



# Evaluating Capabilities for Assurance of Third Party Intellectual Property

**Vivian Guzman Kammler | R&D Cybersecurity Lead**

**Virtual Event | November 4, 2021**



**Sandia National Laboratories**

*Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.*

***This presentation describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the presentation do not necessarily represent the views of the U.S. Department of Energy or the United States Government.***

***Presenting an evaluation of tools. This does not constitute an endorsement.***



**SANDIA IS A FEDERALLY FUNDED RESEARCH AND  
DEVELOPMENT CENTER (FFRDC) MANAGED AND OPERATED BY**

**National Technology & Engineering  
Solutions of Sandia, LLC, a wholly  
owned subsidiary of Honeywell  
International Inc.**

**Government owned, contractor operated**

**FFRDCs are long-term strategic partners  
to the federal government, operating in the  
public interest with objectivity and  
independence and maintaining core  
competencies in missions of national  
significance**

## The nuclear deterrent should

**Always**

**NEVER**

always be available for use when needed

never go off unless authorized

# Zero Trust and Quantifiable Assurance

## Hardware Assurance

An activity to ensure a level of confidence that microelectronics (also known as microcircuits, semiconductors, and integrated circuits, including its embedded software and/or **intellectual property**) function as intended and are **free of known vulnerabilities**, either intentionally or unintentionally designed or inserted as part of the system's hardware and/or its embedded software and/or intellectual property, **throughout the life cycle**.

DAU Glossary



## Constraints

*“activity to ensure a level of confidence”*

- **3PIP is not encrypted**
- **3PIP can be parsed by analysis tools**
- **Automation**
- *Evaluator has limited design knowledge*
- *There is no “golden” model (known good)*

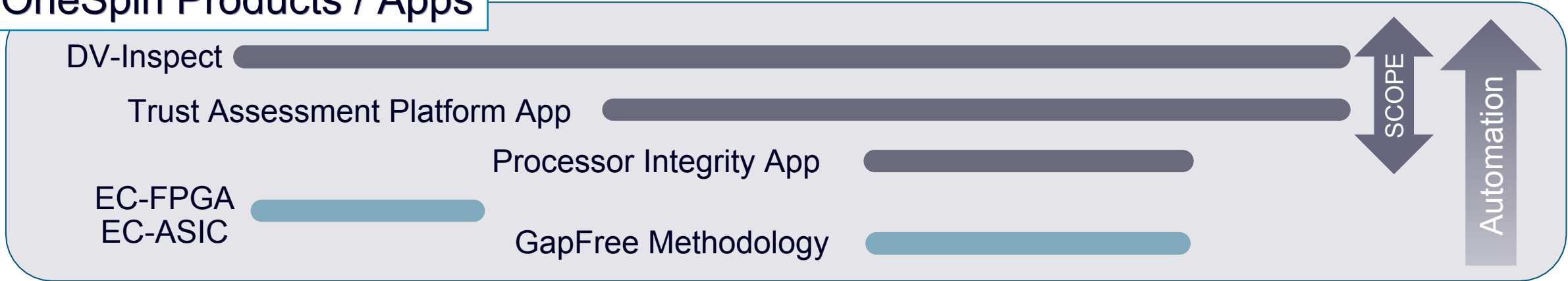
## Success Criteria

*3PIP is “free of known vulnerabilities”*

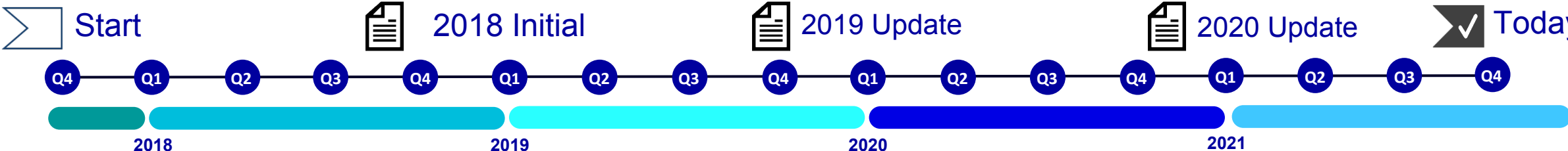
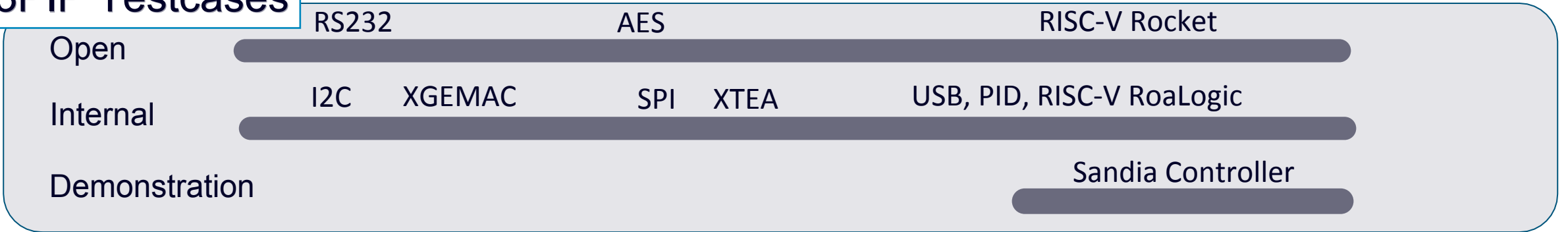
- **Verified functionality**
  - Available or provided functional checks pass
- **Detection of Hardware Trojans**
- *Verified absence of vulnerabilities based on prior knowledge (catalogs, databases)*

# Evaluating OneSpin Tools for Assurance of 3PIP

## OneSpin Products / Apps



## 3PIP Testcases



# How is 3PIP provided?

← OneSpin tools most useful here →

- 1 – based on ethernet MAC
- 2 – based on RISC-V RocketCore

```
always @(posedge clk_xgmii_rx or negedge reset_xgmii_rx_n) begin
46 if (reset_xgmii_rx_n == 1'b0) begin
147 curr_state <= SM_INIT;
148 col_cnt <= 8'b0;
149 last_seq_type <= LINK_FAULT_OK;
150 link_fault <= LINK_FAULT_OK;
151 seq_cnt <= 3'b0;
152 end
153 else begin
154 case (curr_state)
155 SM_INIT:
156 SM_COUNT:
157 SM_FAULT:
158 begin
159 col_cnt <= col_cnt + 8'd2;
160 if (!fault_sequence[0] && col_cnt >= 8'd127) begin
161 // No new fault in lower lanes and almost
162 // reached the 128 columns count, abort fault.
163 curr_state <= SM_INIT;
164 end
165 else if (col_cnt > 8'd127) begin
166 // Reached the 128 columns count, abort fault.
167 curr_state <= SM_INIT;
168 end
169 else if (!fault_sequence) begin
170 // Clear the column count each time we see a fault,
171 // if fault changes, go no next state.
172 col_cnt <= 8'd0;
173 if (seq_type != last_seq_type) begin
174 curr_state <= SM_NEW_FAULT;
175 end
176 end
177 end
178 SM_NEW_FAULT:
179
```

HDL

28

state bits/line of code<sup>1</sup>

```
assign _S0_S15 = (~40'00); read_x0; // @CSA.scale 136:56:115:Freeform.unleashed.DevC1588FFGDesign_u170DevC1588FFG_Flg@011168.4
180 assign pending_interrupts = _S0_S15 & reg_n0; // @CSA.scale 136:56:115:Freeform.unleashed.DevC1588FFGDesign_u170DevC1588FFG_Flg@011168.4
181 assign _S0_S14 = (~34'00); is_interrupts_debug; // @CSA.scale 137:41:115:Freeform.unleashed.DevC1588FFGDesign_u170DevC1588FFG_Flg@011162.4
182 assign d_interrupts = _S0_S14 & (~34); // @CSA.scale 137:41:115:Freeform.unleashed.DevC1588FFGDesign_u170DevC1588FFG_Flg@011162.4
183 assign _T_301 = _T_3416; reg_status_n0; // @CSA.scale 138:51:115:Freeform.unleashed.DevC1588FFGDesign_u170DevC1588FFG_Flg@011164.4
184 assign _T_301 = pending_interrupts; // @CSA.scale 138:51:115:Freeform.unleashed.DevC1588FFGDesign_u170DevC1588FFG_Flg@011164.4
185 assign _T_304 = _T_301; reg_n0; // @CSA.scale 138:51:115:Freeform.unleashed.DevC1588FFGDesign_u170DevC1588FFG_Flg@011164.4
186 assign _T_304 = ~_T_301; reg_n0; // @CSA.scale 138:51:115:Freeform.unleashed.DevC1588FFGDesign_u170DevC1588FFG_Flg@011164.4
187
```

Generated HDL

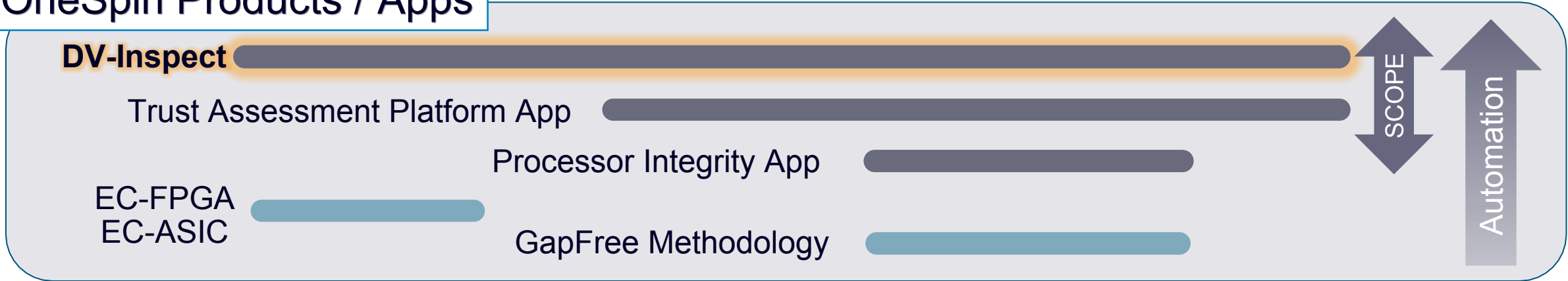
0.4

state bits/line of code<sup>2</sup>

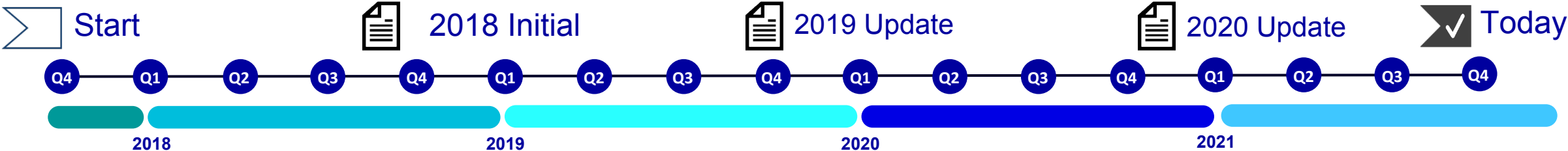
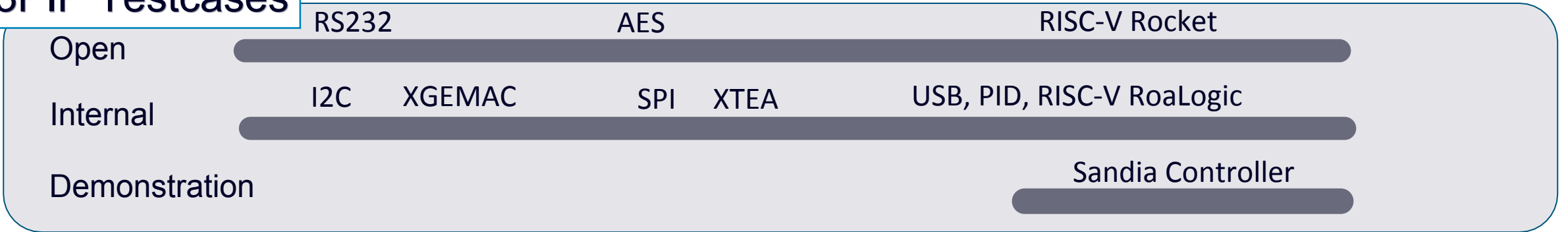
```
X_BUF #[
188 .LOC { "SLICE_X189V175" }
189 ]
190 VHS4_USED {
191 .I1904,
192 .O1904,
193 .O1904_0
194 }
195 X_LUT4 #[
196 .INIT { "16'bffff" },
197 .LOC { "SLICE_X189V175" }
198 ]
199 Vtx_data_fifo0_fifo0_ctrl0_wb0_nck_level [2],
200 .ADDR1[Vtx_data_fifo0_fifo0_ctrl0_nck_level [2]],
201 .ADDR2[Vtx_data_fifo0_fifo0_ctrl0_nck_level [1]],
202 .ADDR3[Vtx_data_fifo0_fifo0_ctrl0_nck_level [0]],
203 .O1904
204 ]
205 X_BUF #[
206 .LOC { "SLICE_X189V175" }
207 ]
208 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D0MUX {
209 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [0]],
210 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [0], D0MUX_03476 ]
211 }
212 X_BUF #[
213 .LOC { "SLICE_X189V175" }
214 ]
215 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D1MUX {
216 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [1]],
217 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [1], D1MUX_03467 ]
218 }
219 X_BUF #[
220 .LOC { "SLICE_X189V175" }
221 ]
222 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D2MUX {
223 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [2]],
224 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [2], D2MUX_03457 ]
225 }
226 X_BUF #[
227 .LOC { "SLICE_X189V175" }
228 ]
229 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D3MUX {
230 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [3]],
231 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [3], D3MUX_03456 ]
232 }
233 X_BUF #[
234 .LOC { "SLICE_X189V175" }
235 ]
236 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D4MUX {
237 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [4]],
238 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [4], D4MUX_03456 ]
239 }
240 X_BUF #[
241 .LOC { "SLICE_X189V175" }
242 ]
243 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D5MUX {
244 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [5]],
245 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [5], D5MUX_03456 ]
246 }
247 X_BUF #[
248 .LOC { "SLICE_X189V175" }
249 ]
250 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D6MUX {
251 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [6]],
252 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [6], D6MUX_03456 ]
253 }
254 X_BUF #[
255 .LOC { "SLICE_X189V175" }
256 ]
257 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D7MUX {
258 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [7]],
259 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [7], D7MUX_03456 ]
260 }
261 X_BUF #[
262 .LOC { "SLICE_X189V175" }
263 ]
264 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D8MUX {
265 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [8]],
266 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [8], D8MUX_03456 ]
267 }
268 X_BUF #[
269 .LOC { "SLICE_X189V175" }
270 ]
271 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D9MUX {
272 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [9]],
273 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [9], D9MUX_03456 ]
274 }
275 X_BUF #[
276 .LOC { "SLICE_X189V175" }
277 ]
278 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D10MUX {
279 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [10]],
280 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [10], D10MUX_03456 ]
281 }
282 X_BUF #[
283 .LOC { "SLICE_X189V175" }
284 ]
285 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D11MUX {
286 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [11]],
287 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [11], D11MUX_03456 ]
288 }
289 X_BUF #[
290 .LOC { "SLICE_X189V175" }
291 ]
292 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D12MUX {
293 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [12]],
294 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [12], D12MUX_03456 ]
295 }
296 X_BUF #[
297 .LOC { "SLICE_X189V175" }
298 ]
299 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D13MUX {
300 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [13]],
301 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [13], D13MUX_03456 ]
302 }
303 X_BUF #[
304 .LOC { "SLICE_X189V175" }
305 ]
306 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D14MUX {
307 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [14]],
308 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [14], D14MUX_03456 ]
309 }
310 X_BUF #[
311 .LOC { "SLICE_X189V175" }
312 ]
313 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D15MUX {
314 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [15]],
315 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [15], D15MUX_03456 ]
316 }
317 X_BUF #[
318 .LOC { "SLICE_X189V175" }
319 ]
320 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D16MUX {
321 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [16]],
322 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [16], D16MUX_03456 ]
323 }
324 X_BUF #[
325 .LOC { "SLICE_X189V175" }
326 ]
327 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D17MUX {
328 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [17]],
329 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [17], D17MUX_03456 ]
330 }
331 X_BUF #[
332 .LOC { "SLICE_X189V175" }
333 ]
334 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D18MUX {
335 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [18]],
336 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [18], D18MUX_03456 ]
337 }
338 X_BUF #[
339 .LOC { "SLICE_X189V175" }
340 ]
341 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D19MUX {
342 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [19]],
343 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [19], D19MUX_03456 ]
344 }
345 X_BUF #[
346 .LOC { "SLICE_X189V175" }
347 ]
348 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D20MUX {
349 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [20]],
350 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [20], D20MUX_03456 ]
351 }
352 X_BUF #[
353 .LOC { "SLICE_X189V175" }
354 ]
355 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D21MUX {
356 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [21]],
357 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [21], D21MUX_03456 ]
358 }
359 X_BUF #[
360 .LOC { "SLICE_X189V175" }
361 ]
362 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D22MUX {
363 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [22]],
364 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [22], D22MUX_03456 ]
365 }
366 X_BUF #[
367 .LOC { "SLICE_X189V175" }
368 ]
369 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D23MUX {
370 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [23]],
371 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [23], D23MUX_03456 ]
372 }
373 X_BUF #[
374 .LOC { "SLICE_X189V175" }
375 ]
376 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D24MUX {
377 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [24]],
378 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [24], D24MUX_03456 ]
379 }
380 X_BUF #[
381 .LOC { "SLICE_X189V175" }
382 ]
383 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D25MUX {
384 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [25]],
385 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [25], D25MUX_03456 ]
386 }
387 X_BUF #[
388 .LOC { "SLICE_X189V175" }
389 ]
390 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D26MUX {
391 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [26]],
392 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [26], D26MUX_03456 ]
393 }
394 X_BUF #[
395 .LOC { "SLICE_X189V175" }
396 ]
397 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D27MUX {
398 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [27]],
399 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [27], D27MUX_03456 ]
400 }
401 X_BUF #[
402 .LOC { "SLICE_X189V175" }
403 ]
404 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D28MUX {
405 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [28]],
406 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [28], D28MUX_03456 ]
407 }
408 X_BUF #[
409 .LOC { "SLICE_X189V175" }
410 ]
411 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D29MUX {
412 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [29]],
413 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [29], D29MUX_03456 ]
414 }
415 X_BUF #[
416 .LOC { "SLICE_X189V175" }
417 ]
418 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D30MUX {
419 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [30]],
420 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [30], D30MUX_03456 ]
421 }
422 X_BUF #[
423 .LOC { "SLICE_X189V175" }
424 ]
425 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D31MUX {
426 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [31]],
427 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [31], D31MUX_03456 ]
428 }
429 X_BUF #[
430 .LOC { "SLICE_X189V175" }
431 ]
432 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D32MUX {
433 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [32]],
434 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [32], D32MUX_03456 ]
435 }
436 X_BUF #[
437 .LOC { "SLICE_X189V175" }
438 ]
439 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D33MUX {
440 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [33]],
441 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [33], D33MUX_03456 ]
442 }
443 X_BUF #[
444 .LOC { "SLICE_X189V175" }
445 ]
446 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D34MUX {
447 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [34]],
448 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [34], D34MUX_03456 ]
449 }
450 X_BUF #[
451 .LOC { "SLICE_X189V175" }
452 ]
453 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D35MUX {
454 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [35]],
455 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [35], D35MUX_03456 ]
456 }
457 X_BUF #[
458 .LOC { "SLICE_X189V175" }
459 ]
460 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D36MUX {
461 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [36]],
462 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [36], D36MUX_03456 ]
463 }
464 X_BUF #[
465 .LOC { "SLICE_X189V175" }
466 ]
467 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D37MUX {
468 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [37]],
469 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [37], D37MUX_03456 ]
470 }
471 X_BUF #[
472 .LOC { "SLICE_X189V175" }
473 ]
474 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D38MUX {
475 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [38]],
476 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [38], D38MUX_03456 ]
477 }
478 X_BUF #[
479 .LOC { "SLICE_X189V175" }
480 ]
481 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D39MUX {
482 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [39]],
483 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [39], D39MUX_03456 ]
484 }
485 X_BUF #[
486 .LOC { "SLICE_X189V175" }
487 ]
488 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D40MUX {
489 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [40]],
490 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [40], D40MUX_03456 ]
491 }
492 X_BUF #[
493 .LOC { "SLICE_X189V175" }
494 ]
495 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D41MUX {
496 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [41]],
497 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [41], D41MUX_03456 ]
498 }
499 X_BUF #[
500 .LOC { "SLICE_X189V175" }
501 ]
502 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D42MUX {
503 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [42]],
504 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [42], D42MUX_03456 ]
505 }
506 X_BUF #[
507 .LOC { "SLICE_X189V175" }
508 ]
509 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D43MUX {
510 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [43]],
511 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [43], D43MUX_03456 ]
512 }
513 X_BUF #[
514 .LOC { "SLICE_X189V175" }
515 ]
516 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D44MUX {
517 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [44]],
518 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [44], D44MUX_03456 ]
519 }
520 X_BUF #[
521 .LOC { "SLICE_X189V175" }
522 ]
523 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D45MUX {
524 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [45]],
525 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [45], D45MUX_03456 ]
526 }
527 X_BUF #[
528 .LOC { "SLICE_X189V175" }
529 ]
530 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D46MUX {
531 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [46]],
532 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [46], D46MUX_03456 ]
533 }
534 X_BUF #[
535 .LOC { "SLICE_X189V175" }
536 ]
537 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D47MUX {
538 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [47]],
539 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [47], D47MUX_03456 ]
540 }
541 X_BUF #[
542 .LOC { "SLICE_X189V175" }
543 ]
544 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D48MUX {
545 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [48]],
546 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [48], D48MUX_03456 ]
547 }
548 X_BUF #[
549 .LOC { "SLICE_X189V175" }
550 ]
551 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D49MUX {
552 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [49]],
553 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [49], D49MUX_03456 ]
554 }
555 X_BUF #[
556 .LOC { "SLICE_X189V175" }
557 ]
558 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D50MUX {
559 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [50]],
560 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [50], D50MUX_03456 ]
561 }
562 X_BUF #[
563 .LOC { "SLICE_X189V175" }
564 ]
565 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D51MUX {
566 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [51]],
567 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [51], D51MUX_03456 ]
568 }
569 X_BUF #[
570 .LOC { "SLICE_X189V175" }
571 ]
572 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D52MUX {
573 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [52]],
574 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [52], D52MUX_03456 ]
575 }
576 X_BUF #[
577 .LOC { "SLICE_X189V175" }
578 ]
579 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D53MUX {
580 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [53]],
581 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [53], D53MUX_03456 ]
582 }
583 X_BUF #[
584 .LOC { "SLICE_X189V175" }
585 ]
586 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D54MUX {
587 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [54]],
588 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [54], D54MUX_03456 ]
589 }
590 X_BUF #[
591 .LOC { "SLICE_X189V175" }
592 ]
593 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D55MUX {
594 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [55]],
595 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [55], D55MUX_03456 ]
596 }
597 X_BUF #[
598 .LOC { "SLICE_X189V175" }
599 ]
600 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D56MUX {
601 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [56]],
602 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [56], D56MUX_03456 ]
603 }
604 X_BUF #[
605 .LOC { "SLICE_X189V175" }
606 ]
607 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D57MUX {
608 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [57]],
609 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [57], D57MUX_03456 ]
610 }
611 X_BUF #[
612 .LOC { "SLICE_X189V175" }
613 ]
614 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D58MUX {
615 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [58]],
616 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [58], D58MUX_03456 ]
617 }
618 X_BUF #[
619 .LOC { "SLICE_X189V175" }
620 ]
621 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D59MUX {
622 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [59]],
623 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [59], D59MUX_03456 ]
624 }
625 X_BUF #[
626 .LOC { "SLICE_X189V175" }
627 ]
628 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D60MUX {
629 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [60]],
630 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [60], D60MUX_03456 ]
631 }
632 X_BUF #[
633 .LOC { "SLICE_X189V175" }
634 ]
635 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D61MUX {
636 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [61]],
637 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [61], D61MUX_03456 ]
638 }
639 X_BUF #[
640 .LOC { "SLICE_X189V175" }
641 ]
642 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D62MUX {
643 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [62]],
644 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [62], D62MUX_03456 ]
645 }
646 X_BUF #[
647 .LOC { "SLICE_X189V175" }
648 ]
649 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D63MUX {
650 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [63]],
651 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [63], D63MUX_03456 ]
652 }
653 X_BUF #[
654 .LOC { "SLICE_X189V175" }
655 ]
656 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D64MUX {
657 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [64]],
658 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [64], D64MUX_03456 ]
659 }
660 X_BUF #[
661 .LOC { "SLICE_X189V175" }
662 ]
663 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D65MUX {
664 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [65]],
665 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [65], D65MUX_03456 ]
666 }
667 X_BUF #[
668 .LOC { "SLICE_X189V175" }
669 ]
670 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D66MUX {
671 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [66]],
672 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [66], D66MUX_03456 ]
673 }
674 X_BUF #[
675 .LOC { "SLICE_X189V175" }
676 ]
677 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D67MUX {
678 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [67]],
679 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [67], D67MUX_03456 ]
680 }
681 X_BUF #[
682 .LOC { "SLICE_X189V175" }
683 ]
684 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D68MUX {
685 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [68]],
686 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [68], D68MUX_03456 ]
687 }
688 X_BUF #[
689 .LOC { "SLICE_X189V175" }
690 ]
691 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D69MUX {
692 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [69]],
693 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [69], D69MUX_03456 ]
694 }
695 X_BUF #[
696 .LOC { "SLICE_X189V175" }
697 ]
698 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D70MUX {
699 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [70]],
700 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [70], D70MUX_03456 ]
701 }
702 X_BUF #[
703 .LOC { "SLICE_X189V175" }
704 ]
705 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D71MUX {
706 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [71]],
707 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [71], D71MUX_03456 ]
708 }
709 X_BUF #[
710 .LOC { "SLICE_X189V175" }
711 ]
712 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D72MUX {
713 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [72]],
714 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [72], D72MUX_03456 ]
715 }
716 X_BUF #[
717 .LOC { "SLICE_X189V175" }
718 ]
719 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D73MUX {
720 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [73]],
721 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [73], D73MUX_03456 ]
722 }
723 X_BUF #[
724 .LOC { "SLICE_X189V175" }
725 ]
726 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D74MUX {
727 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [74]],
728 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [74], D74MUX_03456 ]
729 }
730 X_BUF #[
731 .LOC { "SLICE_X189V175" }
732 ]
733 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D75MUX {
734 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [75]],
735 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [75], D75MUX_03456 ]
736 }
737 X_BUF #[
738 .LOC { "SLICE_X189V175" }
739 ]
740 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D76MUX {
741 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [76]],
742 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [76], D76MUX_03456 ]
743 }
744 X_BUF #[
745 .LOC { "SLICE_X189V175" }
746 ]
747 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D77MUX {
748 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [77]],
749 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [77], D77MUX_03456 ]
750 }
751 X_BUF #[
752 .LOC { "SLICE_X189V175" }
753 ]
754 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D78MUX {
755 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [78]],
756 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [78], D78MUX_03456 ]
757 }
758 X_BUF #[
759 .LOC { "SLICE_X189V175" }
760 ]
761 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D79MUX {
762 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [79]],
763 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [79], D79MUX_03456 ]
764 }
765 X_BUF #[
766 .LOC { "SLICE_X189V175" }
767 ]
768 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D80MUX {
769 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [80]],
770 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [80], D80MUX_03456 ]
771 }
772 X_BUF #[
773 .LOC { "SLICE_X189V175" }
774 ]
775 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D81MUX {
776 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [81]],
777 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [81], D81MUX_03456 ]
778 }
779 X_BUF #[
780 .LOC { "SLICE_X189V175" }
781 ]
782 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D82MUX {
783 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [82]],
784 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [82], D82MUX_03456 ]
785 }
786 X_BUF #[
787 .LOC { "SLICE_X189V175" }
788 ]
789 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D83MUX {
790 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [83]],
791 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [83], D83MUX_03456 ]
792 }
793 X_BUF #[
794 .LOC { "SLICE_X189V175" }
795 ]
796 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D84MUX {
797 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [84]],
798 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [84], D84MUX_03456 ]
799 }
800 X_BUF #[
801 .LOC { "SLICE_X189V175" }
802 ]
803 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D85MUX {
804 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [85]],
805 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [85], D85MUX_03456 ]
806 }
807 X_BUF #[
808 .LOC { "SLICE_X189V175" }
809 ]
810 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D86MUX {
811 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [86]],
812 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [86], D86MUX_03456 ]
813 }
814 X_BUF #[
815 .LOC { "SLICE_X189V175" }
816 ]
817 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D87MUX {
818 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [87]],
819 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [87], D87MUX_03456 ]
820 }
821 X_BUF #[
822 .LOC { "SLICE_X189V175" }
823 ]
824 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D88MUX {
825 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [88]],
826 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [88], D88MUX_03456 ]
827 }
828 X_BUF #[
829 .LOC { "SLICE_X189V175" }
830 ]
831 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D89MUX {
832 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [89]],
833 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [89], D89MUX_03456 ]
834 }
835 X_BUF #[
836 .LOC { "SLICE_X189V175" }
837 ]
838 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D90MUX {
839 .I[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [90]],
840 .O[Vtx_hold_fifo0_fifo0_ctrl0_wb0_ptr [90], D90MUX_03456 ]
841 }
842 X_BUF #[
843 .LOC { "SLICE_X189V175" }
844 ]
845 Vtx_hold_fifo0_fifo0_ctrl0_wb0_D91MUX {
846 .I[V
```

# Evaluating OneSpin Tools for Assurance of 3PIP

## OneSpin Products / Apps



## 3PIP Testcases



Low Effort: 20 minutes

Check Type	Number of Checks	Number of Holds	Number of Fails	Number of Opens
Array Index	792	784	8	0
Truncation	452	36	114	302
Resolution-X	400	0	400	0
Signal-Domain	64	64	0	0
Init	78312	6816	71496	0
Dead-Code	23672	11987	2908	8777
Stick	56170	17374	3876	34920
Integer	2788	2303	14	471
Total	162650	39364	78816	44470

Useful as a first step

Enforces good coding habits

Which of these checks are assurance-relevant?

*DV-Inspect autochecks run on MIT-LL Common Evaluation Platform (CEP) containing Rocket Core. <https://github.com/mit-ll/CEP>*

Normal Effort: 3 hrs, 13 min

Check Type	Number of Checks	Number of Holds	Number of Fails	Number of Opens
Array Index	792	784	8	0
Truncation	452	42 (+6)	124 (+10)	286 (-16)
Resolution-X	400	0	400	0
Signal-Domain	64	64	0	0
Init	78312	6816	71496	0
Dead-Code	23672	11987	3007 (+99)	8678 (-99)
Stick	56170	17374	3876	34920
Integer	2788	2303	14	471
Total	162650	39370 (+6)	78925 (+109)	44355 (-115)

(+/-) compared to Low Effort

Useful as a first step

Enforces good coding habits

Which of these checks are assurance-relevant?

*DV-Inspect autochecks run on MIT-LL Common Evaluation Platform (CEP) containing Rocket Core. <https://github.com/mit-ll/CEP>*

Normal Effort +: 1 day, 12 hrs, 50 min

Check Type	Number of Checks	Number of Holds	Number of Fails	Number of Opens
Array Index	792	784	8	0
Truncation	452	42	124	286
Resolution-X	400	0	400	0
Signal-Domain	64	64	0	0
Init	78312	6816	71496	0
Dead-Code	23672	12398 (+411)	4075 (+1068)	7199 (-1479)
Stick	56170	18099 (+725)	4193 (+317)	33878 (-1042)
Integer	2788	2303	14	471
Total	162650	40506 (+1136)	80310 (+1385)	41834 (-2521)

(+/-) compared to Normal Effort

Useful as a first step

Enforces good coding habits

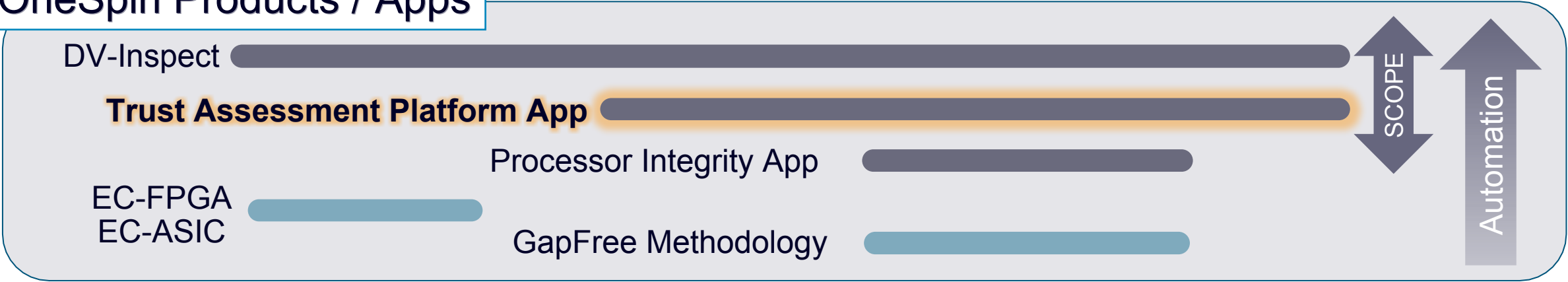
Which of these checks are assurance-relevant?

## What does this mean? When are we done?

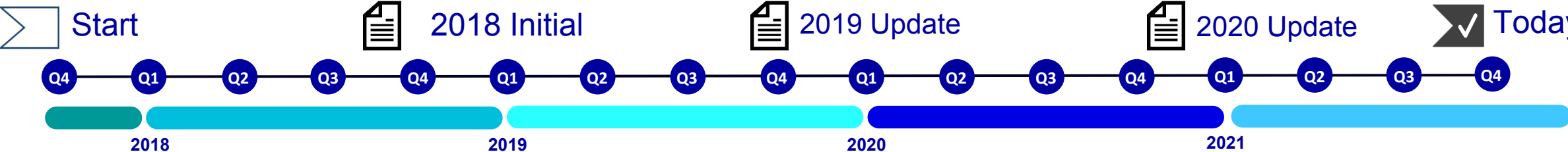
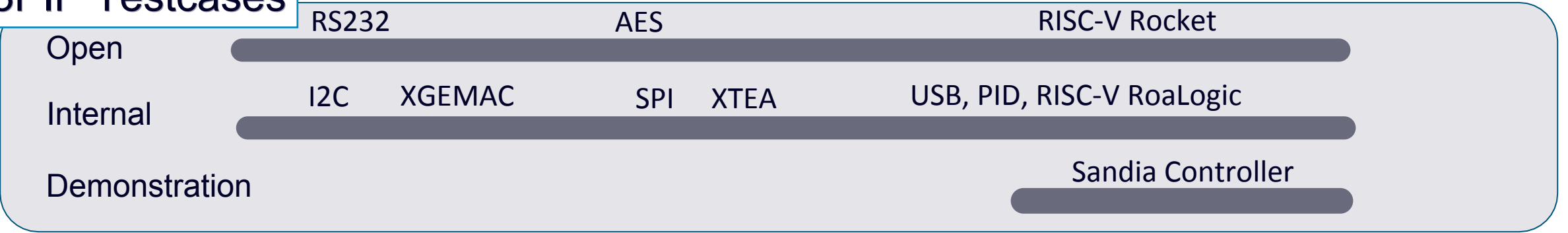
*DV-Inspect autochecks run on MIT-LL Common Evaluation Platform (CEP) containing Rocket Core. <https://github.com/mit-ll/CEP>*

- Reached out to OneSpin to assist with interpretation of results, prover/disprover modifications, targeting only dead-code and stick checks. No finite state machines were automatically recognized.

## OneSpin Products / Apps

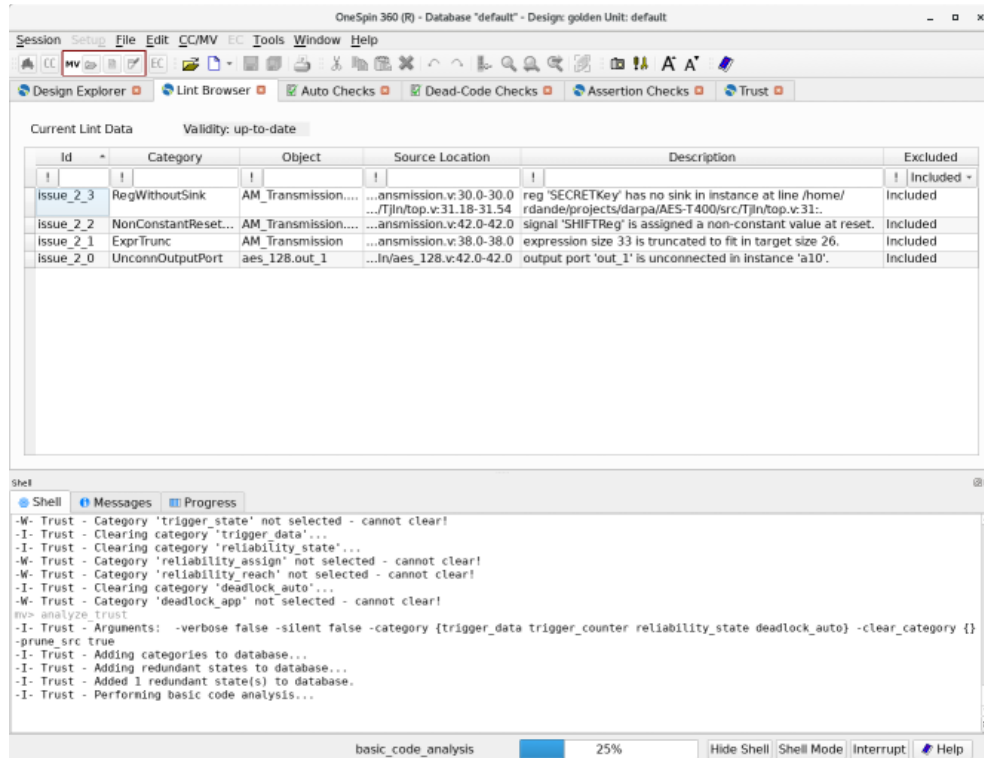


## 3PIP Testcases



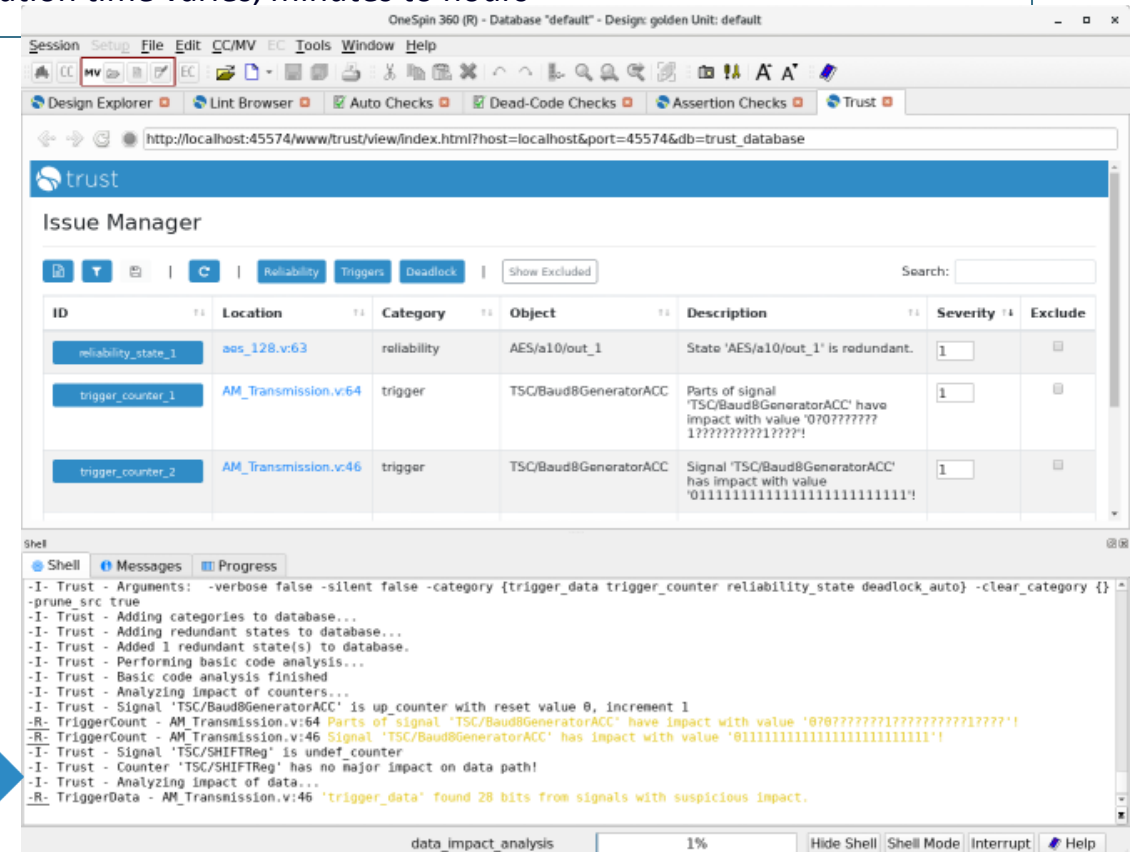
## Lint browser (DV-Inspect)

- basic lint check, automatically performed on read-in design



## analyze\_trust (TAP)

- runs a suite of tests and presents a list of discovered issues
- the user can view the source of issues by clicking the list item
- execution time varies, minutes to hours



## Hardware Trojan R&D

### EDA Tool Evaluation

#### Experimental Results

Source	Name	Runtime	Issues Reported	Trojan Inserted	Automatic Detection
TrustHub*	AES	11 hours	2	Yes	Yes
TrustHub	PIC16	<1 min	72	Yes	Yes
TrustHub	RS232	<1 min	3	Yes	Yes
TrustHub	BasicRSA	<1 min	17	Yes	Yes
GitHub	RISC-V Rocketcore	28 min	12	No	Yes
OneSpin	UART	<1 min	10	Yes	Yes
Aerospace**	SpaceWire	<1 min	3	No/Yes	No
Aerospace	RISC-V Taiga	13 min	46	No/Yes	No
Aerospace	Leon3	6 hours	423	No/Yes	Yes***

\* TrustHub designs averaged results over multiple articles

\*\* Aerospace designs contained 1 golden, 3 with Trojans

\*\*\* Leon3 articles consisted of 3 Trojan designs, 1/3 Trojans discovered

(Results were from OneSpin TAP 2020.2)



- Test suite
  - 90 designs with and without Trojans inserted
  - Size range: 100 to 100K FFs
- Results
  - Representative selection of IP designs shown in table
  - Few trigger-type issues reported
  - Numerous reliability issues reported
  - Very few false alarms
  - Some Trojans have been missed
  - Runtime is short

© 2020 OneSpin Osmosis | Page 18

Our RISC-V results consistent with Aerospace once we synced on same version

TAP helps filter results

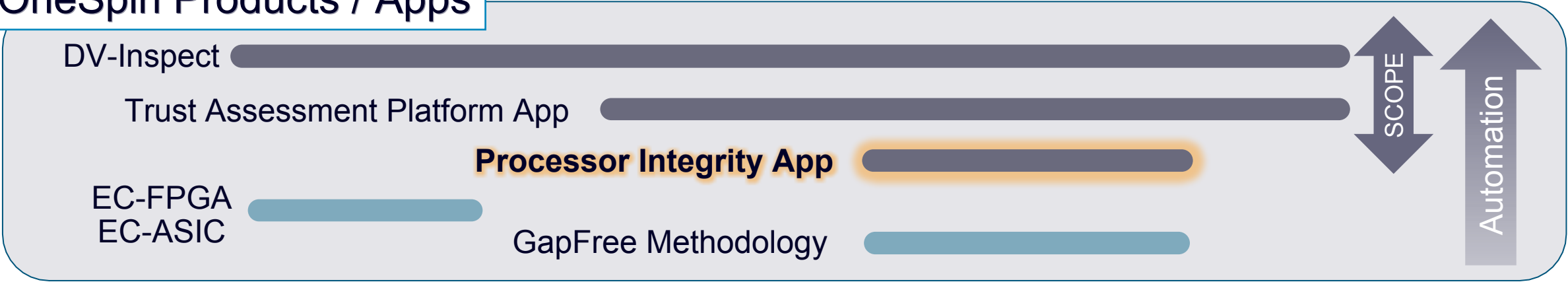
TAP does not accelerate results

Best working with HDL

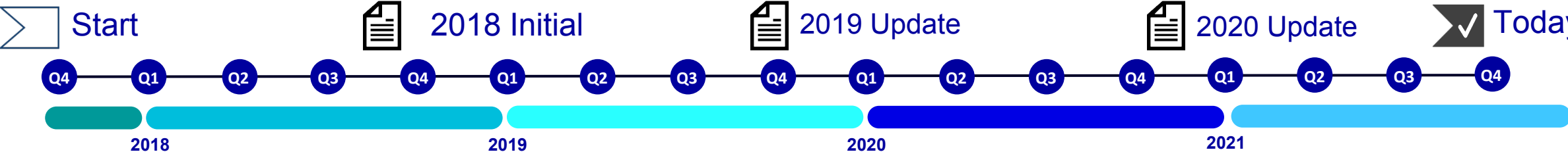
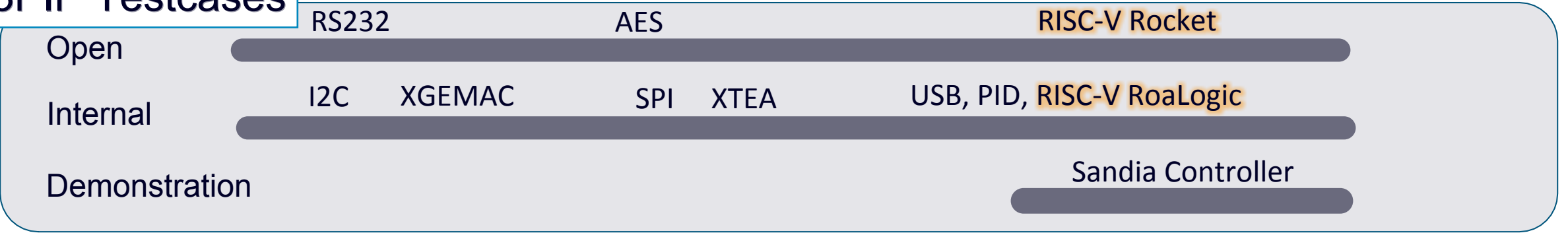
*Evaluation of OneSpin Trust Assessment for Hardware Trojan Detection. Chan, Garrett and Rao, Vikram. Aerospace. Osmosis for DoD 2020.*

# Evaluating OneSpin Tools for Assurance of 3PIP

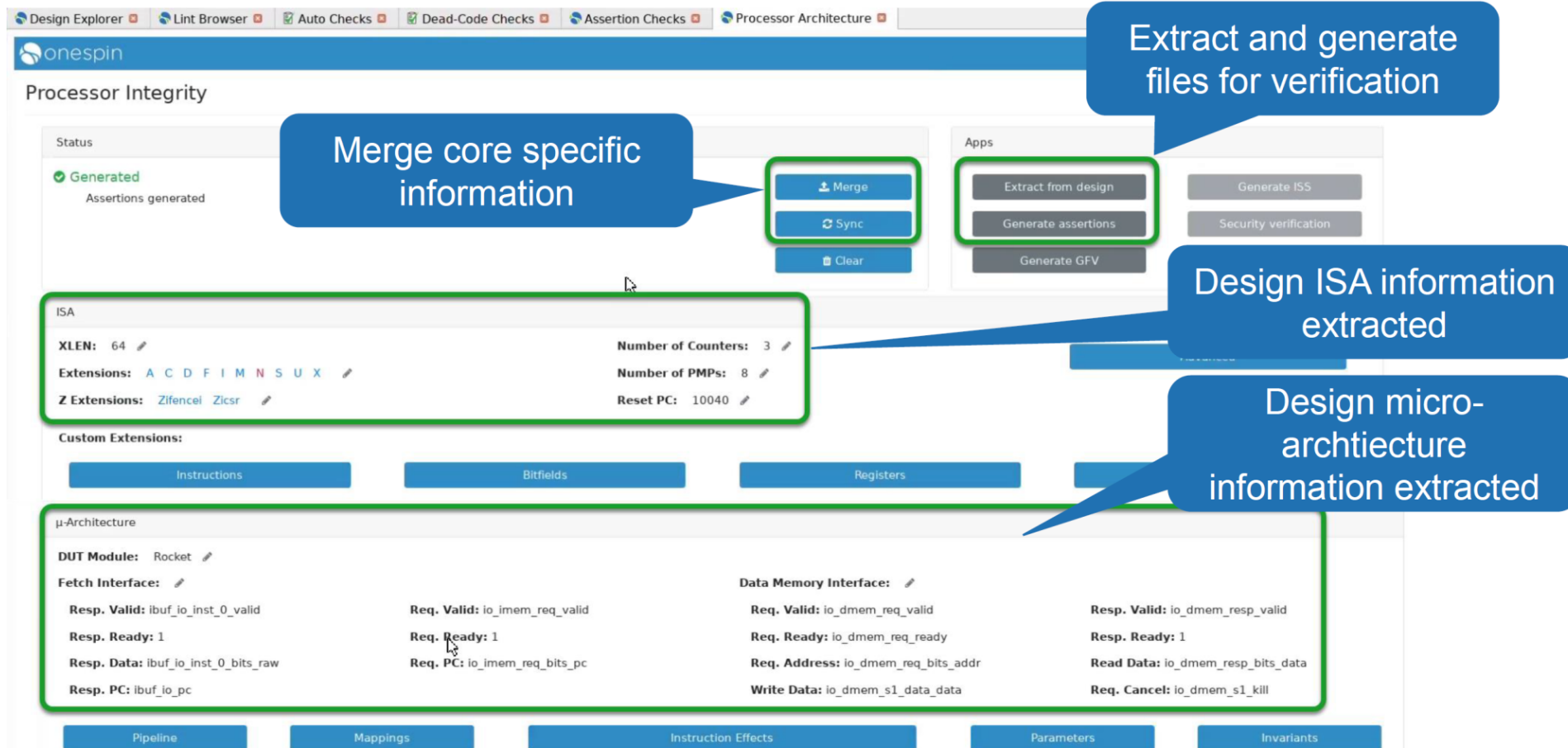
## OneSpin Products / Apps



## 3PIP Testcases



# Processor Integrity App v2020.2



The screenshot shows the Processor Integrity App interface. The top navigation bar includes tabs for Design Explorer, Lint Browser, Auto Checks, Dead-Code Checks, Assertion Checks, and Processor Architecture. The main content area is divided into several sections:

- Status:** Shows a green checkmark and the text "Generated" and "Assertions generated".
- Apps:** Contains buttons for "Extract from design", "Generate assertions", "Generate GFV", "Generate ISS", and "Security verification".
- ISA:** Displays configuration details for XLEN (64), Extensions (A C D F I M N S U X), Z Extensions (ZifenceI Zicsr), Number of Counters (3), Number of PMPs (8), and Reset PC (10040).
- Custom Extensions:** Includes tabs for Instructions, Bitfields, and Registers.
- μ-Architecture:** Shows the DUT Module (Rocket) and Fetch Interface details, including Resp. Valid, Resp. Ready, Resp. Data, and Resp. PC.
- Data Memory Interface:** Displays Req. Valid, Req. Ready, Req. Address, Write Data, Resp. Valid, Resp. Ready, Read Data, and Req. Cancel.

Callouts highlight specific features:

- "Merge core specific information" points to the Merge, Sync, and Clear buttons.
- "Extract and generate files for verification" points to the Extract from design button.
- "Design ISA information extracted" points to the ISA configuration section.
- "Design micro-architecture information extracted" points to the μ-Architecture section.

Open-source RISC-V instruction set architecture and Processor Integrity App enables access to GapFreeVerification™ rigor via automation

Name	Proof Status	Witness Status	Prover	Runtime
! <any status>	! <any status>	! <any status>		
Properties				
RV_chk.ops.bubble_a	hold	pass (1)	approver1:0	00:00:55
RV_chk.ops.flush_a	hold	pass (3)	approver1:0	00:00:25
RV_chk.ops.interrupt_handle_a	fail (7)	pass (2)	disprover1	00:03:34
RV_chk.ops.mispred_a	hold	pass (3)	approver1:0	00:00:13
RV_chk.ops.RV32A.all_a	hold	pass (2)	approver1:0	00:01:22
RV_chk.ops.RV32D.all_a	hold	pass (7)	approver1:0	00:01:45
RV_chk.ops.RV32F.all_a	hold	pass (7)	approver1:0	00:01:05
RV_chk.ops.RV32I.ADD_a	hold	pass (2)	approver1:0	00:04:31
RV_chk.ops.RV32I.Arith_a	hold	pass (2)	approver1:0	00:07:30
RV_chk.ops.RV32I.AUIPC_a	hold	pass (2)	approver1:0	00:01:00
RV_chk.ops.RV32I.Branch_a	hold	pass (2)	approver1:0	00:01:00
RV_chk.ops.RV32I.CallBreak_a	fail (7)	pass (2)	disprover1	00:06:09
RV_chk.ops.RV32I.CSR_a	fail (2)	pass (2)	disprover3	00:02:11
RV_chk.ops.RV32I.FENCE_a	hold	pass (2)	approver1:0	00:00:34
RV_chk.ops.RV32I.FENCE_I_a	fail (2)	pass (2)	disprover3	00:01:28
RV_chk.ops.RV32I.Jump_a	hold	pass (2)	approver1:0	00:02:52
RV_chk.ops.RV32I.LUI_a	hold	pass (2)	approver1:0	00:00:46
RV_chk.ops.RV32I.Mem_a	hold	pass (2)	approver1:0	00:00:34
RV_chk.ops.RV32I.RET_a	hold	pass (2)	approver1:0	00:00:34
RV_chk.ops.RV32I.Supervisor.SFENCE_VMA_a	hold	pass (2)	approver1:0	00:01:14
RV_chk.ops.RV32M.all_a	hold	pass (2)	approver1:0	00:01:27
RV_chk.ops.RV64A.all_a	hold	pass (2)	approver1:0	00:01:13
RV_chk.ops.RV64D.all_a	hold	pass (7)	approver1:0	00:01:06
RV_chk.ops.RV64F.all_a	hold	pass (7)	approver1:0	00:00:53
RV_chk.ops.RV64I.Arith_a	hold	pass (2)	approver1:0	00:03:41
RV_chk.ops.RV64I.Mem_a	hold	pass (2)	approver1:0	00:02:18
RV_chk.ops.RV64M.all_a	hold	pass (2)	approver1:0	00:01:14
RV_chk.ops.RVC.Arith_a	hold	pass (2)	approver1:0	00:16:50
RV_chk.ops.RVC.Branch_a	hold	pass (2)	approver1:0	00:00:45
RV_chk.ops.RVC.Jump_a	hold	pass (2)	approver1:0	00:01:05
RV_chk.ops.RVC.Mem_a	hold	pass (2)	approver1:0	00:01:53
RV_chk.ops.replay_a	hold	pass (2)	approver1:0	00:00:35
RV_chk.ops.replay_mem_a	hold	pass (2)	approver1:0	00:02:19
RV_chk.ops.replay_wb_a	hold	pass (2)	approver1:0	00:01:55
RV_chk.ops.reset_a	hold	pass (1)	approver1:0	00:00:11
RV_chk.ops.xcpt_fetch_dec_a	fail (2)	pass (2)	disprover3	00:01:52
RV_chk.ops.xcpt_mem_a	fail (12)	pass (7)	disprover1	00:31:15
RV_chk.ops.xcpt_wb_a	fail (7)	pass (2)	disprover1	00:14:14

Semi-automated to get to this point  
Extract and map ISA registers,  
custom extensions, exceptions,  
interrupts

## SUPPORTED

### Rocket / MIT-LL CEP

<https://github.com/mit-ll/CEP>

- OneSpin had previously identified bugs contained in our version
  - Illegal instruction exception not raised when expected\*\*
- Our CEP (v2.2) implementation had outdated Rocket Core
  - Issues have been closed since this finding

→ ***Identified processor integrity and provenance tracking issues***

## NOT YET SUPPORTED

### Roa Logic

<https://github.com/RoaLogic/RV12>

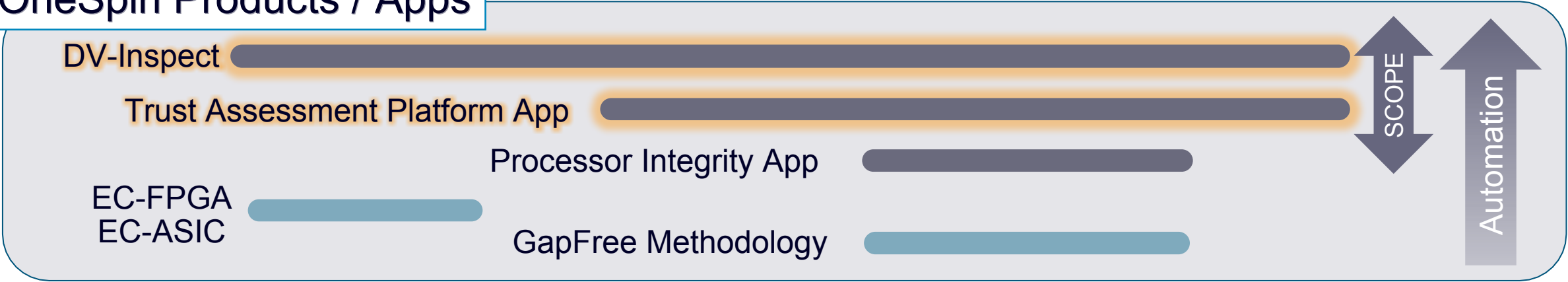
- Full automated extraction not possible
- Custom JSON configuration file needed
- Modifications to internal SV files needed
- Did not pursue further

→ ***Others should expect to work with OneSpin for new processor configurations***

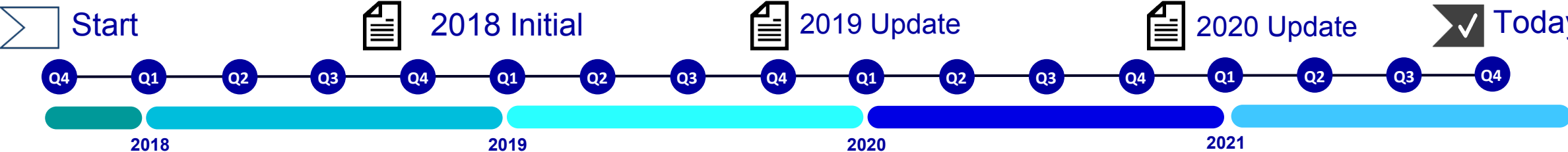
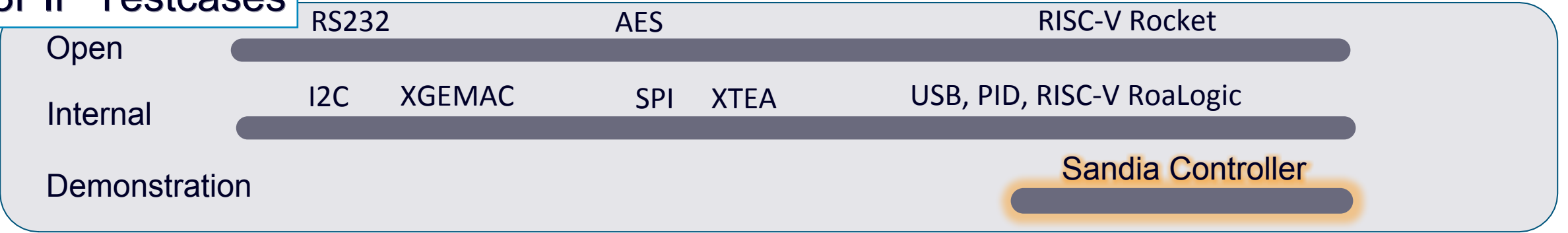
\*\* <https://github.com/mit-ll/CEP/issues/8>  
<https://github.com/chipsalliance/rocket-chip/issues/1861>  
<https://github.com/chipsalliance/rocket-chip/issues/1949>

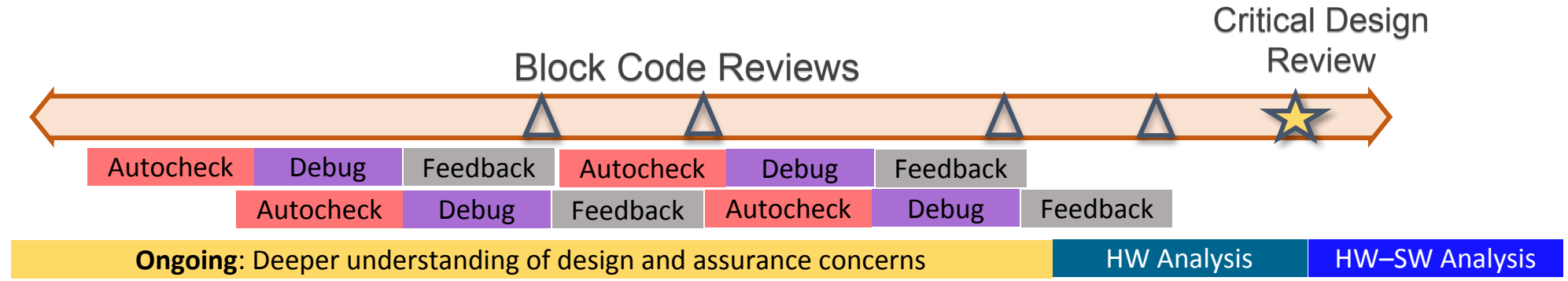
# Evaluating OneSpin Tools for Assurance of 3PIP

## OneSpin Products / Apps



## 3PIP Testcases





## Three teams: Design, Verification, and Independent Assessment

### Autochecks save time, but should not define “done”

- Need tips and tricks for getting through dead-code and FSM analysis.

### Formal verification does HW integrity really well

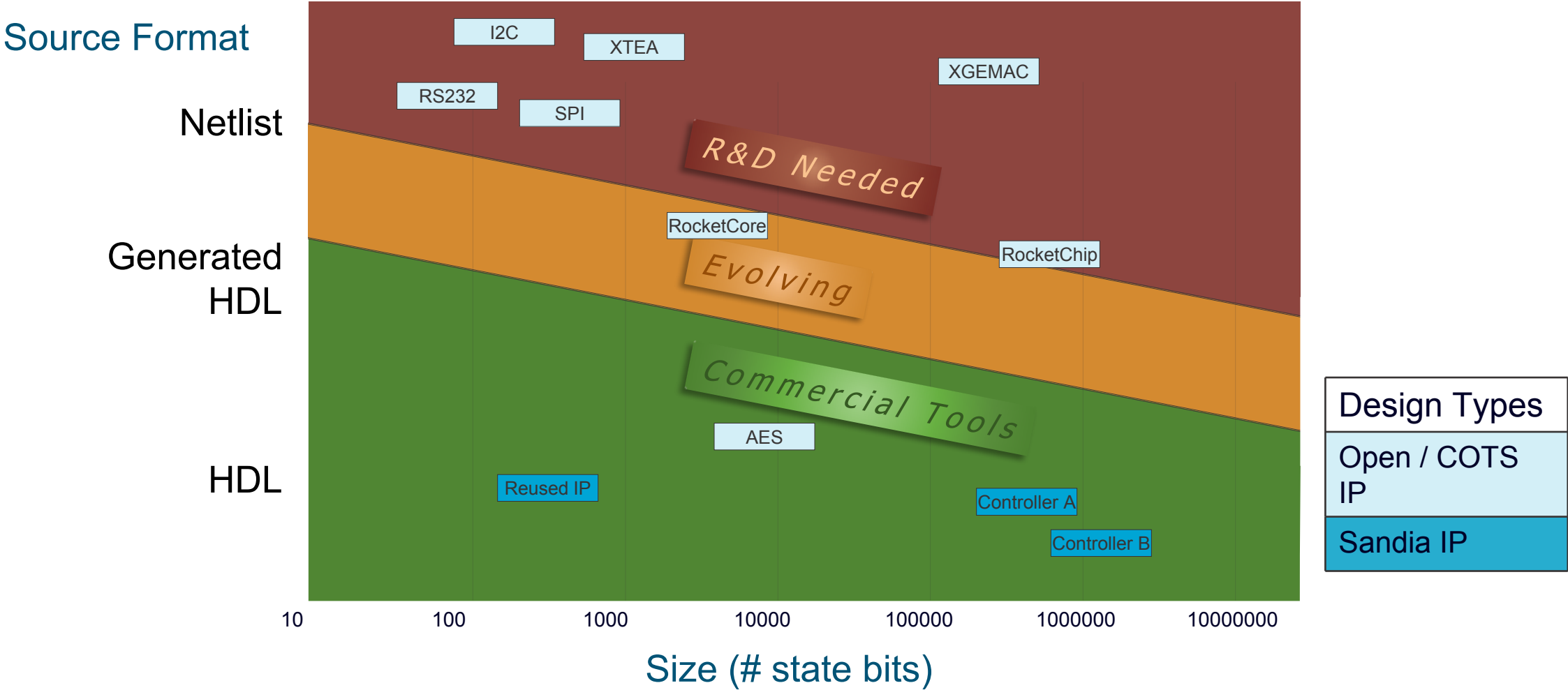
- **Caveats:** Many ways to configure 3PIP to violate constraints, assumptions. Need capabilities for HW-SW analysis.

### Level of experience and training matters

- We were more effective and knowledgeable of design as time when on. Don't always have that time.

### Application and integration of IP matters

# State of the Practice for 3PIP Assurance





## Track Developments

Accellera Systems Initiative:  
Security Assurance for Electronic  
Design Integration Standard  
(rev 1.0, July 2021)



## Target Relevant 3PIP

Inform best practices for  
Quantifiable Assurance (5200.xx)



## Evaluate Tools

Incorporate tools into assessment  
workflows

# Joint Federated Assurance Center (JFAC)



The JFAC is a federation of DoD organizations that promotes and enables software and hardware assurance by providing expertise and support to defense acquisition programs and supporting activities.

JFAC Portal:

<https://jfac.navy.mil>

SharePoint Site:

<https://intelshare.intelink.gov/sites/jfac>

# Contact

**Vivian Guzman Kammler | R&D Cybersecurity Lead | Sandia National Laboratories**  
[vgkamml@sandia.gov](mailto:vgkamml@sandia.gov) | +1 505 284 3528