Data-Triggered Breakpoint for In-Circuit Debug without Re-implementation

* Yutaka Tamiya Yoshinori Tomita Toshiyuki Ichiba Kaoru Kawamura

Fujitsu Laboratories Ltd., Japan

- Implemented with Simple HW
  Linearity of CRC
  (Cyclic Redundancy Check)

Realize Fast & Flexible Debug

- At-Speed & Realtime
- With Small Area Overhead
- No Re-implementation

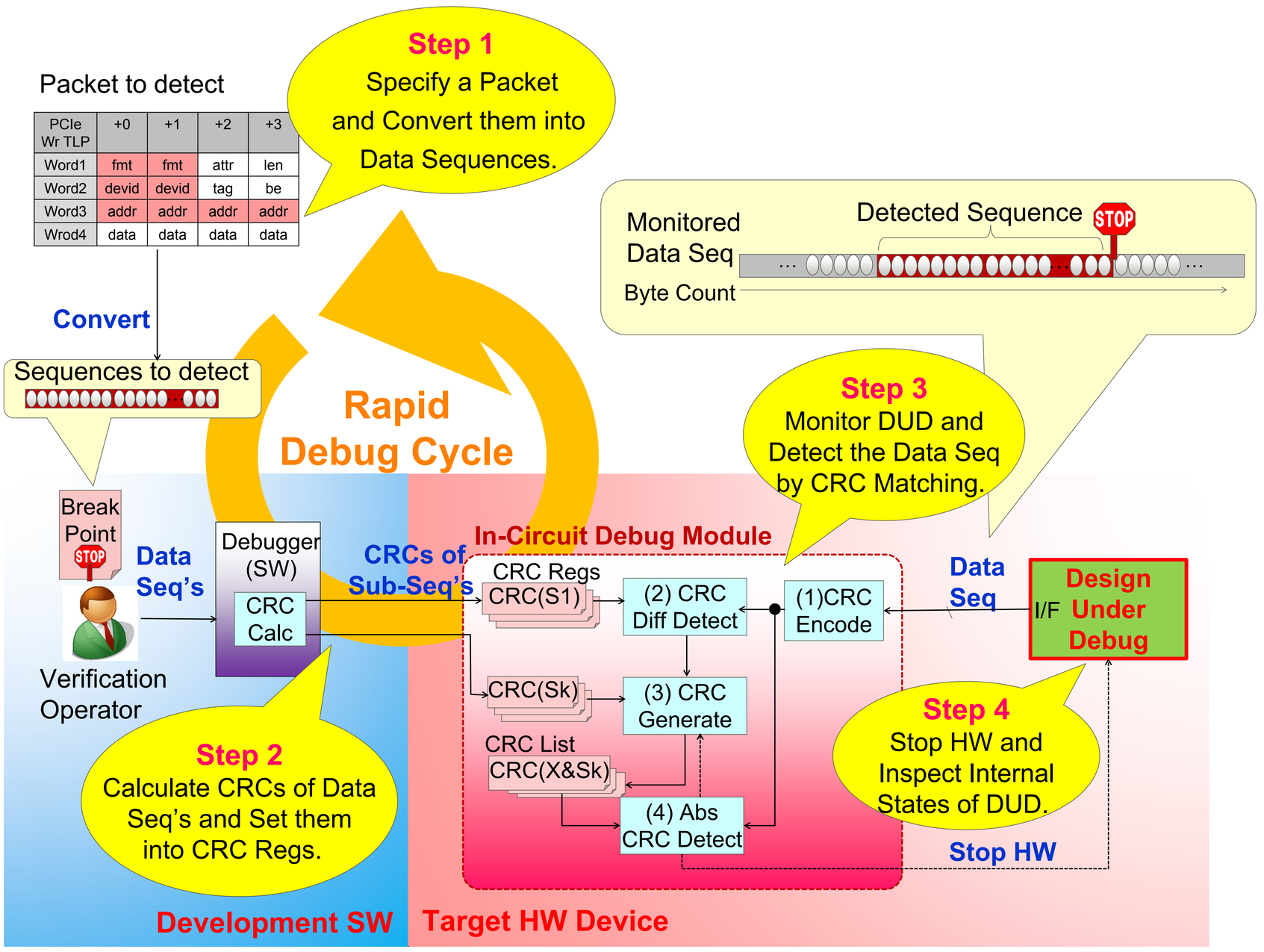
Background

Problems of Post-Silicon Debug

- Logical Bugs remain even after Pre-Silicon.
  Impractical Simulation with exhaustive Test Patterns.
- Needs much Efforts to Detect & Fix Bugs.
  Poor Controllability & Observability
- Takes Long TAT w/ In-Circuit Logic Analyzer
  : ChipScope, SignalTap II, etc.
  HW re-implementation required on changing Breakpoints.

Growth of Hardware: Size, Speed, Func.

- Ethernet=40~100Gbps, PCIe=5~8Gbps/Lane
- Almost all systems consist of HW & SW



Algorithms to Detect Data Sequences

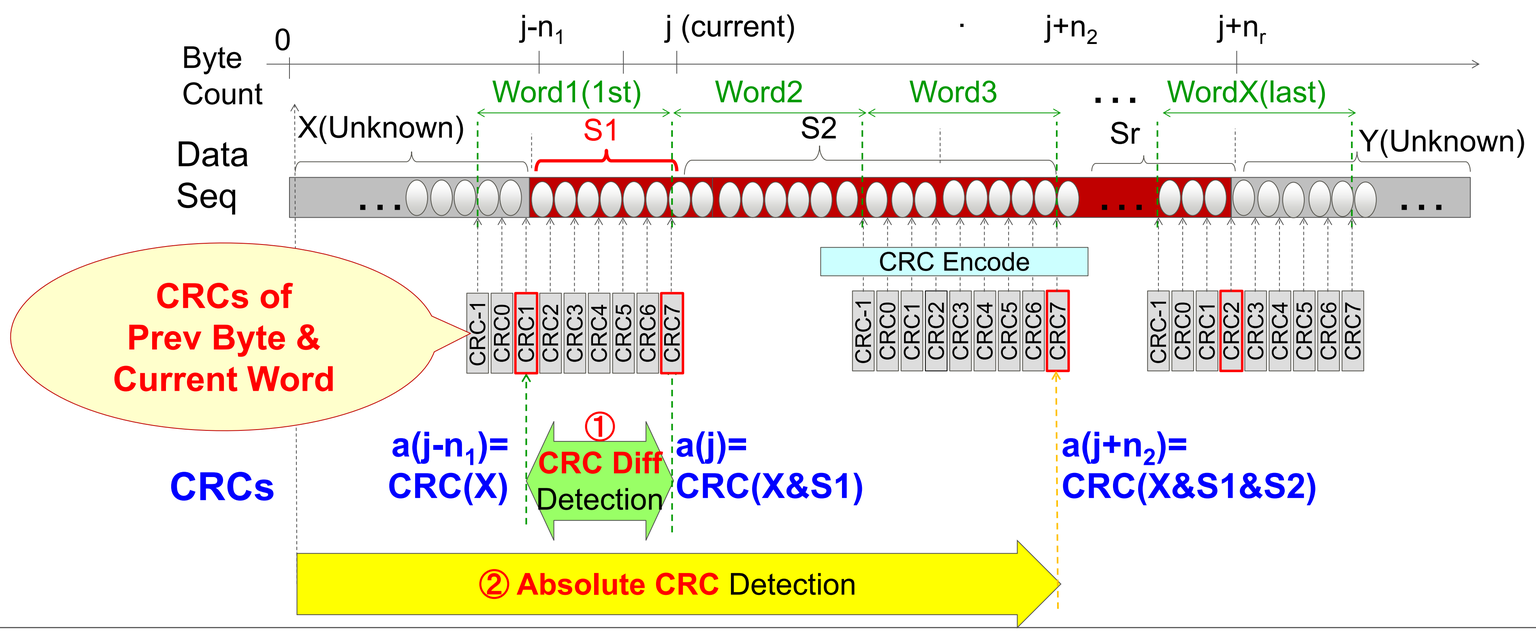
- Divide **Data Seq. *S*** (arbitrary length of ***n***) into
  Sub-Seq's: *S1, S2, … Sr* (Len of Sub-Seq's: n_1 , n_2, …, n_r)
  - *S* = *S1* & *S2* & … & *Sr*
- ***S1*** : The 1st Sub-Sequence included in the 1st Word.
  ➔ ① **CRC Difference Detection**.
- ***Sk*'s** : The 2nd and Latter Sub-Sequence's (*k=2,3,…,r*) with arbitrary Lengths n_2 , …, n_r.
  ➔ ② **Absolute CRC Detection**.

①CRC Diff Detection for S1

- Try to detect ***S1*** in every word:

②Abs CRC Detection for S2

- ***S2***, following ***S1***, can be detected by:



Results of Resource Usage

- Synthesized with Xilinx Kintex7 @Clock 250MHz by Xilinx ISE Tool.
- Small Area Overheads with 4 or less of CRC Width.

Word Width =8	#LUTs						
CRC Width	(1) CRC Encode	(2) CRC Diff	(3)CRC Generate	(4) Abs CRC	Others	Total (Usage of Kintex7-410T)	
CRC-2	16	83	100	33	60	292	(0.11%)
CRC-4	43	152	106	83	0	384	(0.15%)
CRC-8(*)	96	987	103	128	0	1,314	(0.52%)
CRC-16(*)	244	422	3,565	304	0	4,535	(1.78%)
CRC-32(*)	519	779	10,571	600	0	12,469	(4.91%)
CRC-64(*)	141	1,496	8,790	713	108	11,248	(4.42%)

* Failed to implement with clock of 250MHz

Results of False Positives

- False Positives can be reduced by CRC-Width, # of Sub-Sequences, and Word Width.
- Data Seq = 595 MB JPEG stream (as Random Data)
- A 64-byte "SOS" sequence to be detected

CRC Width (polynomial)	# of Sub-Seq's	Word Width (Max Depth of CRC List) 1		2		4		8	
CRC-2	2	38,380,954	(58)	2,368,503	(15)	9,244	(5)	43	(3)
(0x3)	4	2,432195	(28)	9,167	(9)	0	(4)	0	(3)
	8	9,106	(18)	1	(7)	0	(4)	0	(2)
CRC-4	2	2,435,408	(54)	9,160	(6)	18	(2)	8	(2)
(0x3)	4	9,245	(22)	0	(5)	0	(2)	0	(2)
	8	0	(9)	0	(4)	0	(2)	0	(2)
CRC-8	2	7,654	(51)	1	(2)	1	(2)	0	(2)
(0xb1)	4	0	(21)	0	(2)	0	(2)	0	(2)
	8	0	(9)	0	(2)	0	(1)	0	(1)
CRC-16	2	34	(51)	0	(2)	0	(2)	0	(2)
(0xe613)	4	0	(21)	0	(2)	0	(2)	0	(2)
	8	0	(9)	0	(2)	0	(1)	0	(1)
CRC-32	2	0	(51)	0	(2)	0	(2)	0	(2)
(0xedb88320)	4	0	(21)	0	(2)	0	(2)	0	(2)
	8	0	(9)	0	(2)	0	(1)	0	(1)
CRC-64	2	0	(51)	0	(2)	0	(2)	0	(2)
(0xd800000000000000)	4	0	(21)	0	(2)	0	(2)	0	(2)
	8	0	(9)	0	(2)	0	(1)	0	(1)

(Appendix) Linearity of CRC

$$\begin{aligned} CRC(X \,\&\, S1) &= CRC((X \,\&\, 0^m) + (0^p \,\&\, S1)) \\ &= CRC(X \,\&\, 0^m) + CRC(0^p \,\&\, S1) \\ &= CRC(X \,\&\, 0^m) + CRC(S1) \\ &= H(m) \cdot CRC(X) + CRC(S1) \end{aligned}$$

- **+** : XOR, 0^m : m-times consecutive 0 Data.
- **p** : the Length of unknown Sequence X.
  Pre-fixed 0's can be ignored in CRC calculation.
  ∵ $CRC(0^p \,\&\, S1) = CRC(S1)$
- **H(m)** : a constant **Hamming Matrix**, which shifts CRC by m-clocks.
  It acts as if the Data Sequence is followed by 0^m .

Future Work

- Investigate more complex Breakpoint Conditions w/ closer Collaboration between Debugger(SW) & Debug Module(HW)
- Combine our breakpoints w/ other Observability Technique and Realize a practical Debug Tool.