# Leveraging RISC-V for Flexible and Adaptive Real-Time Radar Sequencing


Michael Atzmüller[1], Rainer Findenig[1], Bernhard Greslehner-Nimmervoll[1], Wolfgang Ecker[2], Daniel Große[3]

[1]Infineon Technologies Austria AG, Austria

[2]Infineon Technologies AG, Germany

[3]Johannes Kepler University Linz, Austria

Email: michael.atzmueller@infineon.com



2025 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

MUNICH, GERMANY

OCTOBER 14-15, 2025

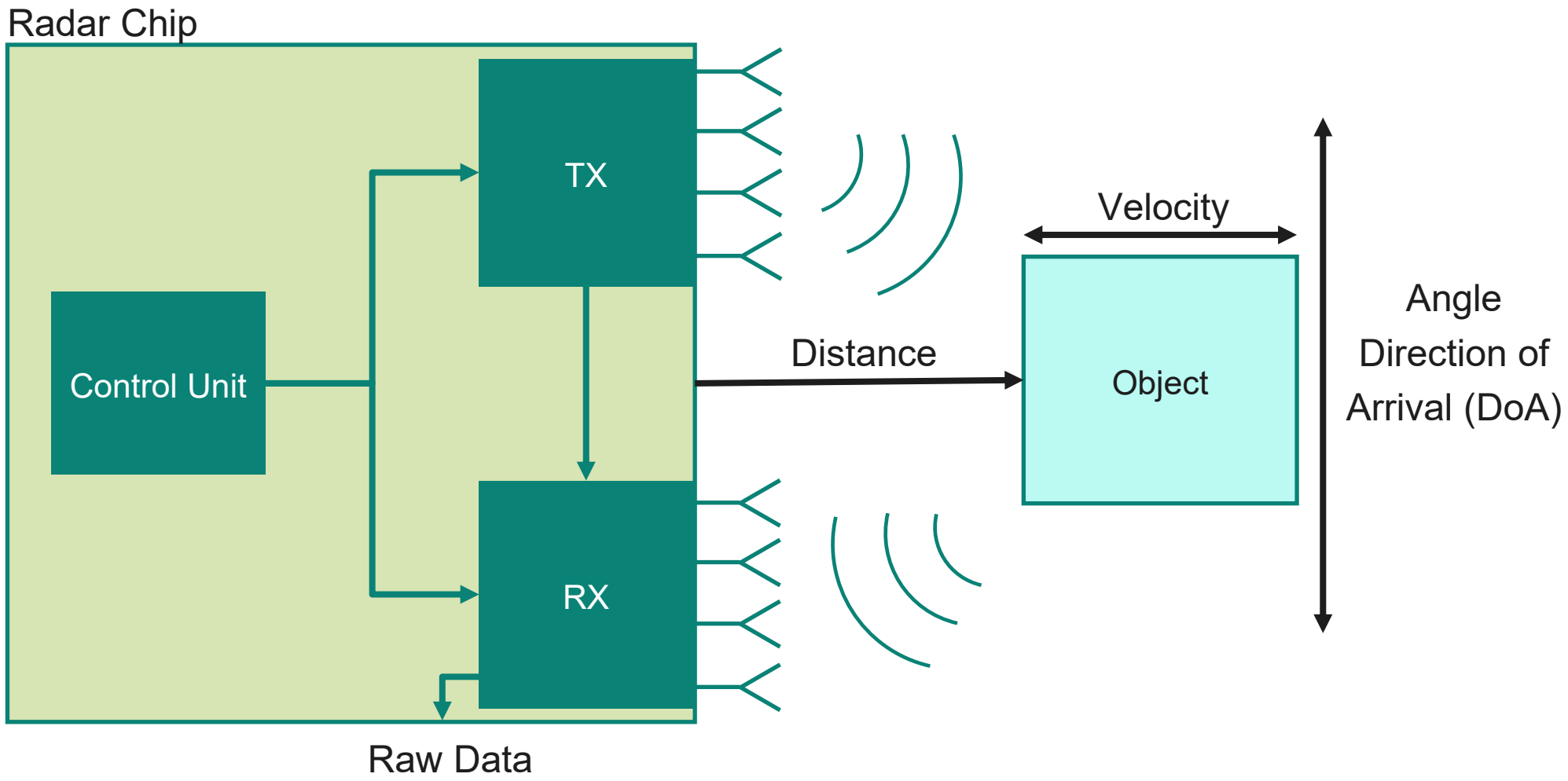

JOHANNES KEPLER UNIVERSITY LINZ

Infineon

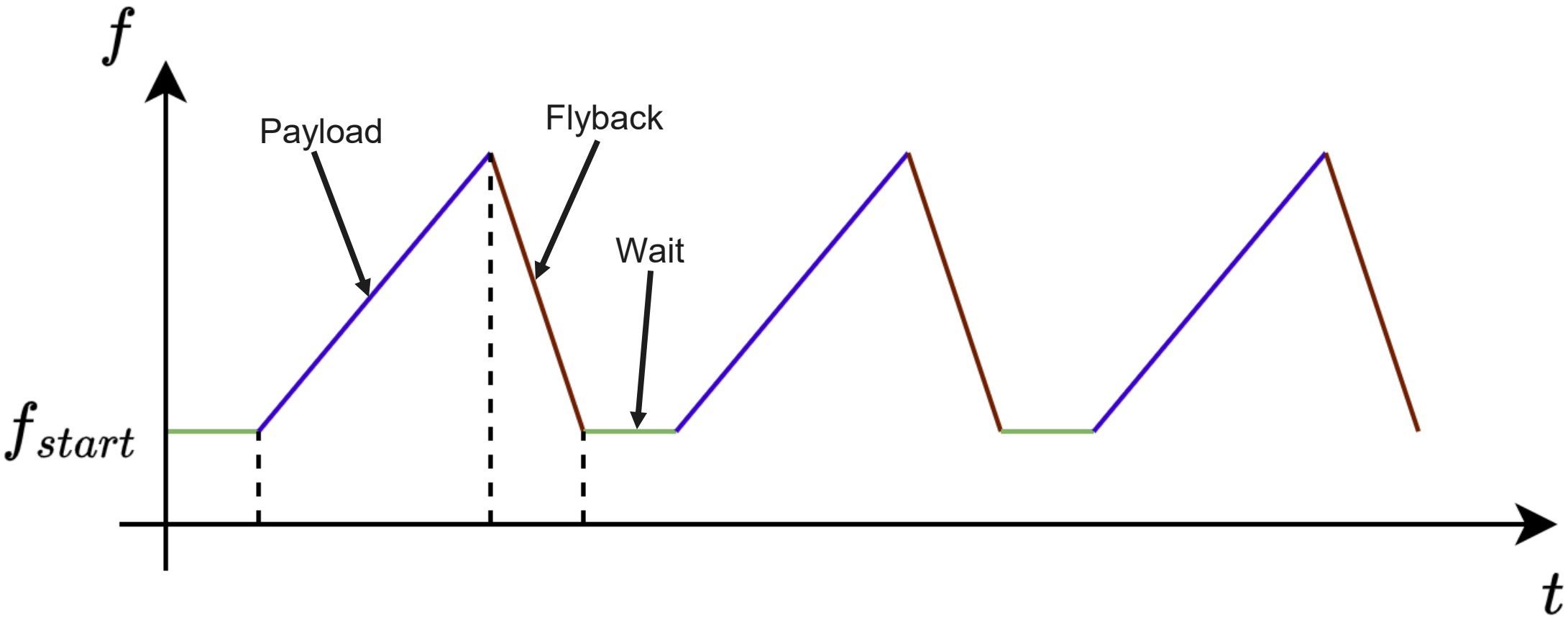
# Table of contents

# Table of contents

# Radar Intro (1)

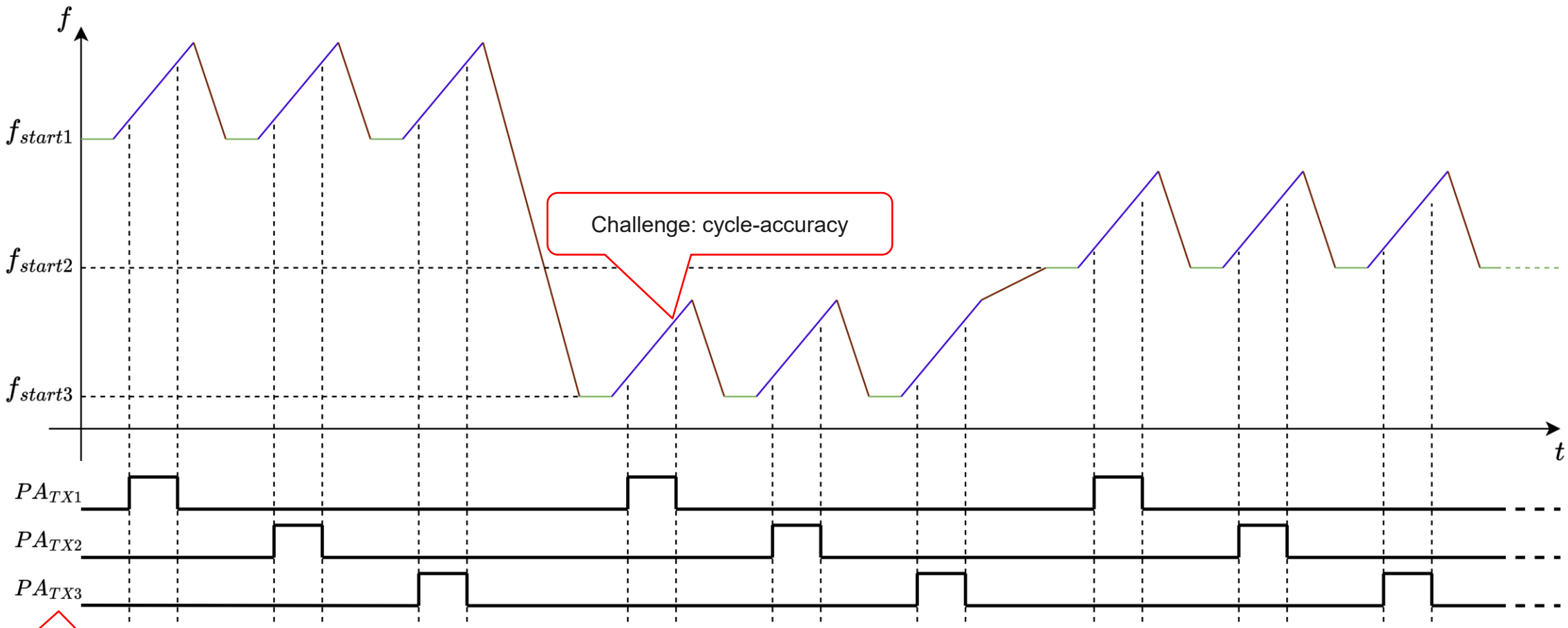


# Radar Intro (2)



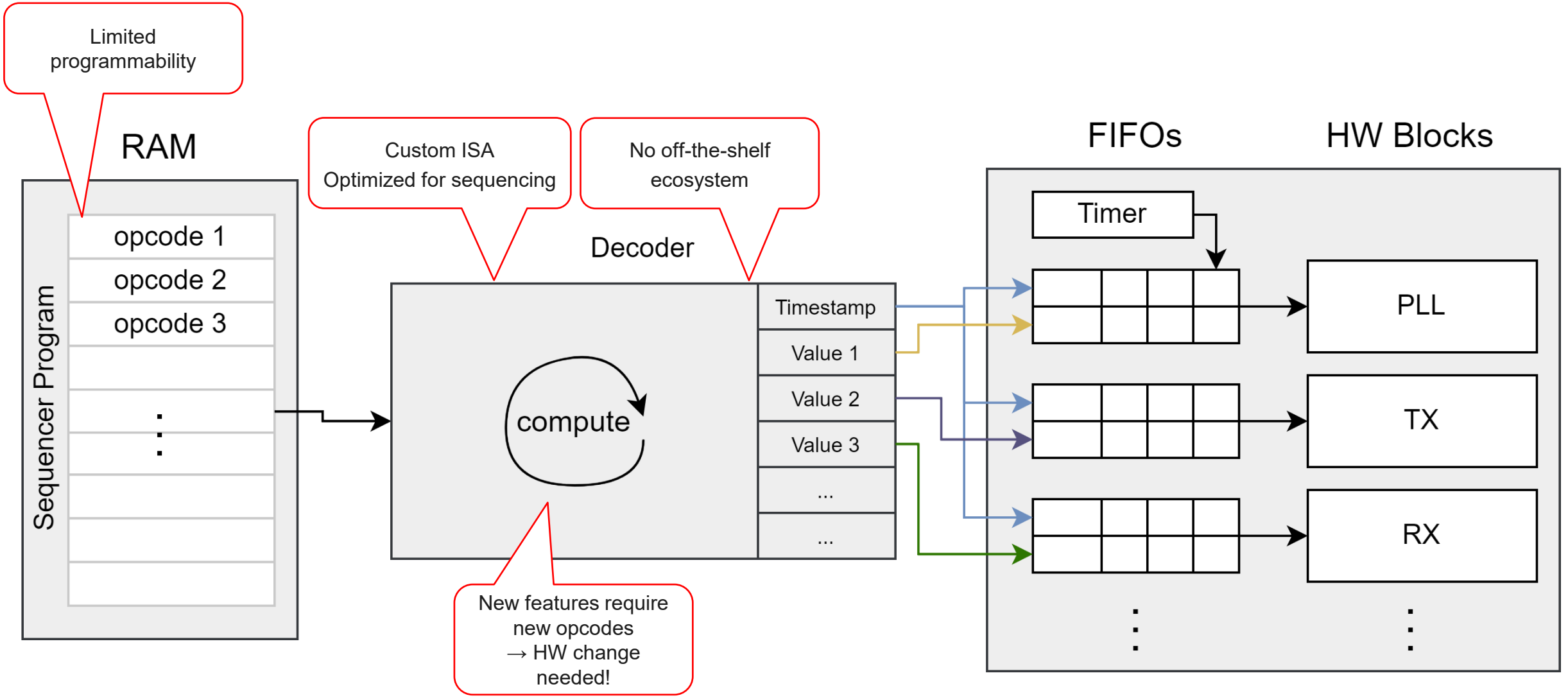
# Table of contents

# Introduction to the Problem



# Table of contents

# Doman-Specific Sequencer - Architecture



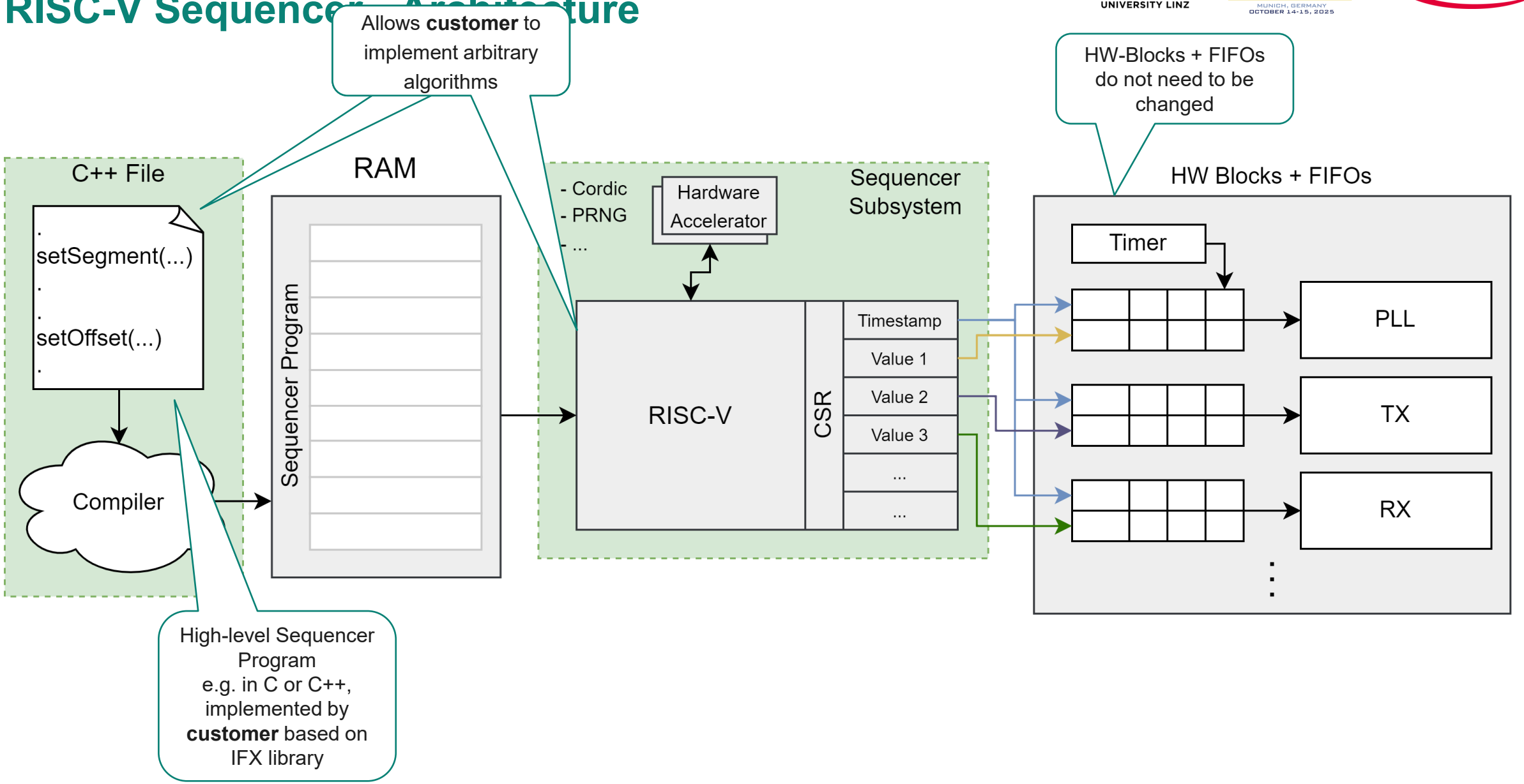
# Doman-Specific Sequencer - Programming

Example sequencer program in custom ISA

```
1  LOOP 1024
2
3  SEG f_start, f_diff, t_chirp     ; payload segment
4  SEG f_stop, -f_diff, t_flyback   ; flyback segment
5  SEG f_start, 0, ARRAY1[t_idx]    ; wait segment
6
7  MODIFY_IDX t_idx, 1
8
9  LOOP END
```

# Table of contents

# RISC-V Sequencer Architecture



# RISC-V Sequencer – Programming (1)

## Low-level CSR interaction

```
1  #define F_START    ((uint32_t) 0x9C0)
2  #define SLOPE      ((uint32_t) 0x9C1)
3  #define DURATION   ((uint32_t) 0x9C2)
4  #define TIMESTAMP  ((uint32_t) 0x9C3)
5  #define HW_READY   ((uint32_t) 0x9C4)
6
7  template <uint32_t address>
8  inline void write_csr(uint32_t const value) {
9    __asm__ volatile ("csrw␣%0,␣%1" : : "i" (address), "r"
         (value));  // input operand
10 }
11
12 template <uint32_t address>
13 inline uint32_t read_csr(){
14   uint32_t value;
15   __asm__ volatile ("csrr␣%0,␣%1"
16   : "=r" (value)       // output operand
17   : "i" (address));    // input operand
18   return value;
19 }
```

# RISC-V Sequencer – Programming (2)

Example sequencer program in C++ using low-level CSR functions

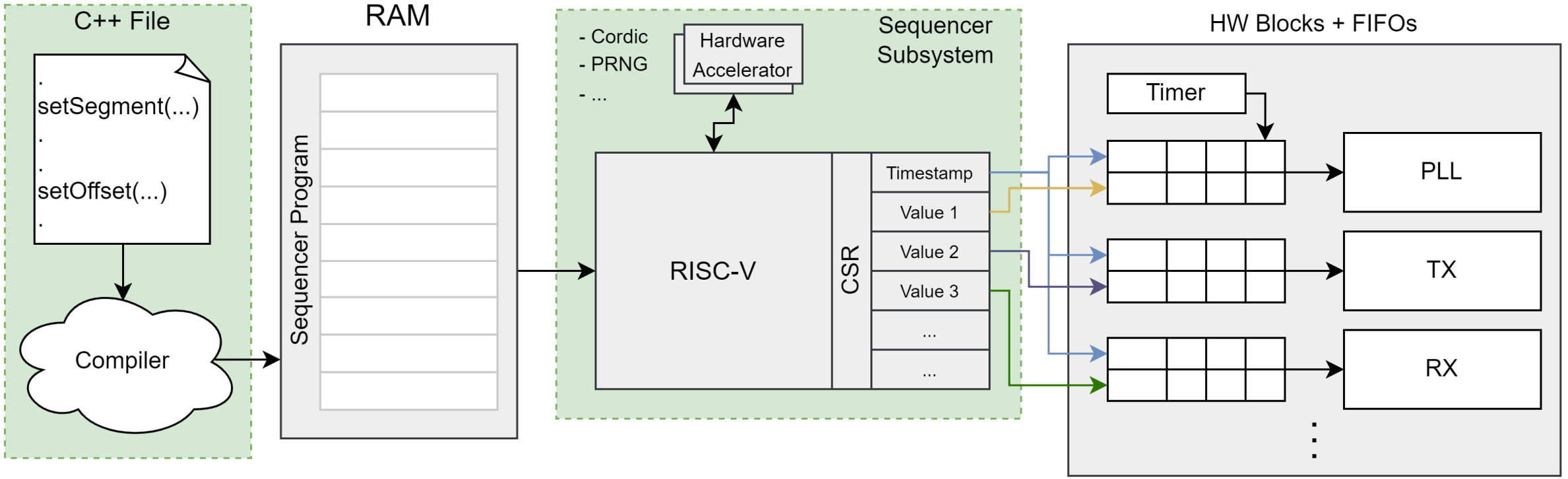
```
1  struct TRamp{uint32_t f_start; uint32_t slope; uint32_t
       duration; uint32_t timestamp; };
2
3  constexpr size_t ramp_cnt = 1024;
4  constexpr size_t segment_cnt = ramp_cnt * 3;  // times 3
         for payload, flyback and wait segment
5  constexpr TRamp ramp_params[segment_cnt] = {...};
6
7  for (int i = 0; i < segment_cnt; ++i){
8    while(!read_csr<HW_READY>());
9    write_csr<F_START>(ramp_params[i].f_start);
10   write_csr<SLOPE>(ramp_params[i].slope);
11   write_csr<DURATION>(ramp_params[i].duration);
12   write_csr<TIMESTAMP>(ramp_params[i].timestamp);
13 }
```

# Table of contents

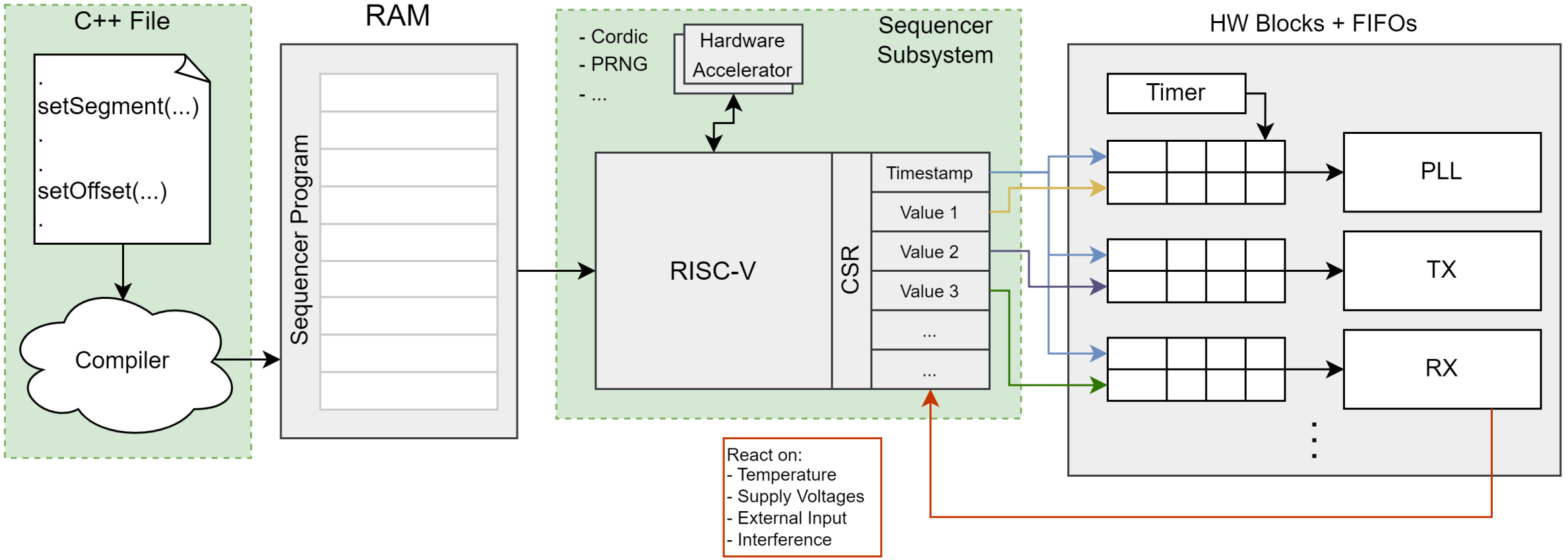
# Implementation

- Used proprietary RISC-V processor (RV32IMCZicsr)
- Verilated RISC-V to get simulation model
- Embedded into virtual prototype written in SystemC of automotive radar chip

# New Functionality – Adaptive Ramp Scenarios (1)



# New Functionality – Adaptive Ramp Scenarios (2)

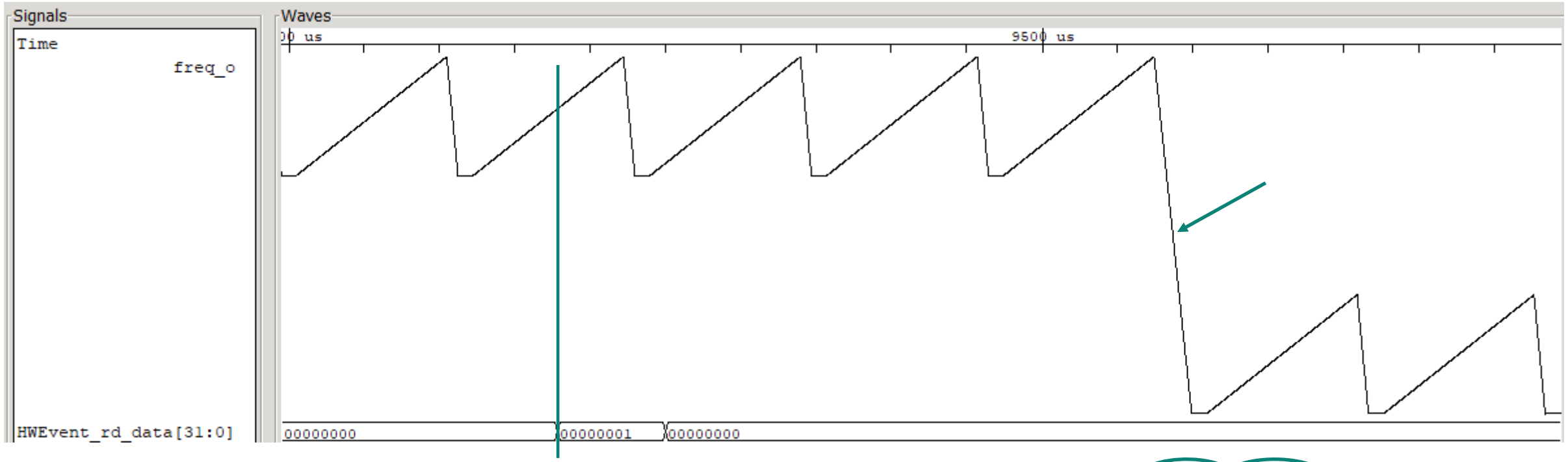


# New Functionality – Adaptive Ramp Scenarios (3)

Reacting to HW events in C++ sequencer program

```
1  RampScenario rmp(f_start, f_diff, t_chirp);
2  constexpr size_t ramp_cnt = 1024;
3
4  for (int i = 0; i < ramp_cnt; ++i){
5    rmp.set_payload_segment(i);
6    rmp.set_flyback_segment(i);
7    rmp.set_wait_segment(i);
8
9    switch (rmp.get_event()){
10     case 0: break; // no event
11     case 1: rmp.frequency_hopping(); break;
12     case 2: rmp.additional_wait(); break;
13     case 3: // critical event
14     default: rmp.abort();
15   }
16 }
```

# New Functionality – Adaptive Ramp Scenarios (4)

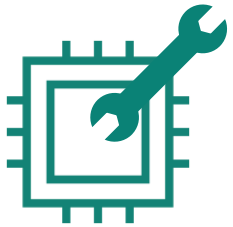



Before: Finish all ramps and obtain potential bad radar data

Now: Recover through adaptive ramp scenarios during its execution

# Results

- Less hardware changes for future chip variants

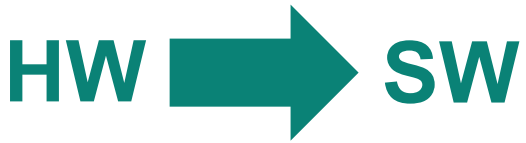


- High-level language for sequencer program comes with:
  - strong computational expressiveness
  - ecosystem available

- Performance
  - Implementation dependent
  - RISC-V sequencer slower than “old“ sequencer, but sufficient

- Area
  - RISC-V sequencer > “old“ sequencer (roughly 20-30%)

- Flexibility due to shift from HW to SW



# Leveraging RISC-V for Flexible and Adaptive Real-Time Radar Sequencing


Michael Atzmüller[1], Rainer Findenig[1], Bernhard Greslehner-Nimmervoll[1], Wolfgang Ecker[2], Daniel Große[3]

[1]Infineon Technologies Austria AG, Austria

[2]Infineon Technologies AG, Germany

[3]Johannes Kepler University Linz, Austria

Email: michael.atzmueller@infineon.com



2025 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

MUNICH, GERMANY

OCTOBER 14-15, 2025


JOHANNES KEPLER UNIVERSITY LINZ

Infineon