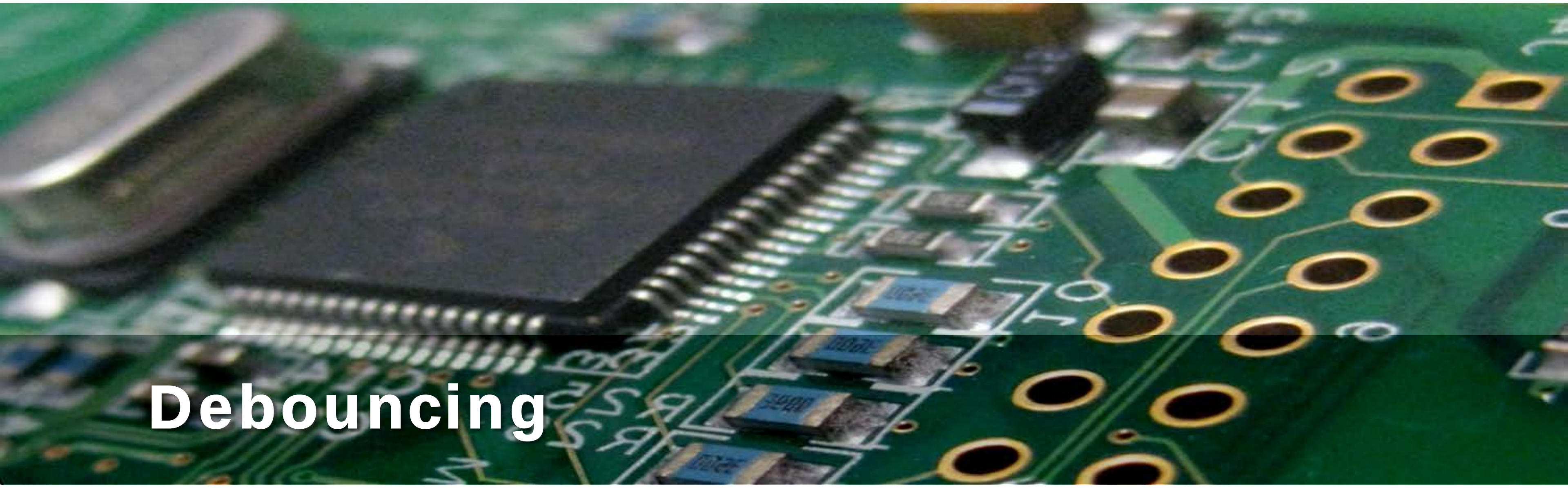
Lucerne University of
Applied Sciences and Arts

HOCHSCHULE
LUZERN

Technik & Architektur



Debouncing

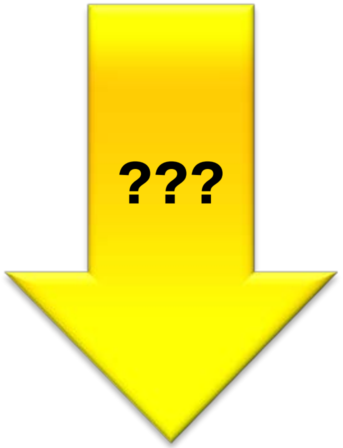
Q: “How much wood would a woodchuck chuck, if a woodchuck could chuck wood?”
A: “As much wood as a woodchuck would, if a woodchuck could chuck wood.”

Prof. Erich Styger
erich.styger@hslu.ch
+41 41 349 33 01

Skript:
Bouncing Switch

Learning Goals

- Goal
  - Debouncing keys with microcontroller
  - Detection of short and long key press
- Keys
  - Bouncing & Debouncing
- Software
  - State Machine
  - Structs
  - Callbacks
  - Event Callbacks
  - Reentrancy



Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN
Technik & Architektur

Goal: Debounce Module

- Debouncing keys
  - Single and multiple keys
- Interrupt keys and polled keys
- Reentrant
- Event/callback for short key press
- Event/callback for long key press

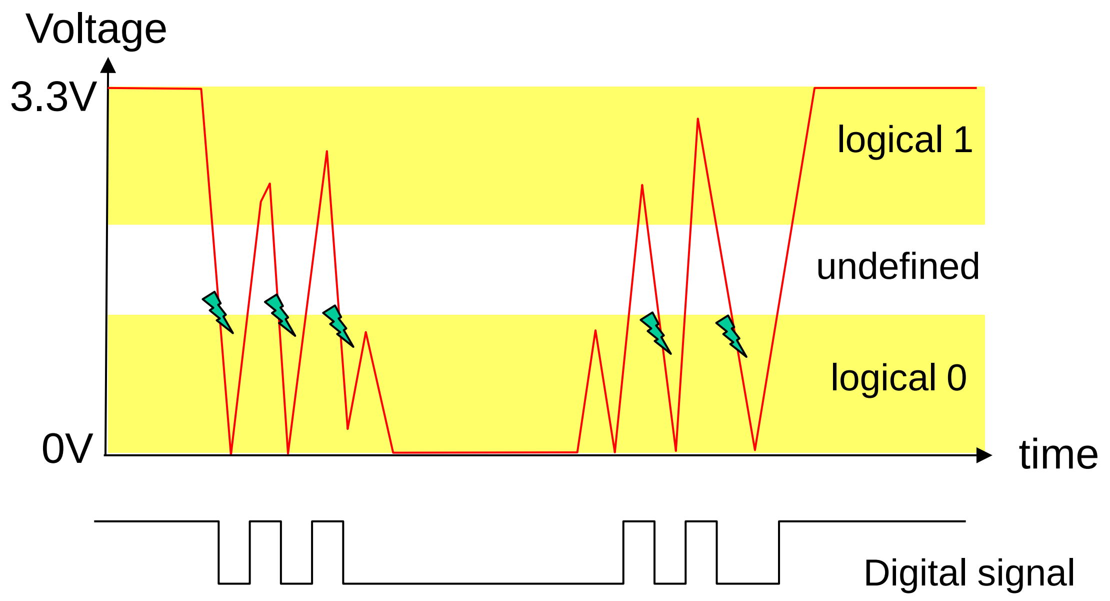
- Define special cases:
  - SW1 pressed, then SW2, SW2 released, SW1 released?
  - ???

Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN
Technik & Architektur

Example: Bouncing

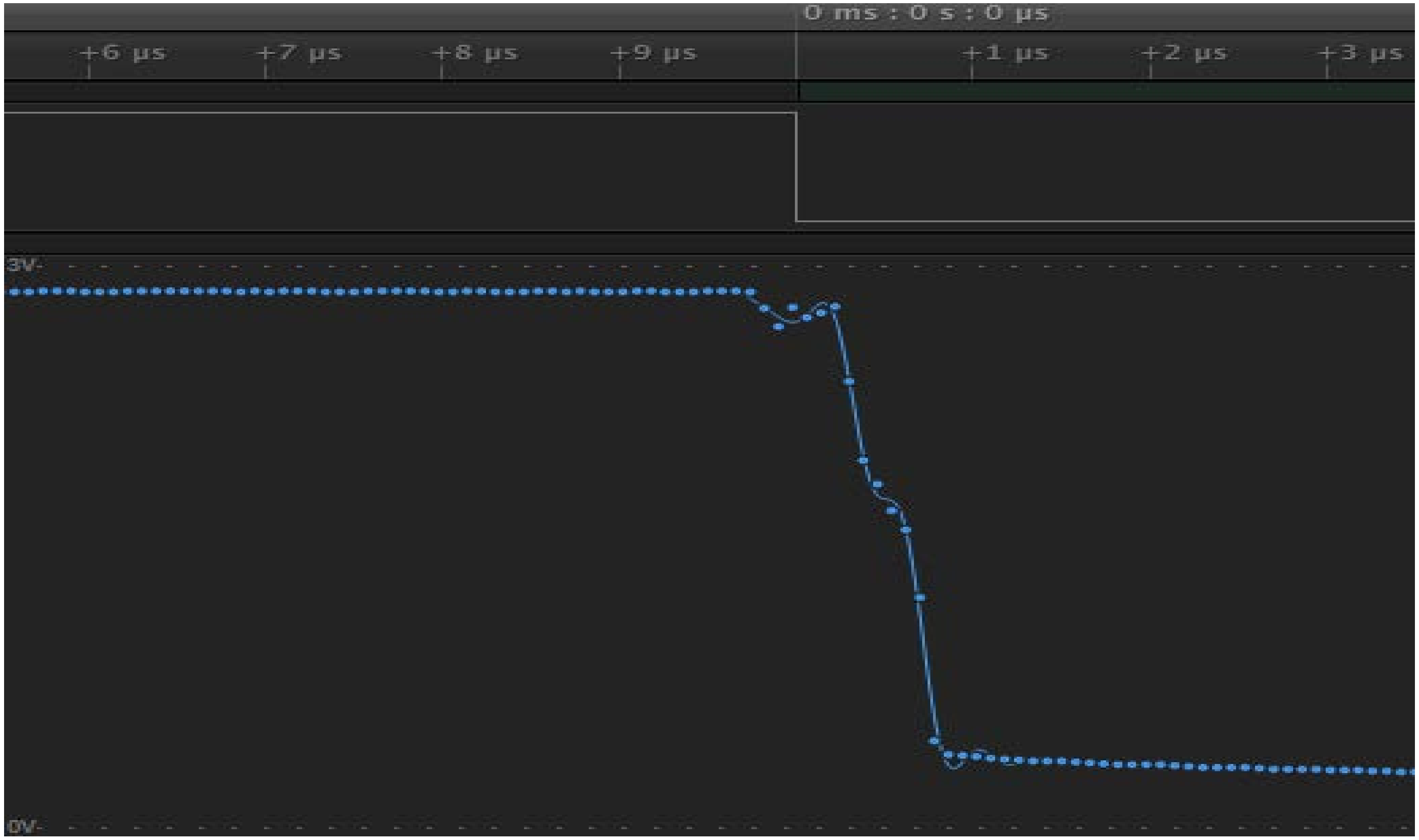
- Mechanical problem
- Contacts are bouncing several times
- Possibility of raising interrupts

it's not the best idea

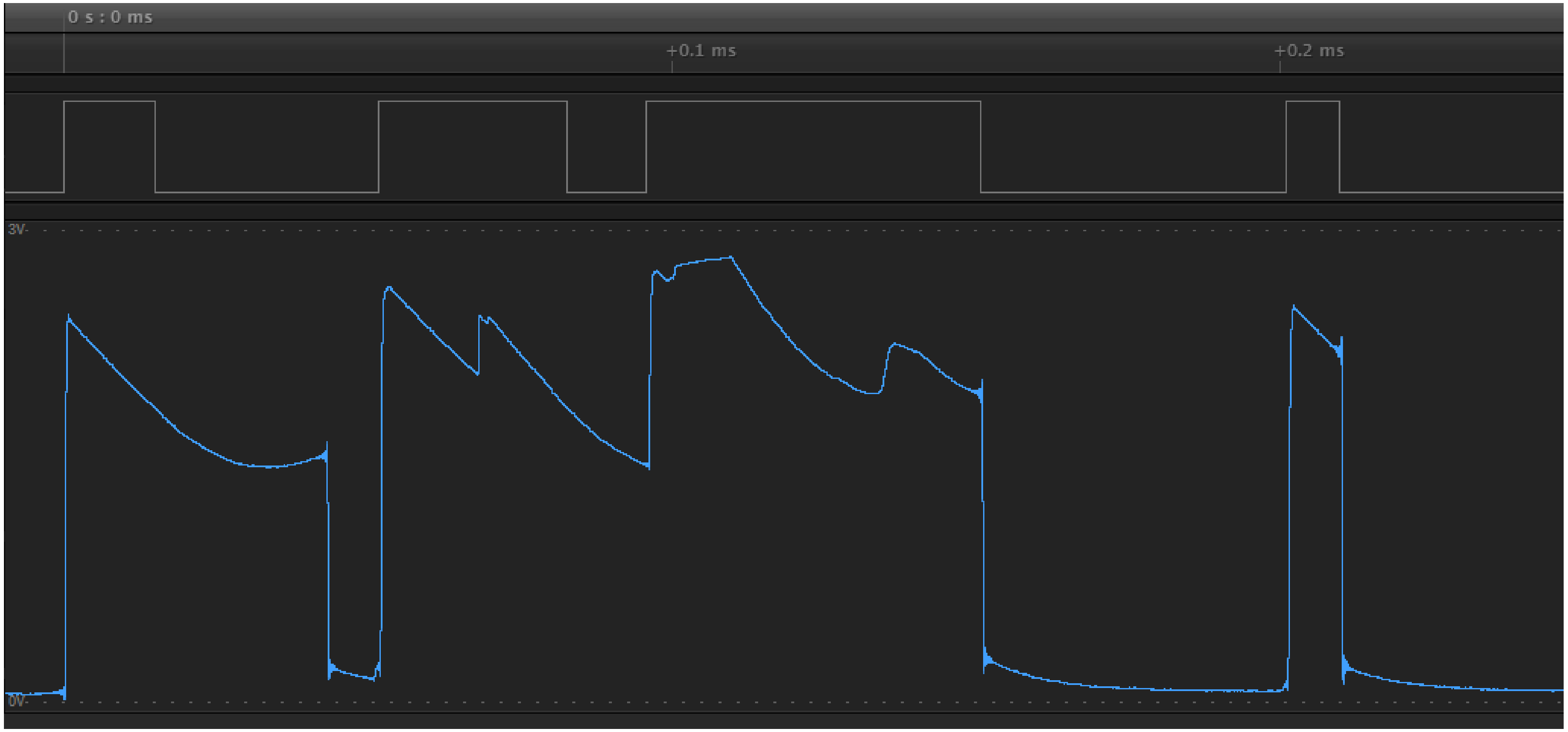


Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN
Technik & Architektur

A 'nice' one….

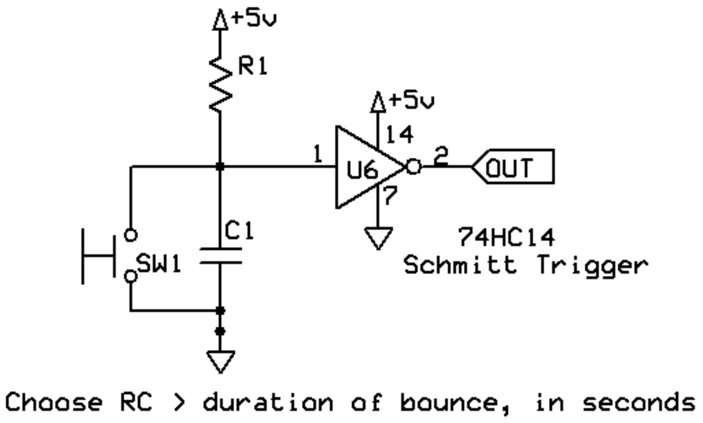


And a not so nice one…

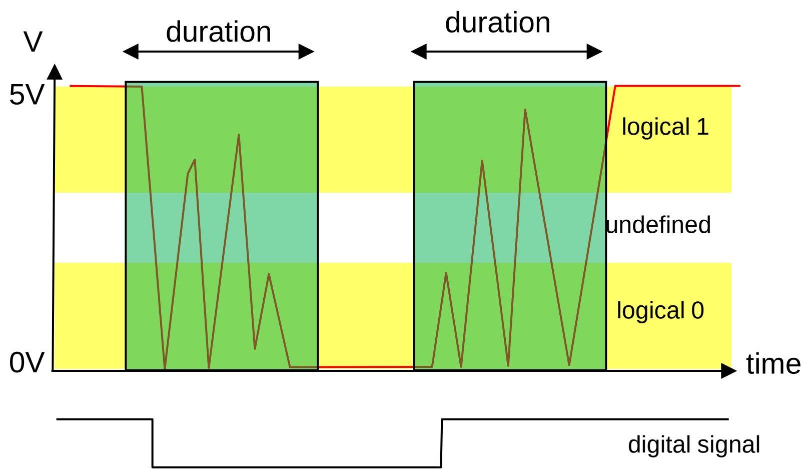


Debouncing

- Idea: Filter
    - Hardware
    - Software
- Filter duration
    - Empirical
    - Measure



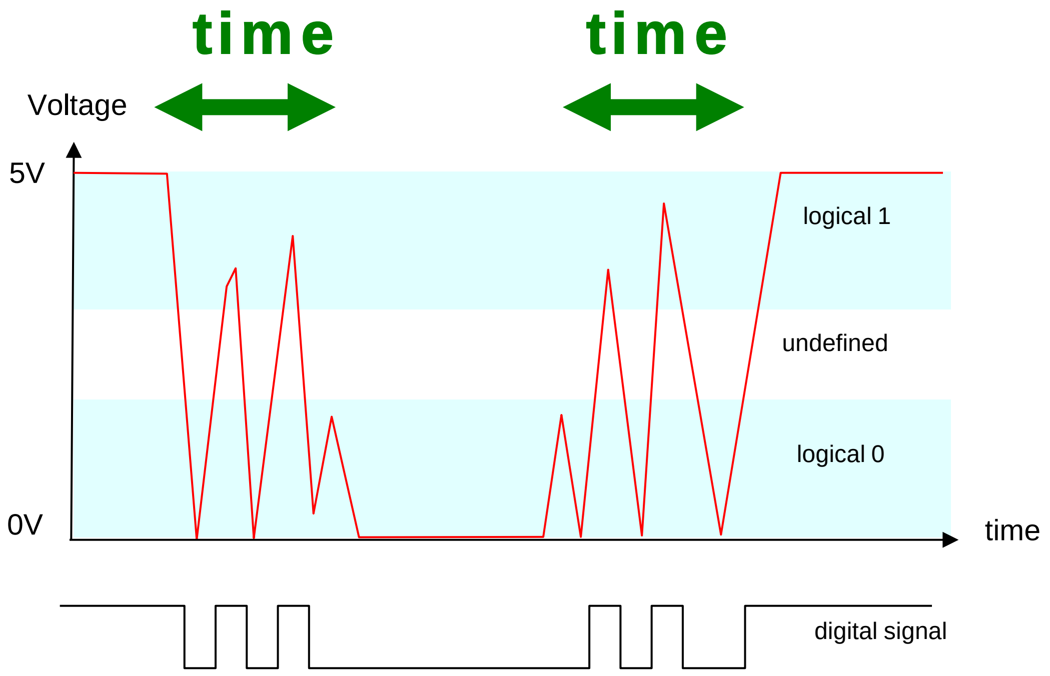
Source: Wikipedia



filter the period of time, you can do it in hardware or software.

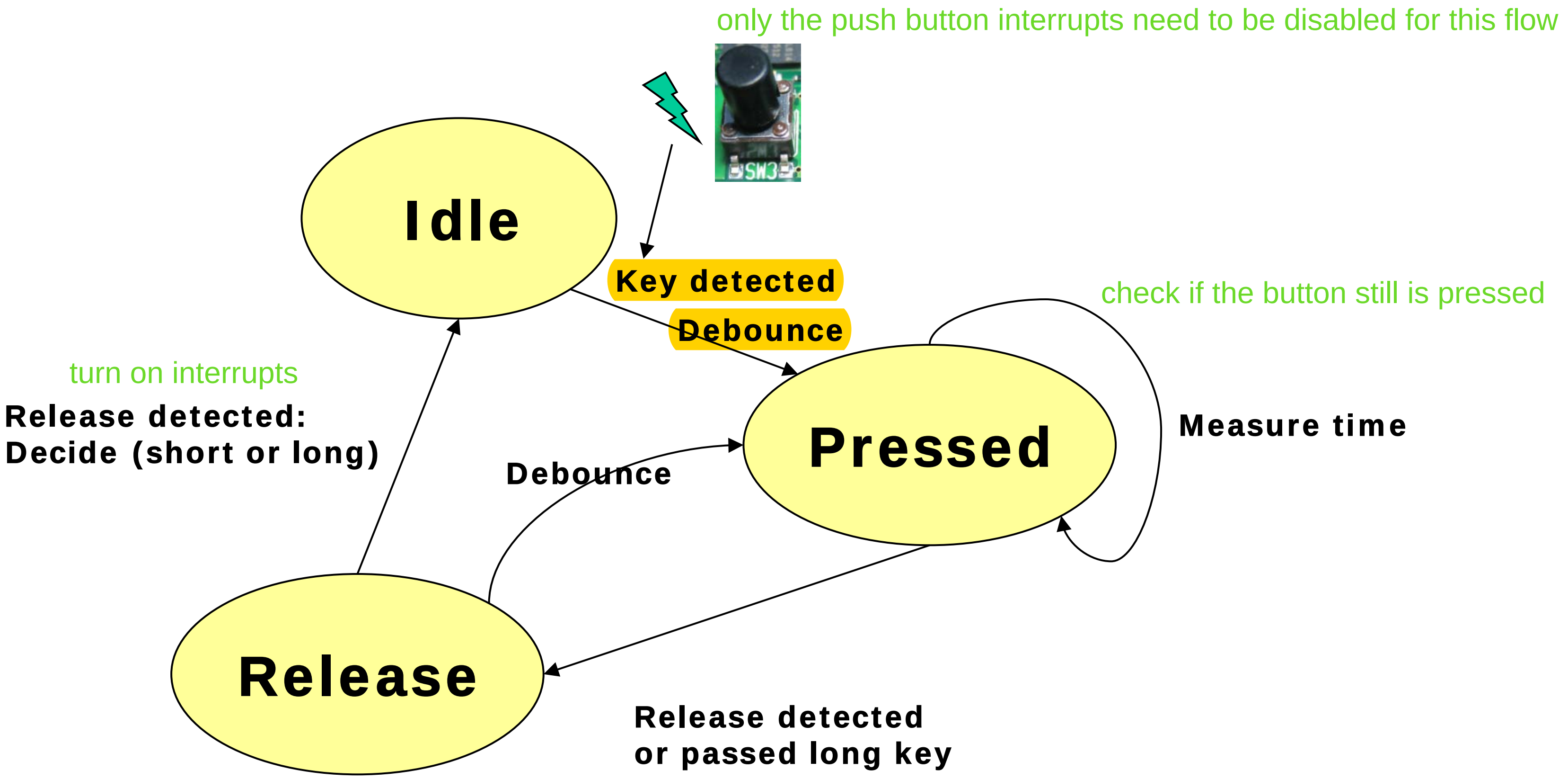
Summary: Bouncing

- Keys on platform might bounce
- Mechanical problem
- Need a filter over time
- Need relative time base or “do something in 500ms”
    - ➔State Machine & Trigger



Debouncing State Machine

- Debouncing key presses
- Measure duration of key press (long or short press)
- Finite State Machine/state diagram

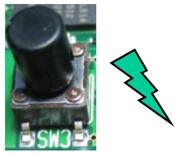


Lucerne University of Applied Sciences and Arts

HOCHSCHULE LUZERN

Technik & Architektur

Keyboard Program Flow (Interrupts)



SW1.c

```
void SW1_ISR(void) {
  ACK_ISR;
  SW1_OnInterrupt();
}
```

resets the interrupt in the hardware

Event.c

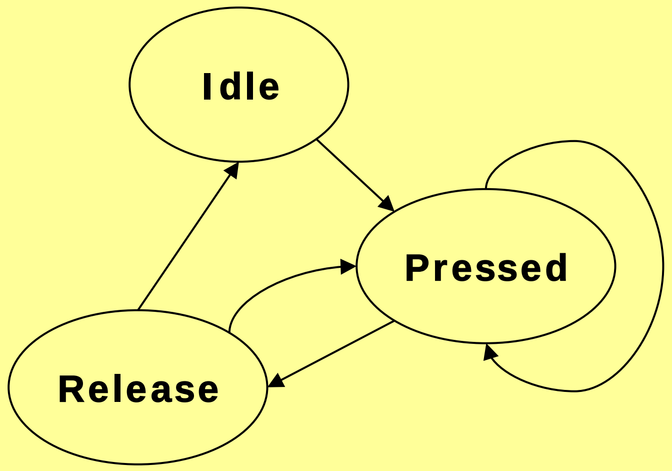
```
void SW1_OnInterrupt(void) {
  KEY_OnInterrupt(BTN1);
}
```

the difference: only if you have a event and if you aren't already debouncing

Key.c

```
void KEY_OnInterrupt(btn) {
  SW1_DisableInterrupts();
  KEYDBNC_Process();
}
```

Debounce.c



Lucerne University of Applied Sciences and Arts

HOCHSCHULE LUZERN

Technik & Architektur

Keyboard Program Flow (Polling)

App.c

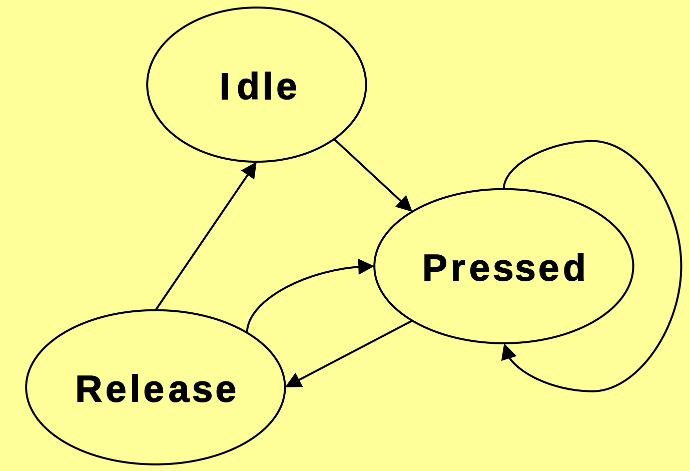
```
void APP_Run(void) {
  for(;;) {
   KEY_Scan();
  }
}
```

Key.c

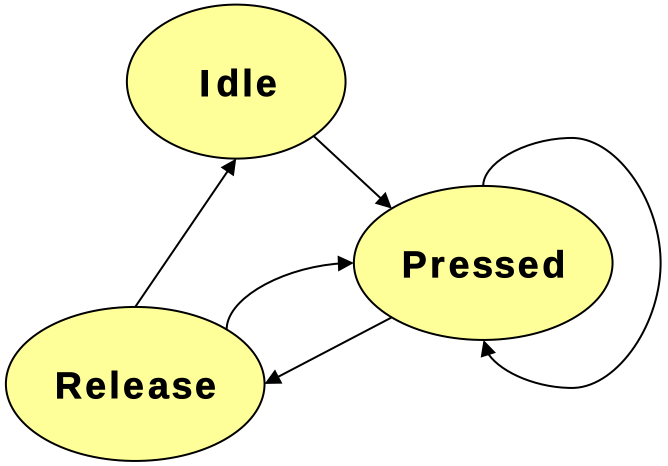
```
void KEY_Scan(void) {
  if (KEY1_Get()) {
    KEYDBNC_Process();
  }
}
```

if not already debounced:

Debounce.c



Debounce State Machine

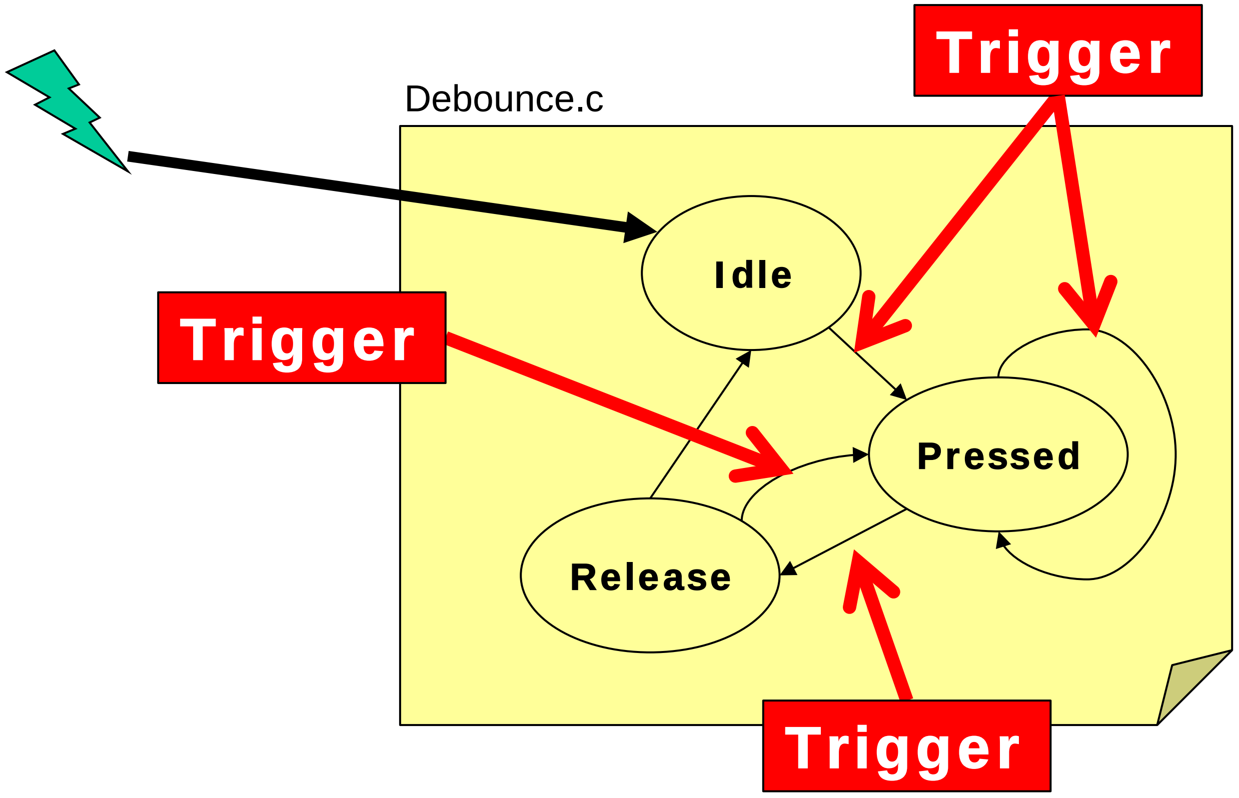


```
/*! \brief States of the key detection state machine. */
typedef enum {
  DBNC_KEY_IDLE = 0,     /*<! Initial idle state */
  DBNC_KEY_PRESSED,      /*<! Key pressing detected, see if it is a long key */
  DBNC_KEY_RELEASE,      /*<! got a key pressed, wait for release */
} DBNC_KeyStateKinds;
```

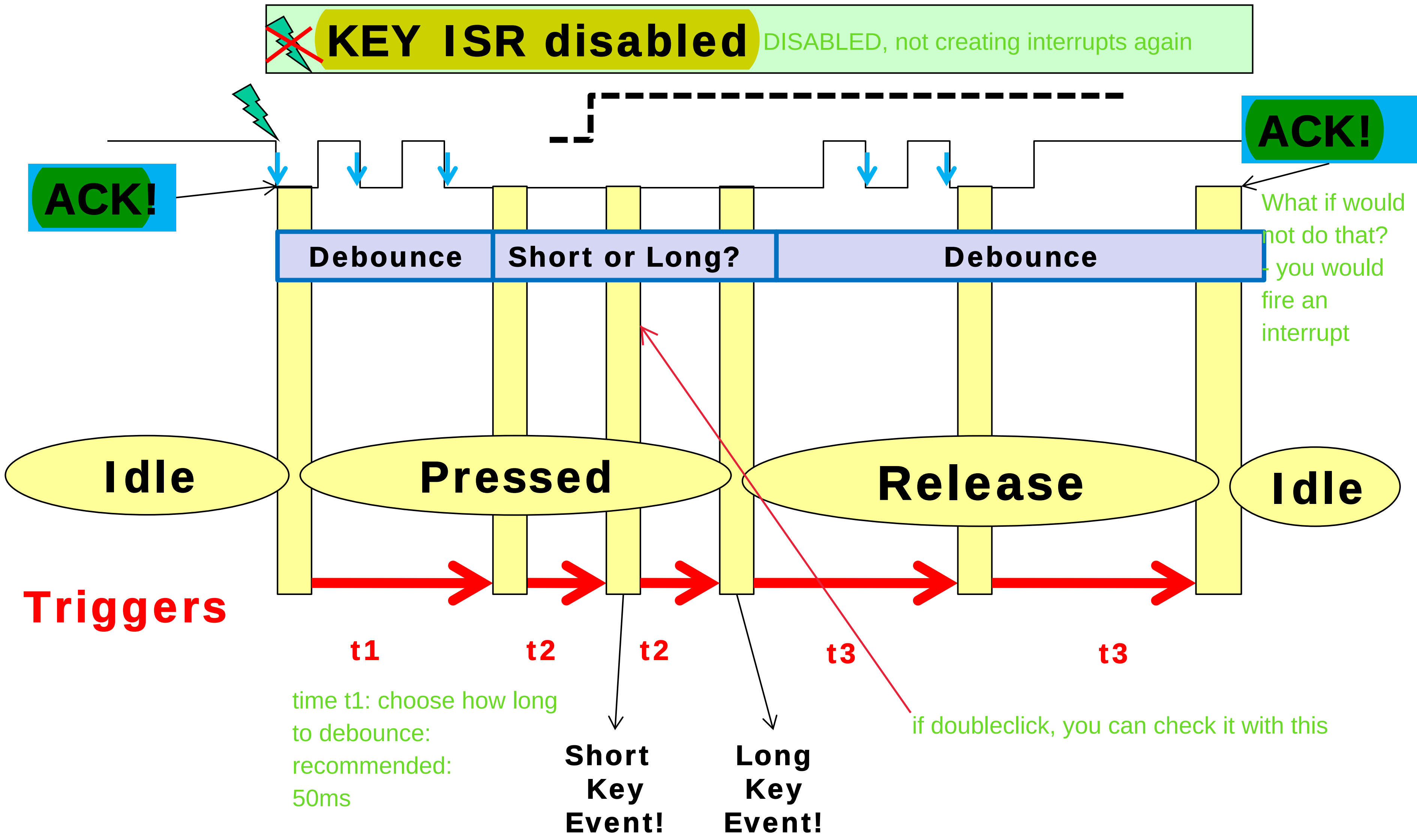
Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN
Technik & Architektur

State Machine Details

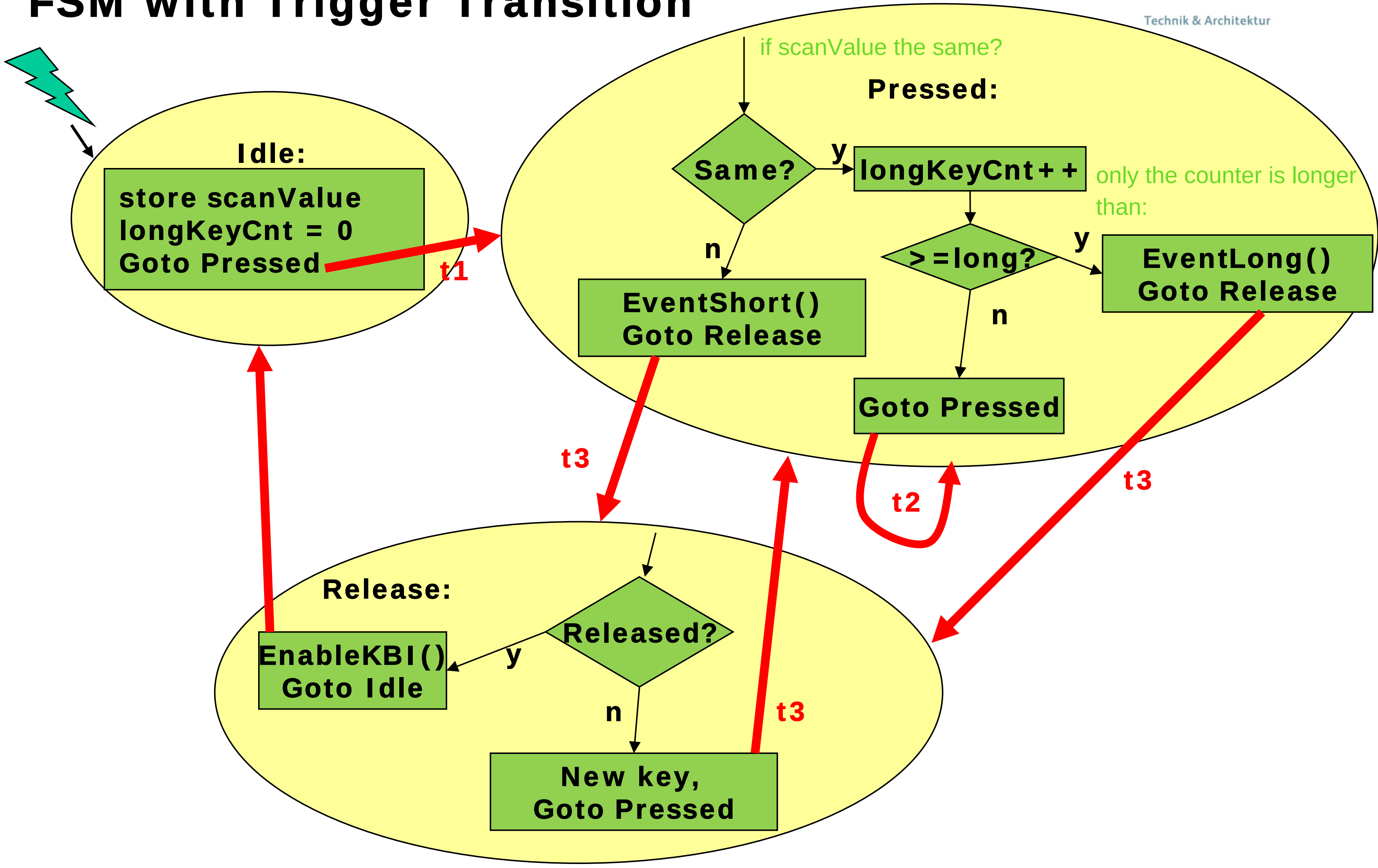
- State machine entered through interrupt
- Cannot stay in state machine!
- Use Trigger to re-enter



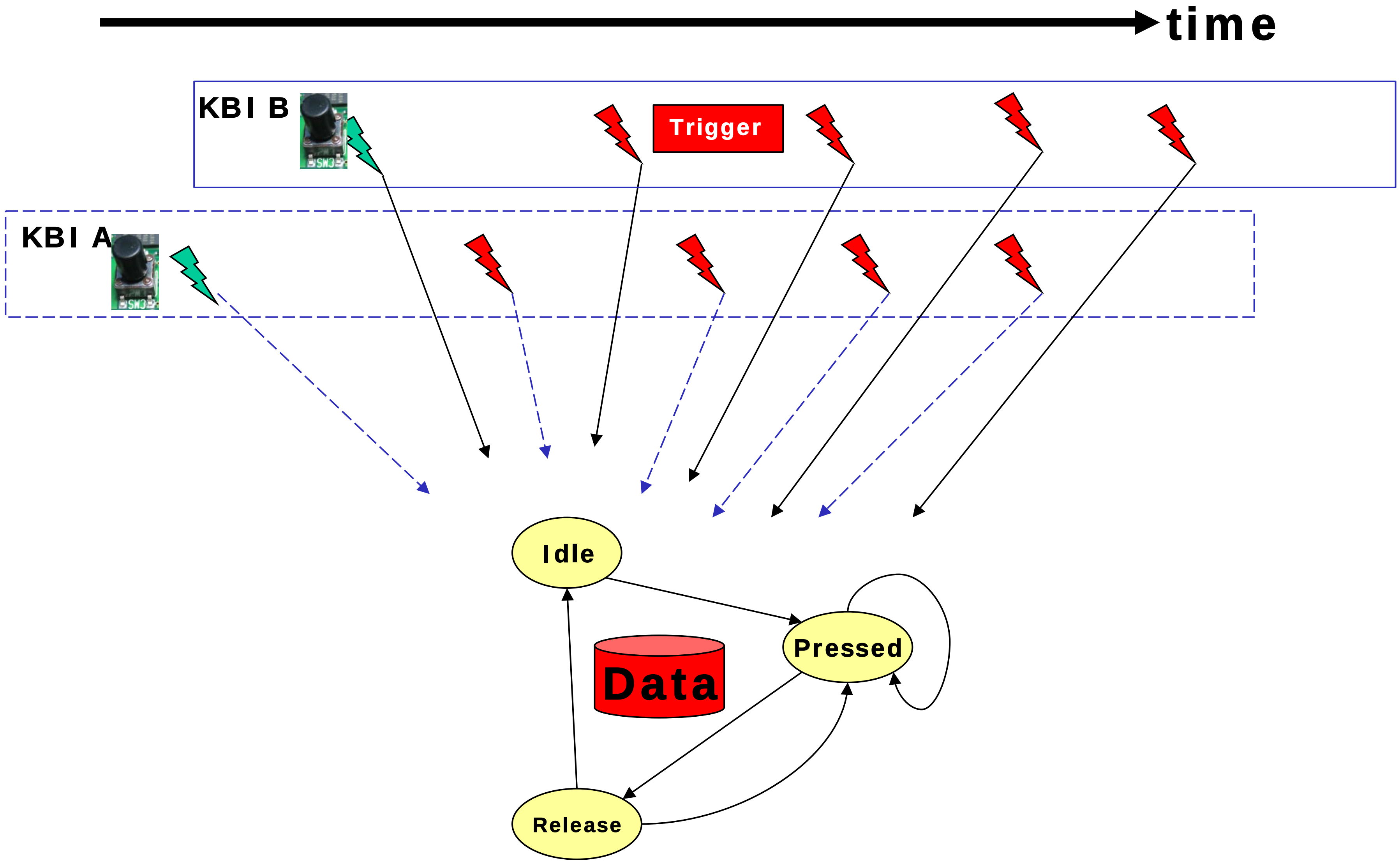
Debounce - FSM


KEY ISR disabled
DISABLED, not creating interrupts again
ACK!
ACK!
What if would not do that?
- you would fire an interrupt
Debounce
Short or Long?
Debounce
Idle
Pressed
Release
Idle
Triggers
t1
t2
t2
t3
t3
time t1: choose how long to debounce: recommended: 50ms
Short Key Event!
Long Key Event!
if doubleclick, you can check it with this

FSM with Trigger Transition



Reentrancy + Interface Problem?

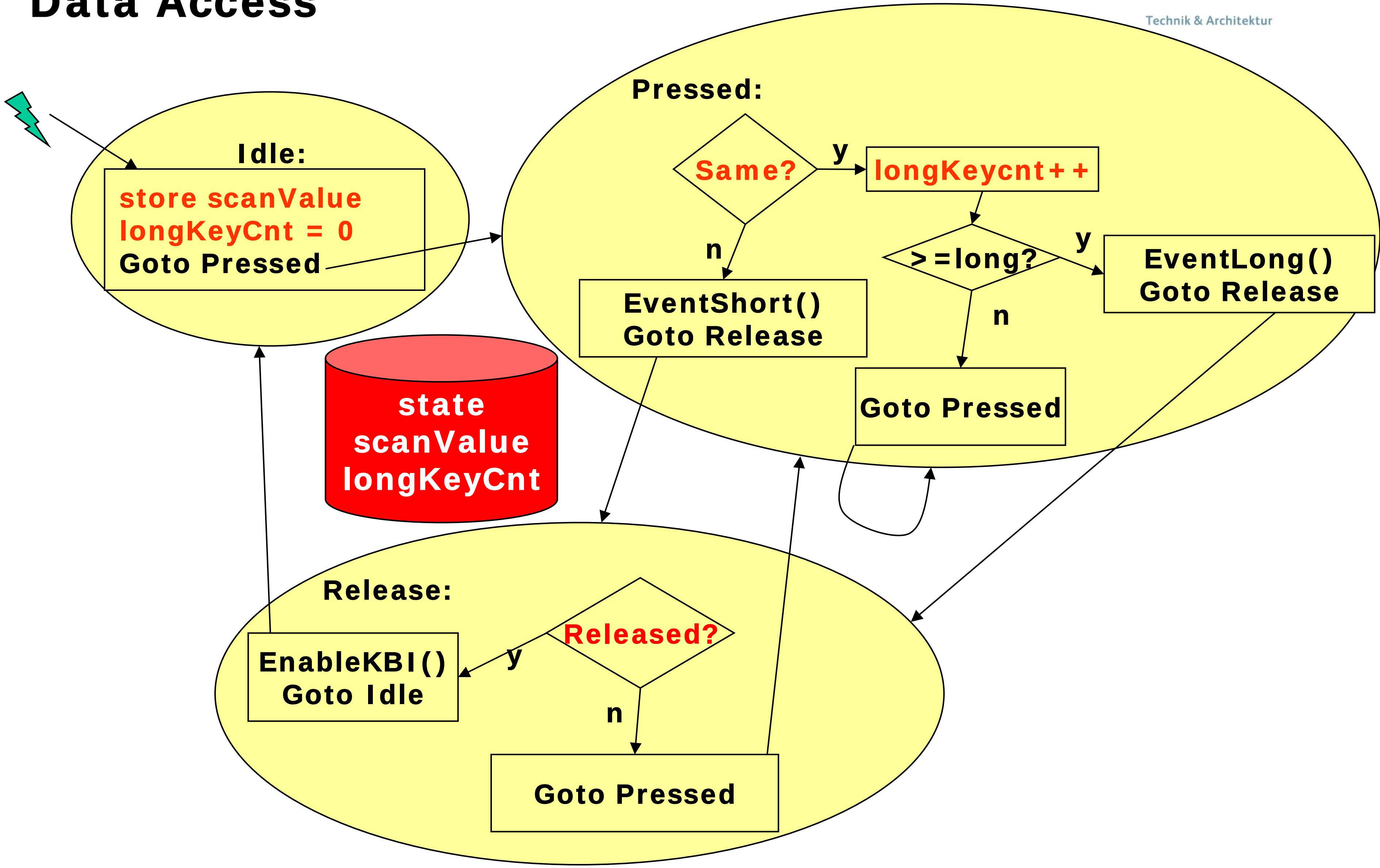


Lucerne University of Applied Sciences and Arts

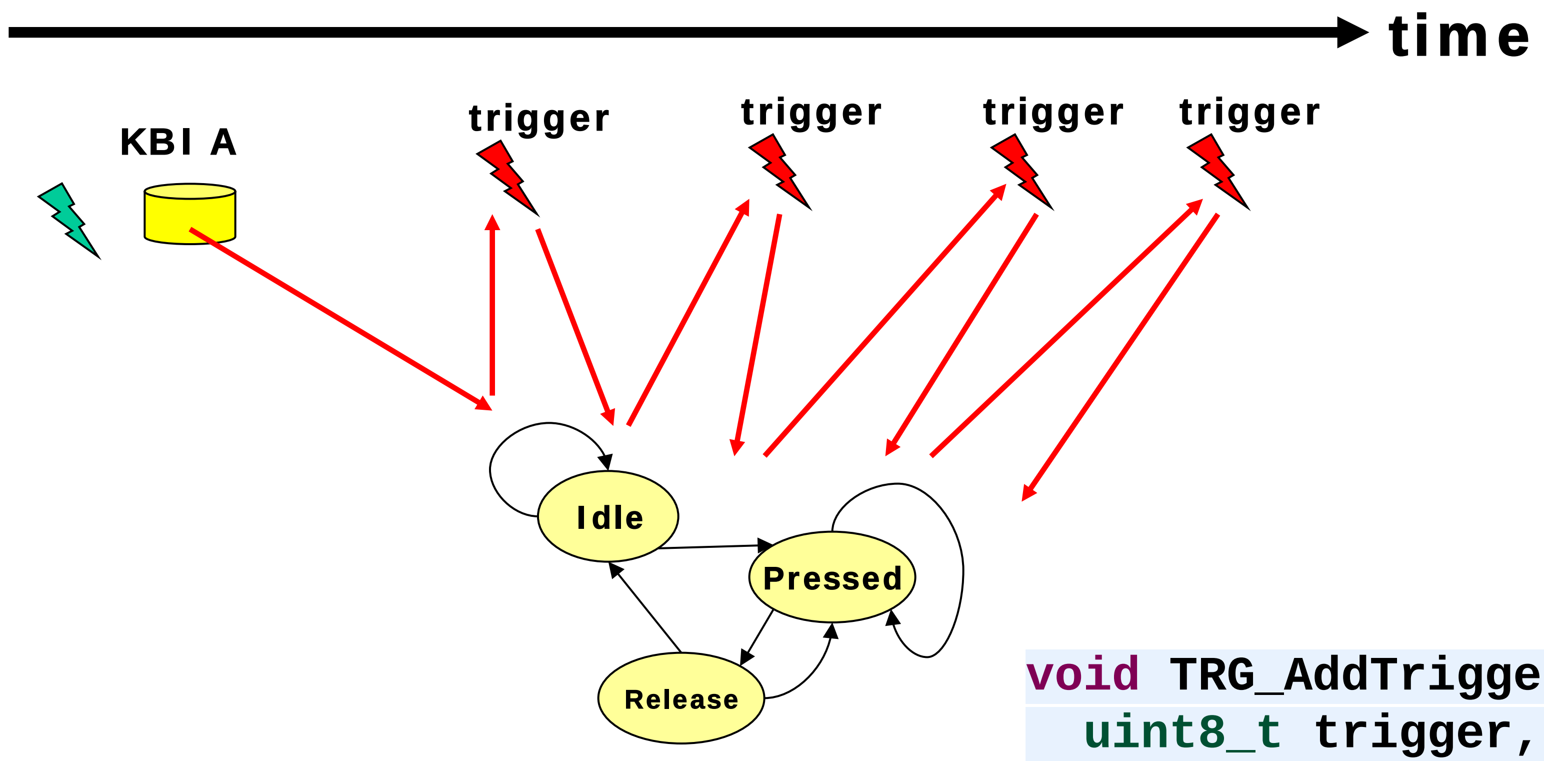
HOCHSCHULE LUZERN

Technik & Architektur

Data Access

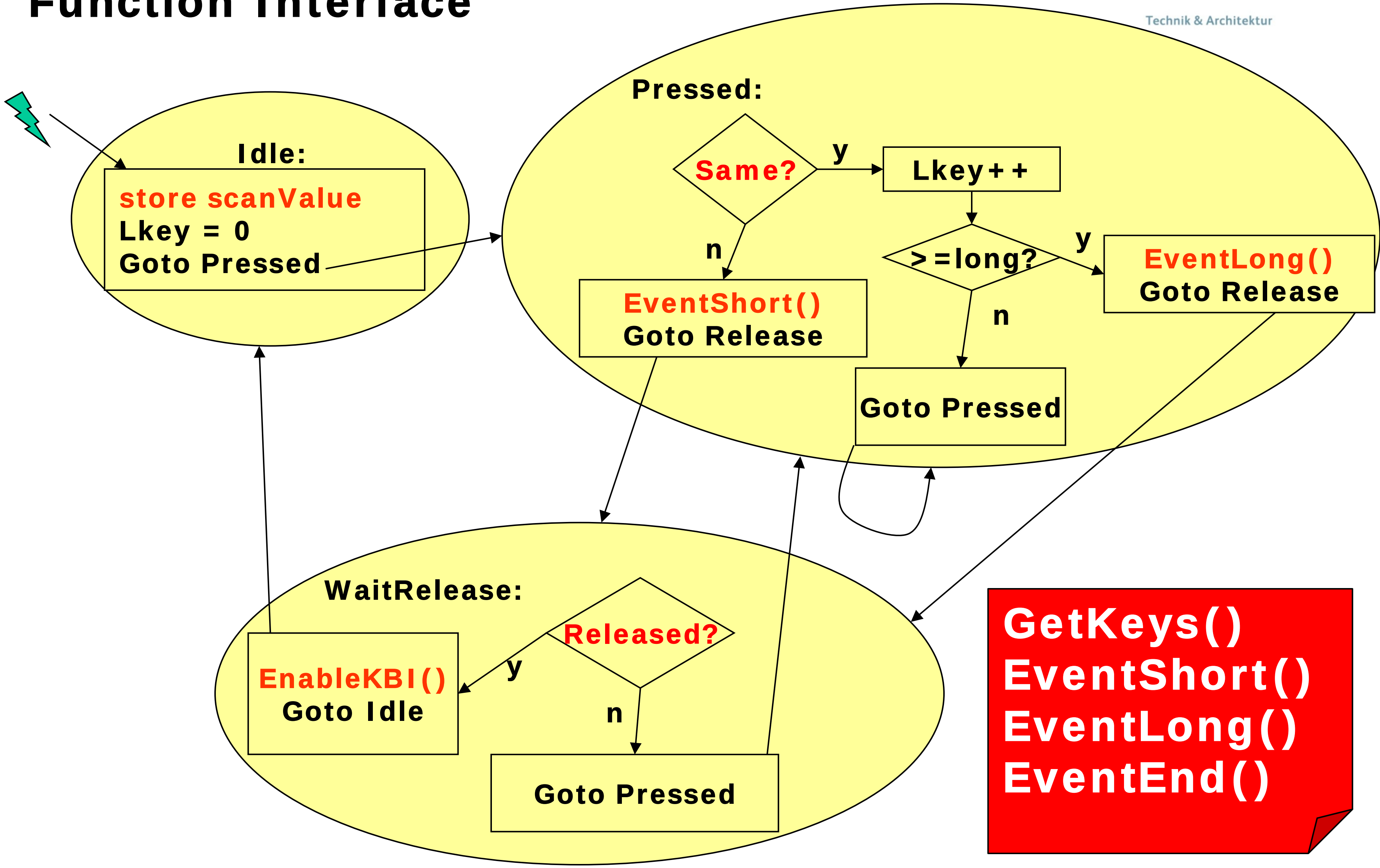


Passing Data through Trigger Events



```
void TRG_AddTrigger(
  uint8_t trigger,
  uint16_t triggerTicks,
  TRG_Callback callback,
  void *data);
```

Function Interface



GetKeys()
EventShort()
EventLong()
EventEnd()

Data Passing with Trigger

```
typedef uint8_t DBNC_KeySet;
typedef DBNC_KeySet (*DBNC_GetKeysFn)(void);
typedef void (*DBNC_EventCallback)(DBNC_EventKinds event,
DBNC_KeySet keys);

typedef struct {
  DBNC_GetKeysFn getKeys;
  DBNC_EventCallback onDebounceEvent;
  DBNC_KeyStateKinds state;
  DBNC_KeySet scanValue;
  TRG_TriggerKind trigger;
  uint16_t debounceTicks;
  uint16_t longKeyTicks;
} DBNC_FSMData;
```

we need a trigger, because every debouncing can use their own trigger

Lucerne University of
Applied Sciences and Arts

HOCHSCHULE
LUZERN

Technik & Architektur

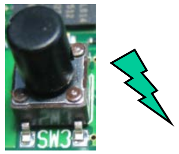
Example Configuration (KeyDebounce)

```
static void KEYDBNC_OnDebounceEvent(DBNC_EventKinds event, DBNC_KeySet keys) {
  if (event==DBNC_EVENT_PRESSED && (keys&(1<<0))) {
    EVNT_SetEvent(EVNT_SW1_PRESSED);
  } …
  if (event==DBNC_EVENT_END) {
    KEY_EnableInterrupts();
  }
}

static DBNC_FSMData KEYDBNC_FSMdata = {
  KEYDBNC_GetKeys, /* returns bit set of pressed keys */
  KEYDBNC_OnDebounceEvent, /* event called */
  DBNC_KEY_IDLE, /* state machine state */
  0, /* key scan value */
  0, /* long key count */
  …
};

void KEY_OnInterrupt(void) {
  KEY_DisableInterrupts();
  DBNC_Process(&KBD_FSMdata); /* starts the state machine */
}
```

reenable the interrupt here!

Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN
Technik & Architektur

Timing!



Ideal: Scan here!

KB1.c

```
void SW1_ISR(void) {
  ACK_ISR;
  SW1_OnInterrupt();
}
```

Event.c

```
void SW1_OnInterrupt(void) {
  KEY_OnInterrupt();
}
```

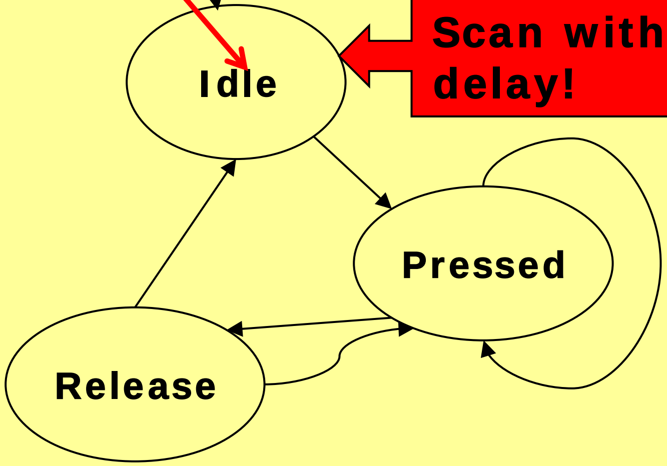
Keys.c

```
void KEY_OnInterrupt(void) {
  KEY_DisableInterrupts();
  DBNC_Process(&KEY_FSMData);
}
```

Debounce.c

could be you get an interrupt

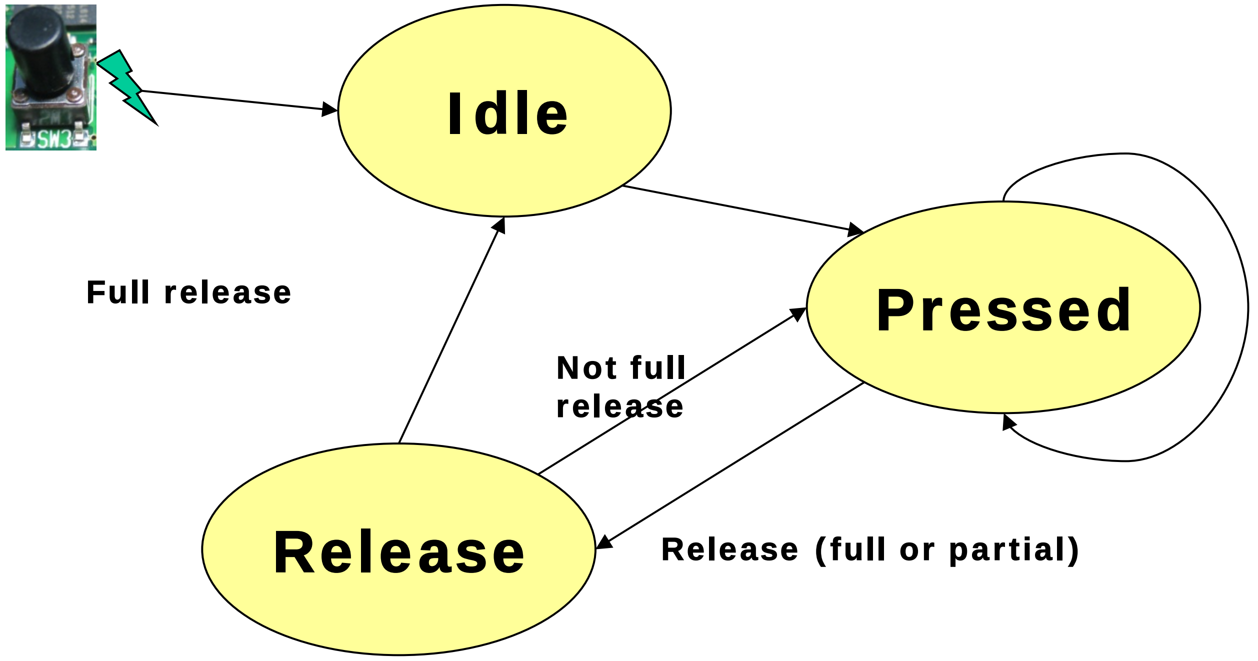
Scan with delay!



Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN
Technik & Architektur

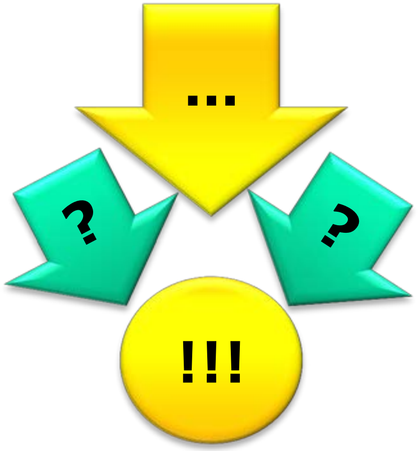
Improved FSM

- Either Short or Long key message (not both for a single press)
- 'Interclicks'
  - SW1 long -> SW1+SW2 long -> SW2 released -> SW1 long -> SW1 release



Summary

- Debouncing
- Reentrancy
- Data Pointer and Callbacks with struct
    - Data
    - Callbacks
    - Event Methods
- Another way to use a FSM ☺



Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN
Technik & Architektur

Lab: Debouncing

- Debounce.c, Debounce.h
- KeyDebounce.c, KeyDebounce.h
- Extend Keys.c
- Debouncing using as state machine
  - short key press SW0
    ➔ Create/handle event
  - Long key press SW0
    ➔ create/handle event
  - Reentrancy!