrency and ISA. In *Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2016, St. Petersburg, FL, USA, January 20 - 22, 2016*, Rastislav Bodík and Rupak Majumdar (Eds.). ACM, 608–621. <https://doi.org/10.1145/2837614.2837615>
- Radha Jagadeesan, Alan Jeffrey, and James Riely. 2020. Pomsets with Preconditions: A Simple Model of Relaxed Memory. *Proc. ACM Program. Lang.* 4, OOPSLA, Article 194 (Nov. 2020), 30 pages. <https://doi.org/10.1145/3428262>
- Ori Lahav, Viktor Vafeiadis, Jeehoon Kang, Chung-Kil Hur, and Derek Dreyer. 2017. Repairing sequential consistency in C/C++11. In *Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2017, Barcelona, Spain, June 18-23, 2017*, Albert Cohen and Martin T. Vechev (Eds.). ACM, 618–632. <https://doi.org/10.1145/3062341.3062352>
- Daniel Lustig, Sameer Sahasrabudde, and Olivier Giroux. 2019. A Formal Analysis of the NVIDIA PTX Memory Consistency Model. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS 2019, Providence, RI, USA, April 13-17, 2019*, Iris Bahar, Maurice Herlihy, Emmett Witchel, and Alvin R. Lebeck (Eds.). ACM, 257–270. <https://doi.org/10.1145/3297858.3304043>
- NVIDIA. 2020. Parallel Thread Execution ISA Version 7.1. <https://docs.nvidia.com/cuda/parallel-thread-execution/index.html#memory-consistency-model>.
- Christopher Pulte, Shaked Flur, Will Deacon, Jon French, Susmit Sarkar, and Peter Sewell. 2018. Simplifying ARM concurrency: multicopy-atomic axiomatic and operational models for ARMv8. *PACMPL* 2, POPL (2018), 19:1–19:29. <https://doi.org/10.1145/3158107>