riscure

driving your security forward

FAULT INJECTION – INSIGHTS FOR FUTURE PRODUCT PLANNING

Pascal van Gimst

AGENDA

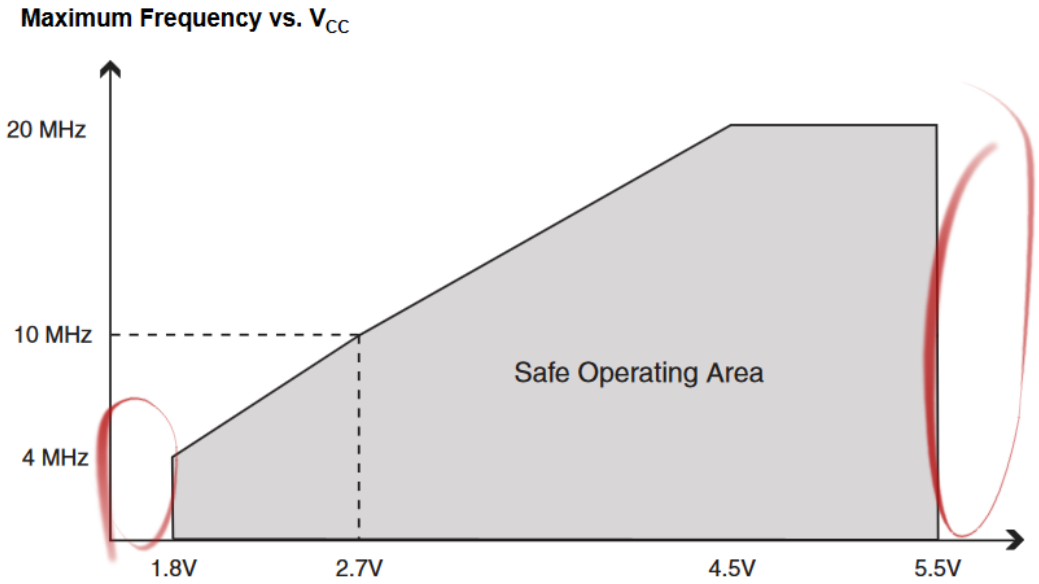
- What is fault injection?
- Why is it relevant for the automotive industry?
- Future product planning

WHAT IS FAULT INJECTION?

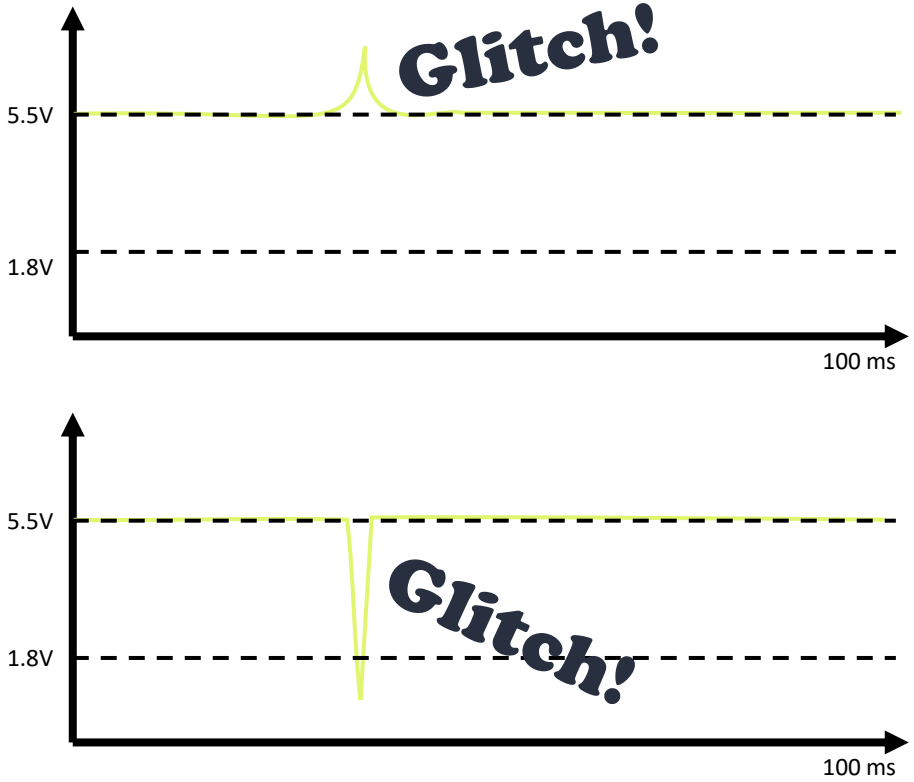
- Manipulating the normal operating environment of a chip to change the program flow of the software running on the chip.

- 3 main types of FI techniques:
  - Voltage or clock glitching
  - ElectroMagnetic glitching
  - Optical (i.e. laser) glitching

Voltage glitching

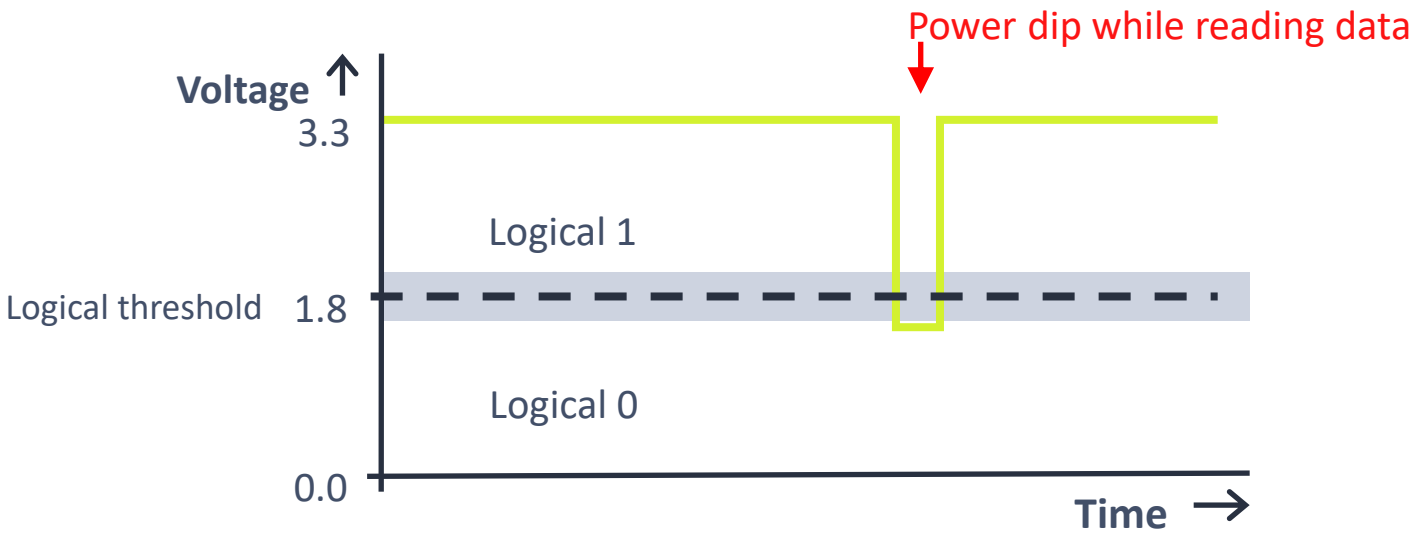


Voltage glitching



Power dip below the threshold

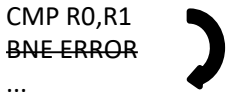
- Inject fault through glitches on e.g. power supply (VCC)



Effects of glitches

Flipping bits

Preventing R/W



Skipping instructions

Enabling/disabling functionality

Chip Destruction

What about industry standards ?

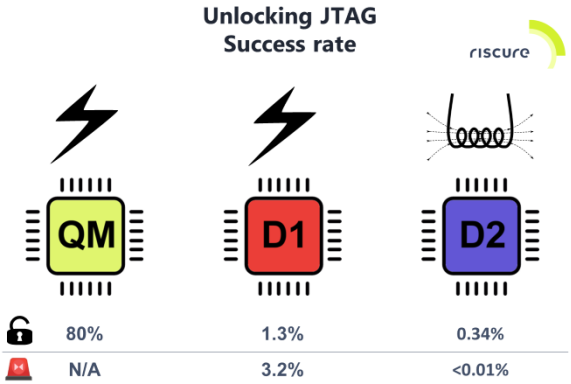
- Does UN R155/156, ISO21434, SAE J3061 or other standards require FI?
- Does type approval have any requirements for FI?
- Now or in the near future?

Why is this relevant?

Safety != Security

On the resilience of ASIL-D certified microcontrollers against fault injection attacks

Nils Wiersma, Ramiro Pareja
Riscure Security Lab
Email: {wiersma, pareja} @ riscure.com



Attacking AUTOSAR using Software and Hardware Attacks

Pascal Nasahl
Graz University of Technology
Graz, Austria
pascal.nasahl@iaik.tugraz.at

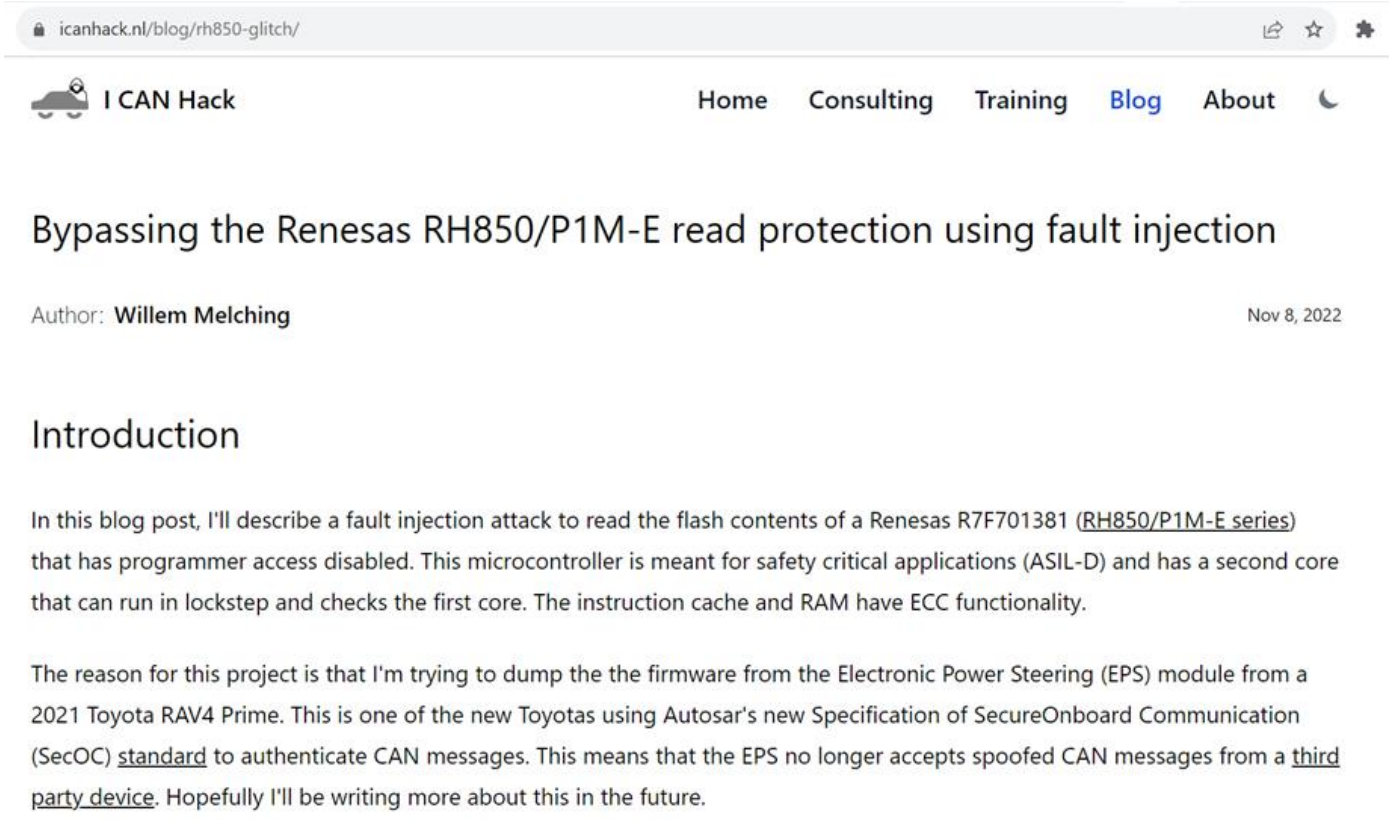
Niek Timmers
Riscure - Security Lab
Delft, The Netherlands
timmers@riscure.com

***Abstract*— The AUTomotive Open System ARchitecture (AUTOSAR) development partnership is a world-wide initiative aiming to jointly develop and establish an open industry standard for automotive E/E software architectures. This standard is rapidly being adopted by the automotive industry and therefore it is important to understand the attack surface of AUTOSAR-based electronic control units (ECU). In this paper we describe several scenarios how software and hardware attacks can compromise the security of AUTOSAR-based ECUs. We consider an attacker with physical access to the ECU who is capable of exploiting both software and hardware vulnerabilities. We discuss how an attacker can use different attack techniques to exploit these vulnerabilities. Moreover, we describe a case study in full detail where we execute arbitrary code on an AUTOSAR-based demonstration ECU by performing a voltage fault injection attack on the AUTOSAR communication stack. Several automotive threats may materialize if an attacker is able to execute arbitrary code on an ECU. For example, it will be possible to persistently modify the ECU's functionality if its code is not authenticated using secure boot.**

binary software is available). During our research we did not have a real AUTOSAR-based ECU software available. We used a freely available AUTOSAR software stack [22] in order to create a demonstration platform. We describe several scenarios how software vulnerabilities may be introduced into an AUTOSAR-based ECU.

An attacker may resort to other types of attacks when software vulnerabilities are unknown. We describe several of these attacks, which range from simple PCB-level attacks (e.g. debug interfaces) to more advanced hardware attacks like fault injection. The results of these hardware attacks may end up in control of the ECU and/or extraction of (secured) information. Additionally, if no proper code signing mechanism (i.e. secure boot) is implemented, it may also lead to modification of the ECU's firmware. This is not an unlikely scenario as it is often believed secure boot is not required for devices that store and execute code only from internal memories.

Why is this relevant?



icanhack.nl/blog/rh850-glitch/

I CAN Hack

Home Consulting Training Blog About

Bypassing the Renesas RH850/P1M-E read protection using fault injection

Author: **Willem Melching**

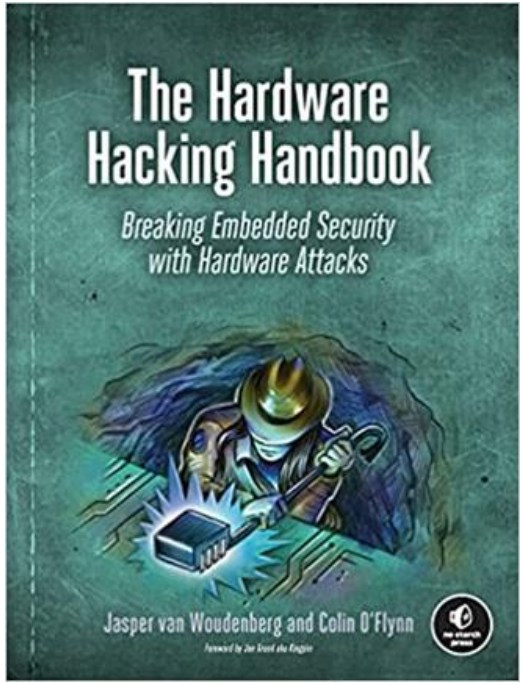
Nov 8, 2022

Introduction

In this blog post, I'll describe a fault injection attack to read the flash contents of a Renesas R7F701381 (RH850/P1M-E series) that has programmer access disabled. This microcontroller is meant for safety critical applications (ASIL-D) and has a second core that can run in lockstep and checks the first core. The instruction cache and RAM have ECC functionality.

The reason for this project is that I'm trying to dump the the firmware from the Electronic Power Steering (EPS) module from a 2021 Toyota RAV4 Prime. This is one of the new Toyotas using Autosar's new Specification of SecureOnboard Communication (SecOC) standard to authenticate CAN messages. This means that the EPS no longer accepts spoofed CAN messages from a third party device. Hopefully I'll be writing more about this in the future.

Why is this relevant?



Future product planning

- Average product development lifecycle of Electric Vehicle is 24-30 months
- Average life span of EV is 10-20 years.
- Recalls to replace ECU’s are very, very expensive
- Hardware security threats will evolve and become even more relevant

- AI/machine learning

Future product planning

- PSA Certified and SESIP lvl 3 include voltage, clock and EM FI testing
- Resistant chips are already available on the market
- Software counter measures must be added
- OEMs and type approval authorities should mandate FI resistance today

Thank you !

riscure

driving your security forward